Deep learning methods for scene text detection

Liu, Zichuan

2020

Liu, Z. (2020). Deep learning methods for scene text detection. Doctoral thesis, Nanyang Technological University, Singapore.

<https://hdl.handle.net/10356/142533>

<https://doi.org/10.32657/10356/142533>

This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License (CC BY-NC 4.0).

Downloaded on 26 Apr 2023 18:29:47 SGT

**NANYANG
TECHNOLOGICAL
UNIVERSITY**

SINGAPORE

DEEP LEARNING METHODS FOR SCENE TEXT DETECTION

LIU ZICHUAN

School of Electrical and Electronic Engineering

August 13, 2020

Deep Learning Methods for Scene Text Detection

LIU ZICHUAN

School of Electrical and Electronic Engineering

A thesis submitted to the Nanyang Technological University

in partial fulfillment of the requirement for the degree of

Doctor of Philosophy

August 13, 2020

Statement of Originality

I hereby certify that the work embodied in this thesis is the result of original research, is free of plagiarised materials, and has not been submitted for a higher degree to any other University or Institution.

19 Aug 2019

.....

Date

Zichuan Liu

.....

Liu Zichuan

Supervisor Declaration Statement

I have reviewed the content and presentation style of this thesis and declare it is free of plagiarism and of sufficient grammatical clarity to be examined. To the best of my knowledge, the research and writing are those of the candidate except as acknowledged in the Author Attribution Statement. I confirm that the investigations were conducted in accord with the ethics policies and integrity standards of Nanyang Technological University and that the research data are presented honestly and without prejudice.

2019/Aug/19

.....
Date

.....
Goh Wang Ling

Authorship Attribution Statement

This thesis contains material from 3 papers published in the following peer-reviewed journal and conferences where I was the first author.

Chapter 3 is related to con paper: Z. Liu, G. Lin, S. Yang, J. Feng, W. Lin, and W. L. Goh, “Learning marko clustering networks for scene text detection,” The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2018, pp. 6936-6944

The contributions of the co-authors are as follows:

- Prof. Goh Wang Ling and Prof. Lin Guoshe g supervised me on the overall research direction and edited the manuscript drafts.
- I proposed the idea, developed the algorithm, conduct the experiments and prepared the manuscript drafts.
- Dr. Yang Sheng, Prof. Feng Jiashi and Prof. Lin Weisi helped me on proofreading.

Chapter 4 is related to journal paper: Z. Liu, W. L. Goh, and G. Lin, “Bottom-up Scene Text Detection with Markov Clustering Networks,” International Journal on Computer Vision, pp.1-24, 2020

The contributions of the co-authors are as follows:

- Prof. Goh Wang Ling and Prof. Lin Guosheng supervised me on the overall research direction and edited the manuscript drafts.
- I proposed the idea, developed the algorithm, conduct the simulation and experiments and prepared the manuscript drafts.

Chapter 5 is related to conference paper: Z. Liu, G. Lin, S. Yang, F. Liu, W. Lin, and W. L. Goh, “Towards Robust Curve Text Detection with Conditional Spatial Expansion,” The IEEE Conference on Computer Vision and Pattern Recognition, pp. 7269–7278, 2019

The contributions of the co-authors are as follows:

- Prof. Goh Wang Ling and Prof. Lin Guosheng supervised me on the overall research direction and edited the manuscript drafts.
- I proposed the idea, developed the algorithm, conduct the experiments and prepared the manuscript drafts.
- Dr. Yang Sheng, Dr. Liu Fayao and Prof. Lin Weise helped me on the experiments and gave comments on the manuscript.

2019/Aug/19

.....
Date

Zichuan Liu

.....
Liu Zichuan

Abstract

Optical Character Recognition (OCR) is a common method to convert typed, hand-written or printed text from documents and scene photos into digital format, which can be edited, searched, stored and displayed. It acts as a uniform information entry for text related higher-level applications, such as cognitive computing, machine translation text-to-speech, and data mining.

OCR is usually divided into two sub-tasks, text detection and recognition. In this thesis, we focus on detecting text in photos captured from natural scenes, which is challenging and actively draws attention from machine learning and computer vision society. Our research started by investigating the deficiency of existing deep learning-based scene text detection methods. We found that the performance of nowadays Convolutional Neural Network (CNN) based scene text detection systems is mainly restricted by the limited receptive field of CNNs, the geometrical mismatching between text objects and predefined references, and the rigid representation of a text object. The limited receptive field prevents a CNN from fully perceiving texts with large aspect ratios and varying scales. The geometrical mismatching leads to under-trained or over-trained classifiers and bounding box regressors in a classic anchor-based detection system. The rigid representation reduces the generality of a text detection system to retrieve text instances with various shapes. In this thesis, based on the systematical analysis of problems mentioned above, we proposed two new methods to detect scene texts with arbitrary shapes and orientations.

Instead of using a prediction from a single location with limited receptive field to

represent the text’s objectness, we proposed a new detection framework called Markov Clustering Networks (MCNs) that represents an object with predictions at multiple locations and can extract text objects by Markov Clustering. Different from traditional top-down text detection methods, our MCN can be treated as a bottom-up method. It iteratively aggregates local predictions into a global object representation, which is robust to texts with large aspect ratios and highly varying orientations. Moreover, this reference-less framework does not suffer from performance degradation caused by mismatching, which enjoys much better generality. We further enhance the capability of MCN in capturing long texts by introducing a 2-Dimensional Position-aware Attention mechanism.

In addition to detection rectangle text objects, we step further to design a method to retrieve curve shape text objects. The main challenge of curve text detection comes from irregular shapes and highly varying orientations. On the one hand, the bounding box representation of a text object does not scale well in the curve scenario, which fails to tightly represent the boundaries of a curve text and results in low recall and precision. On the other hand, irregular shapes and complex layout introduce additional noises in extracted representations by region-based methods, reducing the performance of polygon regression and instance segmentation. To address these problems, we propose a novel Conditional Spatial Expansion (CSE) mechanism to detect text with arbitrary shapes. We explicitly model a curve text object as a 2-dimensional sequence and regard retrieving instance-level curve text region as a conditional prediction problem on the spatial domain. Starting with an arbitrary interior point of a text region, CSE progressively predicts the status of its neighborhoods based on which the related sub-regions are merged into the entire instance region. Our CSE is extremely discriminative, especially when texts are close to each other and provides a controllable approach to extract an expected text region with minimum efforts of post-processing.

Acknowledgment

First of all, I would like to express my gratitude to my previous Ph.D. advisor Professor Hao Yu for accepting me to pursue Ph.D. degree in NTU.

I would also like to thank my current advisor Wang Ling Goh for enthusiastic encouragement, continuous support. Her guidance helped me in both academic research, personal growth and future career outlook.

I would also like to thank my co-advisor, Professor Guosheng Lin, for their constructive supervision, insightful guidance, valuable research resources and opportunities provided during the Ph.D. study.

Besides my advisors, I would like to express my gratitude towards all the members in the VIRTUS Lab and MICL Lab, for the enthusiastic research atmosphere, stimulating discussions, the sleepless nights before deadlines, and the fun we had in the last four years. My special gratitude also goes to the lab administrators of the VIRTUS Lab, for their continuous support and help during the Ph.D. study.

I would like to thank all my friends I met at NTU, especially Yang Sheng, Zhang Sheng, etc. Thank you for helping me when I had a problem and thank you for making my Ph.D. career funny and interesting.

I would especially thank my girlfriend, Dr. Ran Zhang, who has shared all my sadness, happiness, failure and success in everyday of my Ph.D. My life would be completely different without her. Last but most importantly, I thank my family. Without the love from you, I would have nothing.

Contents

Abstract	v
Acknowledgment	vii
List of Figures	xiii
List of Tables	xix
Abbreviations	1
1 Introduction	3
1.1 Background, Motivation and Challenges	3
1.2 Objective and Scope	6
1.3 Contributions	7
1.4 Thesis Organization	9
2 Literature Review	11
2.1 Methods before the rise of Deep Learning	11
2.2 Methods after the rise of Deep Learning	13
2.2.1 Detection	15
2.2.2 Recognition	25
2.2.3 End-to-End Systems	28
2.2.4 Benchmarks and Evaluation Protocols	29

3	Bottom-up Scene Text Detection: Markov Clustering Networks	35
3.1	Introduction	35
3.2	Markov Clustering Networks	38
3.2.1	Overview	38
3.2.2	Object Representation by Stochastic Flow	39
3.2.3	Detecting Object by Markov Clustering	41
3.2.4	Learning Clustering with Flow Labels	42
3.3	Detail Implementation of MCN for Scene Text Detection	46
3.3.1	Neural Network Architecture	46
3.3.2	Bounding Box Generation	48
3.3.3	From Image to Stochastic Flow	48
3.4	Experiments	51
3.4.1	Datasets	51
3.4.2	Experiment Details	52
3.4.3	Ablation Studies	53
3.4.4	Performance Comparison	55
3.4.5	Speed	58
3.5	Conclusion	59
4	Enhancing Markov Clustering Networks: Two-dimensional Position-aware Attention	61
4.1	Introduction	61
4.2	Two-dimensional Position-aware Attention	63
4.2.1	2D Self-attention with Position Encoding	63
4.2.2	Efficient Forward and Backward Pass	64
4.3	Detail Implementation	66
4.3.1	Simplified Network Architecture	66
4.3.2	Training	68

4.3.3	Testing	71
4.4	Experiments	71
4.4.1	Detail Analysis	71
4.4.2	Comparison with Existing Methods	75
4.5	Conclusion	77
5	Curve Text Detection: Conditional Spatial Expansion	79
5.1	Introduction	79
5.2	Method	81
5.2.1	Overview	81
5.2.2	Modeling	83
5.2.3	Conditional Spatial Expansion	84
5.2.4	Seeding	88
5.2.5	Optimization	88
5.3	Implementation Details	90
5.3.1	System Architecture	90
5.3.2	Training	91
5.4	Experiment	93
5.4.1	Datasets	93
5.4.2	Experiment Details	94
5.4.3	Flexibility and Robustness	95
5.4.4	Comparing with Mask RCNN	96
5.4.5	Comparing with Polygon Regression	99
5.4.6	Peer Comparison	101
5.5	Performance on Non-curve Text Benchmarks	103
5.6	Conclusion	103
6	Conclusion and Future Works	105
6.1	Conclusion	105

6.2 Future Works	107
A Publication List	109
A.1 Conference	109
A.2 Journal	109
Bibliography	111

List of Figures

1.1	(a) Scene texts in different languages, fonts, color, sizes, and orientations; (b) Text objects have highly varying aspect ratio, shapes and layouts; (c) Complex background introduces ambiguity and causes false detection; (d) Imperfect imaging condition leads to error detection.	6
2.1	Processing flow for scene text detection before deep learning era.	12
2.2	Deep Learning based Scene Text Detection and Recognition.	14
2.3	The evolution of deep learning based scene text detection and recognition: Deep Learning based Scene Text Detection and Recognition: (a) [1] and (b) [2] are representative multi-step methods; (c) [3] and (d) [4, 5] are simplified systems. The detection branch and recognition in (c) are trained separately, while systems proposed in [4, 5] contain detection and recognition module which are trained jointly.	16
2.4	High-level illustration of existing anchor-based and region proposal methods: (a) Single-level anchor-based methods [6]; (b) Multi-level anchor-based methods [7, 8]; (c) Region proposal methods [9, 10] . . .	17
2.5	Network Architecture of EAST [3].	19
2.6	(a) Processing flows of R ² CNN [9] and (b) processing flow of R ² RPN [11].	20

2.7	(a) SegLink [12] retrieves text objects by detecting text segments and combining them together; (b) [13] extracts text instance by detecting four corners of a text bounding box; (c) [14] detects the boundary, inner area, local text radius and orientation separately, and combine them into an instance-level region by post-processing.	22
2.8	TextSnake [14]	25
2.9	Scene Text Recognition Framework: (a) basic idea; (b) RNN stacked with CNN; (c) FCN based sequence prediction; (d) Attention-based models; (e) Character-guilt attention; (f) Sequence reordering	26
2.10	Example images in ICDAR 2013, ICDAR 2015, ICDAR 2017, ICDAR 2019, CTW1500, Total Text and MSRA-TD500.	31
3.1	All the detectors are trained with the same data. The ground-truth boxes are labeled in red and the detected boxes are labeled in yellow. (a) Bounding boxes proposed by Faster R-CNN; (b) Bounding boxes proposed by Faster R-CNN with deformable convolution; (c) Bounding boxes proposed by MCN. Rectangles in red are ground-truth boxes and those in yellow are predictions.	37
3.2	(a) Input image; (b) Predicted stochastic flow graph by MCN: Nodes correspond to the equidistant overlapping regions in the image and the connections between nodes are referred as flows. The flow intensity is visualized as the width of an edge. A strong flow results in a wide edge while weak flow leads to a narrow edge; (c) Extracted clusters from stochastic flow graph presented in (b) by Markov Clustering; (d) Bounding boxes generated from the clustered nodes in (c).	38
3.3	Markov Clustering Network.	40

3.4	(a) Computing the attractor P_{attr} given a ground-truth bounding box; (b) Adjusting a ground-truth bounding box to include the attractor; (c) Assigning an attractor index for each node; (d) Generating 3D cluster label for each node based on (c).	44
3.5	Computing graph of Markov Clustering. It takes initial flow matrix M_0 as input and outputs a converged flow M_N^P to compute $C_f(M_0)$	45
3.6	Implementation detail of MCN for scene text detection.	47
3.7	(a) Architecture of Fore-/Background Network (FBN); (b) Feature Pyramid Network (FPN) fusing feature maps with different resolutions; (c) 2-dimensional Recurrent Neural Network (2D-RNN) encoding contextual representations; (d) Regional objectness classifier predicting presence of an object with stride of 16×16 pixels.	49
3.8	Local Correlation Network (LCN) with four convolution blocks and a softmax layer outputs correlation measurements S_1, S_2 and S_3 between current anchor and its three neighbors (bottom, right and left).	50
3.9	(a) Prediction results by Faster R-CNN; (b) Prediction results by MCN.	55
3.10	Flow profiling of detecting text with different orientations. (a) Input images; (b) Predicted objectness maps P , link maps S_1, S_2, S_3 , and stochastic flows f_0, f_1, f_2, f_3 ; (c) Visualization of Markov Clustering with dominative flows marked in yellow arrows and attractors marked in blue points; (d) Generated bounding boxes (in yellow) based on predicted clusters and according ground-truths labeled in red boxes.	56
3.11	Example results on ICDAR 2013 and MSRA-TD 500 datasets. Text instances with multiple scales and orientations are detected. The predicted bounding boxes by our method are labeled in yellow and the ground-truth bounding boxes are labeled in red.	56
4.1	2-Dimensional Position-aware Attention.	65

4.2	Architecture of Feature Pyramid Network applied in MCN model. . . .	67
4.3	Architecture of 2D-RNN used for comparison with 2D-PAA.	69
4.4	The first/second row is the performance on ICDAR 2013/ICDAR 2015 with different γ to train 2D-PAA.	74
4.5	Memory usage and computation time of 2D-PAA. First and second rows correspond to forward and backward computation respectively.	74
4.6	Example results on ICDAR 2013, ICDAR 2015 and MSRA-TD 500 datasets. Text instances with multiple scales and orientations are detected. The predicted bounding boxes by our method are labeled in yellow and the ground-truth bounding boxes are labeled in red.	78
5.1	Problems of existing curve text detection methods: Two region proposals shown in (a) and (d) inevitably include unexpected texts since they are closed to each other. Thus, it causes failures for Mask RCNN based methods and polygon regression methods demonstrated in (b) and (e). Our CSE method demonstrated in (c) and (f) shows strong robustness to this situation and brings significant performance gain.	82
5.2	Our CSE inside-out explores the expanding indicator of every node and merges nodes with merging direction pointing to nodes that are already in the object region.	84

5.3	Given a seed located, depicted as a blue cross within the red region in (a), at the interior of a text region, we expand a grid with $S \times S$ points and sample the feature produced by backbone at these locations using bi-linear interpolation, shown in (a). As demonstrated in (b), the CSE computation starts with the seed and spreads to the adjacent feature nodes. The spreading directions are indicated by the arrows in (b). Each node takes the outputs and the hidden states of previous nodes as input and produces new hidden state and expanding indicator predictions, as illustrated in (c). Finally, the nodes associated with the same seed, depicted as the yellow region in (b), are extracted according to the expanding indicators.	85
5.4	Computing Graph inside a Node.	85
5.5	Ground-truth Labeling: All arrows represent the candidate merging directions for nodes, and the corresponding scores predicted by CSE are marked. The candidate directions with highest scores are label as positive directions, which are represented by solid arrows.	89
5.6	System Architecture.	92
5.7	Robustness and Flexibility Analysis: In the first row of each case, the sampling regions are represented by the bounding boxes in yellow and the seed locations are labeled by the cross dash lines. The second row shows the zoom-in of RoIs. The corresponding heatmaps of associate regions are shown in the third row. As shown in (a), our method is very flexible in the seed's locations. (b) and (c) prove the robustness of our method to extract text from a density text region.	95
5.8	Performance v.s. rescaling factor δ_s on Total-Text (a) and CTW-1500 (b); Performance v.s. relocation factor δ_c on Total-Text (c) and CTW-1500 (d).	97

5.9	(a) Examples output by baseline method; (b) Examples output by our CSE method. Compared with the baseline Mask RCNN method, our method shows impressing robustness to the ambiguity caused by adjacent texts and impressing selectivity to the targeted instance.	98
5.10	(a) The images with RoIs; (b) The text boundaries output by baseline method; (c) The text boundaries produced by our CSE method. The baseline method is easily affected by the unexpected texts included in the same boxes, while our method shows strong robustness to this situation.	100
5.11	Detection results on Total-Text and CTW-1500.	102

List of Tables

2.1	Existing datasets and benchmarks for scene text detection and recognition.	30
3.1	Comparison between MCN and baseline model (Local-links) on public benchmarks.	54
3.2	Performance comparison between MCN and Faster R-CNN in detecting scene texts. †: VGG-16 + FPN, ‡: ResNet-50 + FPN	54
3.3	Localization performance on ICDAR-13, ICDAR-15 and MSRA-TD500.	57
3.4	Number of iterations, detection performance, and runtime.	58
3.5	FPS comparison on ICDAR-13 with input size of 512×512	59
4.1	Comparison between baseline MCN and MCN-2DPAA on ICDAR 2013 and MSRA-TD500.	72
4.2	Performance and inference speed with different prediction density. . . .	75
4.3	Localization performance on ICDAR 2013.	76
4.4	Localization performance on ICDAR 2015.	76
4.5	Localization performance on MSRA-TD500.	77
5.1	Performance Comparison between Mask RCNN based method and our CSE method.	98
5.2	Performance Comparison between Polygon Regression based method and our CSE method.	99
5.3	Detection Performance on Total-Text.	101

5.4	Detection Performance on CTW-1500.	101
5.5	Localization performance on ICDAR 2013.	102
5.6	Localization performance on ICDAR 2015.	103
5.7	Localization performance on MSRA-TD500.	103

List of Abbreviations

[ORC]Optical Character Recognition

CNN Convolutional Neural Network

MCN Markov Clustering Network

MC Markov Clustering

2D-PAA 2-dimensional Position-aware Attention

CSE Conditional Spatial Expansion

RoI Region of Interest

CCA Connected Component Analysis

SW Sliding Window

MSER Maximally Stable Extremal Regions

SWT Stroke Width Transform

ER Extreme Regions

CRF Conditional Random Field

TCL Text Central Lines

FCN Feature Prymaid Network

NMS Non-maximum Suppression

RCNN Regional Convolutional Neural Network

RPN Region Proposal Network

RNN Recurrent Neural Network

FCN Fully Convolutional Network

CTC Connectionist Temporal Classification

FAN Focus Attention Network

EP Edit Probability

TP True Positives

FN False Negatives

FP False Positives

FBN Fore/background Network

LCN Local Correlation Network

SGF Stochastic Flow Graph

PCA Principle Component Analysis

Chapter 1

Introduction

In this chapter, the background, motivations, and challenges of scene text detection and recognition will first be introduced in Section 1.1. In Section 1.2, the overview of our objective and research scope is presented. The contribution of our studies will be summarized in Section 1.3 and the organization of this thesis will be provided in Section 1.4.

1.1 Background, Motivation and Challenges

Text is among the most brilliant and influential creation of humankind. As a written form of language, it plays a critical role in spreading knowledge and culture of human beings during lengthy human history and constitutes a cornerstone of human civilization. Text is ubiquitous, where one can discover tremendous value using advanced data mining technology. By analyzing the patient's medical history, the risk of future illness can be inferred, and early prevention and treatment can be made. Road signs and street signs can provide rough location information for navigation. Collecting and analyzing signboards can quickly construct sketches of businesses and public facilities in a city. Therefore, powerful methods and systems are demanded to convert the texts from the real world to the digital world, where they can spread more widely, be stored more efficiently and be

analyzed on a much larger scale.

Optical Character Recognition (OCR) is an electronic conversion of an image with typed, printed or handwritten texts into the machine-encoded format, whether from scanned documents or a scene photo. OCR is widely applied to collect information from printed paper data records – whether passport documents, invoices, bank statements, digital receipts, business cards, mail, printouts of static-data, or any suitable documentation. It localizes and digitizes texts which can be electronically edited, searched, stored, and displayed. OCR also acts as an assistant in modern vision-based robotics automation systems to capture texts in the wild, such as auto-driven mobile, drone-based warehouse management. For example, autonomous vehicles detect and recognize the signage contents for localization, routing, and speed control. The drones in a warehouse also require OCR capability to collect packages' information from their tags, which helps to manage goods efficiently. OCR yields excellent result in processing documents with neat contents, fixed layouts, clean background and good illumination. There has been much open-source or commercial software available, which detects and recognizes texts based on traditional machine learning techniques and handcrafted features. However, the success of OCR in documents does not continue in natural scene scenarios. Texts from natural scenes usually have various fonts, shapes, and orientations. They may be captured under low resolution and uneven illumination and accompanied with complex backgrounds. Therefore, detecting and recognizing scene text is much more challenging than interpreting clean and well-formatted documents. Due to the demands of a growing number of real-world applications and potentially substantial commercial value, reading texts in the wild has come to public consciousness and received increasing attention from industry and academic society in recent years.

However, despite years of research, there still exist several obstinate challenges when detecting and recognizing texts from natural scenes. Texts in the wild exhibit much higher diversity and variability than those in documents. As shown in Fig. 1.1 (a), scene text instances can be in different languages, fonts, colors, sizes, and orientations. Moreover,

text objects have highly varying aspect ratios, shapes, and layouts, Fig. 1.1 (b). All of these variations pose challenges and lead to the detection and recognition framework and paradigms specified for scene texts. The complex background is considered as another challenge for scene text detection and recognition. The backgrounds coupled with ground-true texts objects are unpredictable and consist of patterns similar to text. Fig. 1.1 (c) demonstrates the error detection results caused by text-like patterns. Edges of buildings, traffic signs, windows and bricks, and occlusion caused by foreign objects potentially result in confusion and mistakes and thus reduce the precision of the system. Finally, as depicted in Fig. 1.1 (d), the poor imaging condition further degrades the performance of scene text detection and recognition. The uncontrollable illumination corrupts the image with unrecoverable shadows and highlights. The inappropriate shooting distance or angles reduces the image resolution. Out-of-focus and shaking during capturing blurs the image. The difficulties mentioned runs through the years before the boom of deep learning techniques.

Deep learning has become the mainstream technique for computer vision as well as other applications since AlexNet [15] won ILSVRC2012 [16] competition. It provides a powerful capability of automatic feature learning from large amounts of data and thus frees researchers and engineers from repeatedly designing and testing handcrafted features in different scenarios. The end-to-end characteristics¹ also simplifies the pipeline and achieves much higher performance than previous handcrafted feature-based methods on standard benchmarks.

The performance of scene text detection and recognition is remarkably improved by the recent advances in neural network-based object detection, segmentation, and sequence recognition. Even though significant progress has been made, scene text detection and recognition remain to be an unsolved problem. In scene text detection, the most popular bounding box based method, e.g. Faster-RCNN [17], YOLO [18], SSD [19] and Mask

¹The end-to-end characteristics means that the neural network outputs the final results of a task and can be directly optimized by backpropagation.

Figure 1.1: (a) Scene texts in different languages, fonts, color, sizes, and orientations; (b) Text objects have highly varying aspect ratio, shapes and layouts; (c) Complex background introduces ambiguity and causes false detection; (d) Imperfect imaging condition leads to error detection.

RCNN, are still vulnerable to text with high variation in aspect ratio, size, shapes and orientations due to their limited receptive field of Convolutional Neural Networks (CNNs), heuristic anchor box mechanism and inappropriate geometrical representation of text objects. A long text line could be split into multiple predictions when a CNN does not fully perceive it. An anchor-based text detector could be under-trained due to a low matching rate caused by the high variance of text geometry [20, 7, 7, 12]. In region-based methods [9, 10, 21, 11, 22], a CNN may not be able to distinguish close texts from each other and produces incorrect boxes or polygons containing multiple text objects. These false detections significantly affect subsequent text recognition by introducing incomplete or overmuch text sequences.

1.2 Objective and Scope

The Optical Character Recognition can be divided into two sub-tasks, text detection and recognition. In this thesis, we focus on detecting texts from natural scenes, which

constitutes the first stage of an OCR system and critically affects the performance of subsequent scene text recognition. The objective of our research is to find out the cause of performance decline in neural network-based scene text detection system and to design effective algorithms for robust scene text detection. In particular, we explore the following problems in this thesis:

- To alleviate or even eliminate the negative effects of limited receptive fields on performance;
- To enhance the flexibility to text objects with various sizes, aspect ratios, and orientations;
- To improve the discriminability to close text objects with different shapes and layouts.

By resolving the problem of limited receptive fields, the performance can be greatly improved in detecting long text with extremely large aspect ratio. Traditional text detectors do not scale well in detecting texts with highly varying geometry and layouts and usually provide incomplete and damaged text instances in this case. Therefore, the flexibility and discriminability of a text detector is critical to the performance of the downstream text recognizer.

1.3 Contributions

Firstly, based on the analysis of the limitations of existing deep learning-based scene text detection methods, a novel framework named Markov Clustering Network (MCN) is proposed to retrieve scene texts with a high variation of aspect ratio and scale. MCN predicts instance-level bounding boxes by converting an image into a *stochastic flow graph* where the Markov Clustering (MC) is performed. MCN can detect text objects with an arbitrary size, aspect ratio, and orientation without prior geometrical knowledge of the objects. The *stochastic flow graph* encodes objects' local correlation and semantic

information. An object is modeled as strongly connected nodes, which allows flexible and bottom-up detection for scale-varying and rotated objects. There is no anchor system consisted of the workflow, and thus one is released from repetitive reference box design for various geometric statistics in different scenarios. Also, under-trained classifiers are eliminated. The evaluation of public benchmarks shows that our method outperforms the existing methods by a large margin in detecting multi-oriented text objects.

Secondly, to enhance the representation capacity of a CNN model to capture long text objects and reduce the computation complexity of MCN in both training and testing phase, 2D Position-aware Attention (2D-PAA) mechanism is introduced. The 2D-PAA considers input representations as a 2D sequence. Instead of capturing relative and absolute position information along the spatial dimension directly through the sequential structure of RNN [23], 2D-PAA applies position encoding to expose positional information to the model. Essentially, it explicitly models the correlation between two spatial representations and thereby extends the receptive field of the CNN model by selectively fusing spatial representations in different parts of an image. Compared with the spatial recurrent architecture [24], 2D-PAA is much easier to parallelize with plenty of contextual information. The evaluation of public benchmarks validates the effectiveness of the proposed 2D-PAA mechanism.

Thirdly, a Conditional Spatial Expansion (CSE) mechanism is proposed to detect text with arbitrary shape. This research starts with investigating the deficiency of existing scene text detection methods for arbitrary texts. Based on the analysis of the mainstream methods (instance segmentation based and polygon regression-based methods), we conclude that both methods are easily corrupted by the unexpected texts which are close to each other and included in the same region proposal. Multiple close text objects with arbitrary shape and layout corrupt the representations extracted by a Region of Interest Pooling (RoI Pooling) [25]. The corrupted representations make segmentation-based methods fail to discriminate a text object from the others crowded in the same region proposal. The polygon regression-based methods can neither produce a correct

boundary from the ambiguous representations. The proposed CSE mechanism is capable to discriminatively retrieve instance-level text regions from crowded texts with complex shapes and layouts. Text detection is explicitly modeled as a spatial condition prediction problem. CSE initializes a seed at an arbitrary location within a text and progressively merges neighborhood sub-regions to form the whole text region. The experiments show that CSE is more discriminative than the previous methods in detecting irregularly shaped texts from dense areas.

1.4 Thesis Organization

The remaining parts of this thesis are organized as follows. Chapter 2 systematically reviews the related works in the literature. Chapter 3 presents the Markov Clustering Network (MCN) for scale varying text detection. Chapter 4 presents the 2-dimensional Position-aware Attention module to enhance the performance of our Markov Clustering Network further. Chapter 5 presents the Conditional Spatial Expansion (CSE) mechanism for arbitrarily shaped text detection. The thesis is concluded in Chapter 6 and the potential future works are discussed.

Chapter 2

Literature Review

In this chapter, we present an overview of recent developments in scene text detection and recognition. Since our research focuses on the deep learning-based scene text detection methods, we will generally review methods in both scene text detection and recognition before and after the rise of deep learning techniques, and highlight the progress in deep learning-based scene text detection. In Section 2.1, we present a brief review of methods for scene text detection and recognition proposed before the deep learning era. The deep learning enhanced methods will be illustrated in Section 2.2. In Section 2.2, other than comprehensively introducing classical text recognition methods, we will spend more on the pipeline evolution of the text detection systems and introduce different text detection systems categorized by prediction granularity and specific targets. In addition, we will review all the end-to-end systems which integrate functionalities of both detection and recognition in Section 2.2.3. The public benchmarks and evaluation protocols are introduced in Section 2.2.4.

2.1 Methods before the rise of Deep Learning

In this section, we will briefly go through the algorithms for scene text detection and recognition proposed before the deep learning era. In this period, the algorithm

Figure 2.1: Processing flow for scene text detection before deep learning era.

development for detection and recognition is focusing on feature design.

Most of the text detection methods either adopt *Connected Component Analysis (CCA)* [26, 27, 28, 28, 28, 29, 30, 30] or *Sliding Window (SW)* based classification [31, 32, 33]. The *Connected Component Analysis (CCA)* based methods, depicted in Fig. 2.1, generally perform text/non-text classification on extracted candidate components (through e.g. color clustering or extreme region extraction) using heuristic rules or classifiers trained on hand-crafted features. Two candidate proposal methods are worthy to mention, *Maximally Stable Extremal Regions (MSER)* [34] and *Stroke Width Transform (SWT)* [27]. The MSER based methods consider characters as Extreme Regions (ER) and the text localization is interpreted as finding all sequences of ERs with local maximum probability. The SWT is based on the assumption that texts tends to maintain fixed stroke width. It converts a gray-level image to an array containing likely stroke widths for each pixel which will be grouped into letter candidates if they have similar width. In *Sliding Window (SW)* classification, windows of various sizes slide over the image, where a binary classification is performed to extract positive text segments. These segments will be further grouped into text regions using morphological operations [32], Conditional Random Field [31] and other graph based methods.

For the text recognition, Shi *et al.* [35] and Yao *et al.* [36] proposed character segments based recognition algorithm, where additional information, such as language model and spatial constraints, are utilized to formulate a cost function and the word recognition is realized by minimizing this cost function. Rodriguez *et al.* [37, 37], Gordo *et al.* [38] and Almazan *et al.* [39] utilized label embedding to directly perform matching between strings and images. Michal *et al.* [40] and Phan *et al.* [41] proposed to use strokes and character key-points as features for text recognition, respectively. Another divided the recognition process into a series of sub-problems, including text binarization [42, 42, 43, 44], text line segmentation [45], character segmentation [46, 47, 48], single character recognition [49, 50] and word correction [51, 52, 53, 54, 55].

In addition to independent text detection and recognition, there have been efforts devoted to integrated or end-to-end systems. In Wang *et al.* [33], texts are considered as special objects which can be detect by nearest neighbor classifier trained on HOG features [56] and then grouped into words by Pictorial Structure (PS) based model [57]. Neumann *et al.* [58] proposed a decision delay approach which first detects characters using extremal regions and decodes transcripts from detected character through a dynamic programming algorithm.

In summary, text detection and recognition methods before the deep learning era are mainly focusing on extracting low-level or mid-level hand-crafted features. They require complex and repetitive pre- and post-processing to achieve good performance and vulnerable to diversity environment.

2.2 Methods after the rise of Deep Learning

In this section, we will introduce the recent advances of scene text detection and recognition methods based on deep learning. In the deep learning era, most of the methods utilize deep learning-based models and most researchers are approaching the problem from a diversity of perspectives. The methods powered by deep learning enjoy

Figure 2.2: Deep Learning based Scene Text Detection and Recognition.

the characteristics of automatic feature learning, which release us from repetitive feature design for various applications and scenarios. Meanwhile, the huge representation capacity of deep learning allows researchers to consider the problem from different perspectives and approach it in different ways. Deep learning also enriches and promotes diversity of the scene text detection and recognition system aiming at different targets, e.g. faster and simpler pipeline [3], text for various shaps [12], and synthetic data [59]. The progresses and research trends of deep learning based scene text detection and recognition methods are outlined in Fig. 2.2. We follow this diagram to go through the recent changes and progresses on scene text detection and recognition research.

2.2.1 Detection

We review the scene text detection methods based on deep neural networks from three perspective, the pipeline evolution, prediction granularity and different targets.

Pipeline Evolution

The pipeline evolution of deep learning based scene text detection system is illustrated in Fig. 2.3. Deep neural networks leads a trend of simplified pipeline. The early methods in deep learning era is quite similar to those before the era of deep learning. They cast the task of text detection into a multi-step process [1, 2]. More recent methods adopted simplified and much shorter pipelines, which reduces the training complexity and prevent error propagation among different processing stages [3, 4]. The workflow of recently proposed simplified methods also go through a change from separately-trained-two-stage to jointly-trained, and the latter has surpassed the former.

In the multi-step methods, a *Convolutional Neural Network (CNN)* is applied to an input image to prediction different components of a text, which will be grouped into a text object. In [2], a CNN is used to classified whether a pixel (1) belong a character, (2) is inside a text region, and (3) the text orientation around the pixel. The connected positive responses are considered as a detected characters and text regions. For the characters belonging to the same text region but not belonging to the same text instance, Delaunay triangulation [60] is adopted to partition the characters in to text lines based on orientation attributes. Similarly, [61] applies Fully Convolutional Neural Networks (FCN) to detect text coarse regions from which text blocks are generated. The characters are detected from the text blocks using MSER [34], and grouped into text lines according to a set of geometric constrains. He *et al.* [62] proposed a multi-stage method for instance-level text line detection. They applied an FCN to predict both text blocks and Text Central Lines (TCLs). A TCL is defined as a shrunk version of the original text line region. Afterwards, an instance-aware semantic segmentation is performed on the concatenation of prediction

Figure 2.3: The evolution of deep learning based scene text detection and recognition: Deep Learning based Scene Text Detection and Recognition: (a) [1] and (b) [2] are representative multi-step methods; (c) [3] and (d) [4, 5] are simplified systems. The detection branch and recognition in (c) are trained separately, while systems proposed in [4, 5] contain detection and recognition module which are trained jointly.

Figure 2.4: High-level illustration of existing anchor-based and region proposal methods: (a) Single-level anchor-based methods [6]; (b) Multi-level anchor-based methods [7, 8]; (c) Region proposal methods [9, 10]

binary TLC maps and original images to extract the instance-level text line regions. Note that the mentioned methods contain the undifferentiable operations in their workflows. The training algorithms are not designed to directly optimize the final results. Therefore, the performance could be sub-optimal when compared with the fully end-to-end systems [4, 5, 63].

Recently, most of the methods follow a two-step paradigm, consisting an end-to-end trainable neural network model and a post-processing stage which is much simpler than those in previous multi-stage methods [20, 9, 7, 6, 12, 21, 11, 10, 22]. These methods are mostly inspired by the techniques in general object detection [64, 25, 65, 66, 19, 17] and benefit from the highly integrated neural network modules with instance level prediction capability. Based on the detection paradigm, they can be categorized into two branches: (1) *Anchor-based* methods [20, 7, 7, 12] and (2) *Region proposal* methods [9, 10, 21, 11, 22].

Both anchor-based methods and region proposal methods share the same CNN feature extractor which provides spatial domain representation over an image as shown at the

bottom of Fig. 2.4. The anchor-based methods, mainly inspired by YOLO [18] (Fig. 2.4 (a)) and SSD [64] (Fig. 2.4 (b)), directly perform text/non-text classification and regression on the extracted features and produce final text location after Non-maximum Suppression (NMS). Compared with the general object detection, oriented anchor boxes and polygons are introduced in text detection due to the diversity of text geometry. Liu *et al.* [6] and Liao *et al.* [7] (TextBoxes) adopted YOLO and SSD network respectively to fit the varying orientations and aspect ratios of text lines. In addition to multi-oriented anchor mechanism, Liu *et al.* [6] propose a quadrilaterals anchor mechanism which provides a tighter representation of a text instance. This method is subsequently adopted by Liao *et al.* [8] in their enhanced version of TextBoxes. Another variant of anchor-based methods is EAST [3]. Different from standard SSD networks detecting text on multiple feature maps with different receptive fields, EAST, illustrated in Fig. 2.5, integrates all the feature maps with different resolution by gradually upsampling and performs classification and regression (including box regression and quadrilateral regression) densely on a feature map with size of 1/4 of original input image. EAST enjoys a simple workflow and an efficient network architecture, which has become one of the most popular methods for scene text detection.

Region proposal methods usually inherit from the standard R-CNN [25, 65, 17] framework. In this framework, an region proposal module implemented by *Selective Search* [25] or *Region Proposal Networks (RPNs)* is applied to produce a set of region proposals (represented as bounding boxes) that could contain text objects. A CNN then classifies each proposed region into text/non-text and calibrate the location and size of the proposed bounding boxes. The final predicted boxes are produced by running NMS over all the calibrated boxes with a given Intersection over Union (IoU) threshold. The representative method is the *Rotation Regional Convolutional Neural Networks (R^2CNN)* [9] and *Rotation Region Proposal Networks (R^2RPN)* [11]. As shown in Fig. 2.6 (a), it follows and adapts the standard Faster R-CNN [17] framework using RoI pooling to extract features from proposed regions for classification and box regression.

Figure 2.5: Network Architecture of EAST [3].

But R^2 CNN additionally performs multiple RoI pooling with different sizes and then predicts inclined box coordinates to fit the texts with arbitrary orientations. The R^2 RPN [11], shown in Fig. 2.6 (b), proposed a *rotation region proposal network* that produces coarse arbitrarily oriented box proposals. Moreover, Rotation Region-of-Interest (RRoI) pooling is proposed to project arbitrary-oriented proposals to a feature for text/non-text classification and box regression.

Prediction Granularity

Different from general object, texts are homogeneous as a whole and show locality. Notably, the homogeneity and locality refer to the property that any part of a text instance

Figure 2.6: (a) Processing flows of R²CNN [9] and (b) processing flow of R²RPN [11].

is still a text. Thus, a text instance can be decomposed into multiple components that can be detected separately. This idea results in a set of new methods that only predict sub-texts and then assemble them into a text instance. We will revisit the text detection methods from the perspective of granularity, including instance level and sub-text level, and highlight the new methods based on sub-text prediction.

The *instance-level* method [20, 7, 7, 12, 9, 10, 21, 11, 22] follow the standard object detection paradigms, where a region proposal network and refinement network are cascaded to make a prediction. The region proposal networks coarsely localize possible text instances, and then a refinement network, e.g. RCNN [17, 65], is applied to classify the proposed regions into text/non-text as well as calibrate the locations. These methods work by capturing the global features of an object. In contrast, sub-text level methods [13, 67, 62, 68, 2, 69, 12, 70, 71, 61, 72] appreciate local features and relation between text components. The sub-text level detection methods only predict parts of a text and combine them into a text instance according to predicted cues. As illustrated in Fig. 2.7, the sub-text based methods can be categorized into two types, which are *component-level* methods and *pixel-level* methods.

The component-level methods [13, 12, 70, 73, 72] usually predict at a medium granularity. Base on the observation that components of the same text can appear in clusters, the component-level methods are designed to separate text instance from each other. The earliest component-level method is *Connectionist Text Proposal Network (CTPN)* [70, 73, 72], which inherits the anchor mechanism and the *Recurrent Neural Network (RNN)*. These methods consist of a CNN followed by an RNN which performs sequential binary classification on spatial domain. The CTPNs only predict a segment of a text horizontally and combine these segments into text lines according to predefined geometric thresholds (e.g. horizontal distance and vertical offset). Since the RNN can capture the correlation between adjacent predictions, the CTPN methods are robust to long text lines. However, they are limited to detecting horizontal texts since the RNN can only encode horizontal contextual information in their configuration. Moreover, heuristic

Figure 2.7: (a) SegLink [12] retrieves text objects by detecting text segments and combining them together; (b) [13] extracts text instance by detecting four corners of a text bounding box; (c) [14] detects the boundary, inner area, local text radius and orientation separately, and combine them into an instance-level region by post-processing.

combining strategy may not be adaptable to various scenarios with different font sizes. SegLink [12], shown in Fig 2.7 (a), steps further and learns to determine whether two adjacent segments belong to the same text by explicitly predicting link scores to each segment or pixel. A text instance is extracted by Connected Component Analysis (CCA) with cues provided by link scores. Compared with the CTPN methods, the SegLink is end-to-end trainable and enjoys a better adaptability. However, error link classifications still occur where text instance are close. Another representative component-level method [13], illustrated in Fig. 2.7 (b), predicts four corners of a text instance based on which a instance-level bounding box is constructed by sampling and grouping the predicted corner points.

The pixel-level methods [67, 62, 68, 2, 61] adopt Fully Convolutional Networks (FCNs) to predict a set of dense confidence maps that reveal the objectness and the local geometric information of text instance at current pixel location. A post-processing flow will be included in these methods to assemble pixels into an entire text according to the predicted confidence maps. Yao *et al.* [2] and Zhang *et al.* [61] applied FCN to an image to extract binary text block regions, where a set of post-processing (e.g. MSER based character extraction and character grouping) is performed to produce a text line candidate. The PixelLink [67], as a pixel-level version of SegLink [12], learns to predict whether two adjacent pixels belong to the same text instance by adding link prediction to each pixel. Border learning method [68, 14], shown in Fig. 2.7 (c), treats pixels into three categories, which are text, border and background, and extracts text instances under the assumption that border can well separate text instances.

Specific Targets

Different from the general object detection systems, text detection systems are usually designed for diverse purposes, attempting to approach different difficulties in detecting scene texts. A rough classification is illustrated below:

Long Text: Text is diverse in size and aspect ratio, making general object detection

methods fail to detect long texts. To address this problem, R^2 CNN is proposed to multiple RoI pooling with different pooling sizes (e.g. 7, 3×11 , and 11×3) on the proposed regions by an RPN (Region Proposal Network) to capture the various aspect ratios and sizes. Then the pooled feature are concatenated for classification and box regression. Another methods address this problem by detecting local sub-text components. SegLink [12] detects sub-text components and combines them into text instance according to predicted local links. PixelLink [67] enhances the performance of SegLink by performing the procedure of SegLink on pixel level. Corner localization [13] predicts four corners of a text from which a instance level bounding box is retrieved by sampling and grouping.

Multi-oriented Text: Skewed texts are common in real world. The traditional axis-aligned bounding boxes incorporating excessive noisy background will affect the performance of subsequent recognition stage. Therefore, text detection should be orientation-aware. As the extensions of anchor-based methods, [6, 7] applied rotated default boxes in their anchor system and additionally predict rotation offsets. Similarly, rotated region proposals is adopted in [11] with six different default orientations. Regression-based methods [12, 3] directly predict the rotation or the rotation-insensitive vertex locations. Further, Liao *et al.* [10] adopted *Rotated Response Network (RRN)* [74] to realize orientation in-variance and Wang *et al.* [75] proposed a *Instance Transform Network (ITN)* that learns to predict appropriated affine transform of RCNN's sampling grid, which fits the text's orientation and results in better classification and regression.

Irregularly Shaped Text: Text with irregular shape poses a new challenge to scene text detection due ambiguity introduced by rectangle boxes. The extracted rectangle regions may contain large portion of noisy background or even other text instances, which degrades the performance of text recognition. Extending from the quadrilateral bounding box, a bounding box with more vertices is desirable to locate a curve text with a tighter boundary. Liu *et al.* [21] used a 14-vertex polygon to represent a text object with arbitrary shape and a bidirectional Long-short Term Memory (LSTM) to predict the vertices of a polygon iteratively. Lyu *et al.* regarded curve text detection as an instance segmentation

Figure 2.8: TextSnake [14]

problem and modified the Mask R-CNN framework into an end-to-end system that simultaneously extracts curve text regions and characters for recognition. Long *et al.* [14] presented a novel representation for curve text objects, where a text instance is represented as a series of sliding round disks along the text center line (TCL) and according with the running direction of the text instance, which is illustrated in Fig. 2.8. With the new representation, they proposed a model named TextSnake to detect irregular texts. As shown in Fig. 2.7 (c), TextSnake learns a set of local attributes, including TCL/non-TCL, textregion/non-text-region, radius, and orientation. The intersection of TCL pixel and text region pixels reveals instance-level objectness and the local geometric attributes (e.g. radius and orientation) help to produce ordered point list of a TCL and reconstruct the shape of a text object.

2.2.2 Recognition

In this section, we briefly introduce the deep learning based text recognition methods. As illustrated in section 2.1, the traditional text recognition methods typically consist of three steps, including a pre-processing, character segmentation and recognition. Character segmentation is considered as the most challenging and restricts the recognition performance due the complex background, irregular text layout and text-like patterns. The deep learning based recognition methods take the advantages of sequence-to-sequence

Figure 2.9: Scene Text Recognition Framework: (a) basic idea; (b) RNN stacked with CNN; (c) FCN based sequence prediction; (d) Attention-based models; (e) Character-guilt attention; (f) Sequence reordering

learning and recently proposed methods inherit from Connectionist Temporal Classification [76] and attention mechanism [77], shown in Fig. 2.9.

CTC-based Methods

The CTC method is a general sequence recognition method which has been widely adopted in speech recognition and text recognition. It is primarily proposed by Graves *et al.* for handwriting recognition, shown in Fig. 2.9 (a). CTC computes conditional probability $p(L|Y)$, where $Y = y_1, y_2, \dots, y_T$ represents the per-frame prediction of a sequence and L denotes the label sequence. The CTC method is the fundamental of the existing deep learning based scene text recognition system.

Shi *et al.* [78] proposed a model called CRNN with CNN and RNN stacked together, shown in Fig. 2.9 (b). In CRNN, the CNN is used to extract sequential representations from a cropped image with text. Then, the extracted representations are fed into an RNN,

which predicts label distribution for each frame. Finally, the transcription layer translates the per-frame prediction into final sequence label using CTC. Instead of using RNN to predict per-frame labels, Gao *et al.* [79] adopted fully convolutional neural network to encode sequential information, depicted in Fig. 2.9 (c). The intuition of this approach is that a text consisted of multiple characters is dominated by short-term dependency, which can be captured by a CNN with sufficient receptive field. Yin *et al.* [80] also avoid the RNN in their workflow to speed up the recognition. They proposed sliding window CNN that simultaneously detects and recognizes characters. The detected characters are subsequently fed into a CTC layer to generate transcripts.

Attention-based Methods

The attention mechanism [77] was first proposed in the field of machine translation. As illustrate in Fig. 2.9 (d). It acts as an implicit feature selection mechanism that produces current prediction by dynamically reweighting the input representations at all time frames. This method is widely adopted to improve the performance of scene text detection in recently years.

Cheng *et al.* [81] observed the attention drift problem in existing attention-based methods and proposed an Focus Attention Network (FAN) to attenuate it by introducing per-frame attention supervision. As shown in Fig. 2.9 (e), the character-level label is adopted as additional supervised signal to calibrate the attention map and make it align to each character. In [82], Bai *et al.* proposed an edit probability (EP) metric to describe the misalignment between ground-truth string and probability distributions output by an attention module, Fig 2.9 (f). The EP allows the training process to focus on the missing, superfluous and unrecognized characters, which alleviates the impact of misalignment.

2.2.3 End-to-End Systems

An end-to-end text detection and recognition system is an integration of text detection and recognition. The mainstream end-to-end systems can be categorized into *two-step methods* and *end-to-end trainable methods*.

Generally, the two-step methods first detect single characters [33], words [59] or text lines [1], and then use another recognition model to interpret input cropped images into transcripts. Jaderberg *et al.* [1] used a combination of Edge Box proposals [83] and a trained aggregate channel features detector [84] to generate candidate word bounding boxes. The generated bounding boxes are fed into their recognition model to extract transcripts. Liao [8] adopted SSD [19] to propose rectangle text region, where a CRNN [78] is applied for further text recognition. Lyu *et al.* developed a shape-free text spotting flow by using a Mask R-CNN to reveal the existence of text instance and the locations of each characters. A post-processing is performance based on the output of Mask-RCNN to connect the separated characters into a text instance.

The major deficiency of two-step method is that the error propagation between recognition branch and detection branch will lead to an sub-optimal performance. Researchers start to develop an end-to-end trainable text detection and recognition system based on differentiable computing graph. Bartiz *et al.* [85] proposed a weakly-supervised method to train an end-to-end text spotter without bounding box level annotations. Li *et al.* developed an end-to-end detection and recognition system by replacing the RCNN branch in Faster RCNN with an encoder-decoder network for text recognition. Similar paradigm was also adopted in [86, 5, 4], where text proposals are mapped into fixed height feature maps by bilinear interpolation and transcripts is generated by a CTC-based recognition module.

2.2.4 Benchmarks and Evaluation Protocols

Data has been widely recognized as the fuel of deep learning technologies. In scene text detection and recognition, new datasets aimed at different challenges have been proposed, boosting the performance of both text detection and recognition. In this part, we will introduce the existing widely adopted benchmark datasets and evaluation protocols.

Benchmark Datasets

We mainly introduce the datasets used in our experiments and summarize the features of the other datasets in Tab. 2.2.4.

ICDAR 2013 [87]: It contains mostly horizontal texts with some text slightly oriented. It has been widely adopted for evaluation of scene text detection methods. The dataset consists of 229 training images and 233 testing images for the task of text detection, from which 848 training images and 1095 testing images are cropped for the task of text recognition.

ICDAR 2015 [88]: It is the Challenge 4 of ICDAR 2015 Robust Reading Competition. This challenge features incidental scene text images taken by Google Glasses without care of positioning, image quality and viewpoint. The dataset exhibits large variation in orientation, scale and resolution. The dataset is only for text detection and contains 1000 training images and 500 testing images with word-level annotations.

ICDAR 2017-MLT [89]: It is a large multi-lingual text dataset for text detection. It contains 7200 training images and 1800 testing images with text instances from 9 languages. The text regions are constrained to rectangles but can be arbitrarily rotated.

ICDAR 2019-ART[90]: It is a challenge of scene text understanding, which can be broken down into scene text detection, recognition, and spotting problems. It is the largest dataset containing texts with arbitrary shape. There are 5603 training images and 4563 testing images.

Total Text [91]: It is a newly released benchmark for curve text detection. In addition

Table 2.1: Existing datasets and benchmarks for scene text detection and recognition.

Datasets	Image Num (train/test)	Text Num (train/test)	Orientation	Language	Characteristics	Detection	Recognition
ICDAR 2003	258/251	1110/1156	Horizontal	EN	-	✓	✓
ICDAR 2013	229/233	848/1095	Horizontal	EN	-	✓	✓
ICDAR 2015	1000/500	-/-	Multi-oriented	EN	Blur, Small	✓	✓
ICDAR 2017 RCTW	8034/4229	-/-	Multi-oriented	CN	-	✓	✓
Total Text	1255/300	-/-	Curved	EN	Polygon Label	✓	✓
CTW1500	1000/500	-/-	Curved	EN	Polygon Label	✓	-
MSRA-TD500	300/200	1068/651	Multi-oriented	CN, EN	Long Text	✓	-
HUST-TR400	400/-	-/-	Multi-oriented	CN, EN	Long Text	✓	-
CASIA-10K	7000/3000	-/-	Multi-oriented	CN, EN	-	✓	✓
ICDAR 2017 MLT	7200/1800	-/-	Multi-oriented	9 LANGUAGE	-	✓	-
ICDAR2019 ART	5603/4563	-/-	Curved	CN, EN	-	✓	✓

to rectangle text instances, this dataset especially includes curve text instances. The curve text instances are labeled with polygons with the various number of vertices. We re-sample the vertices and generate a fixed quadrilateral for each text instance as the ground-truth polygon.

CTW1500 [21]: It is another curve text dataset released recently. It contains 1500 images in total with 1000 images for training and 500 images for testing.

MSRA-TD500 [29]: It is a small dataset composed of multi-lingual, arbitrarily oriented and long text lines. It includes 300 training images and 200 testing images with line-level annotation.

Evaluation Protocols

We briefly summarize the evaluation protocols for both scene text detection and recognition. The performance of a scene text detection algorithm is evaluated by its precision, recall and F-score. Precision (P) is calculated as the proportion of predicted text instances that can be matched to ground true. Recall (R) is the proportion of ground true labels that have correspondents in the predicted list. F-score (F) is a comprehensive evaluation by taking precision and recall into account, which is computed by

$$F = \frac{2 \times P \times R}{P + R}. \quad (2.1)$$

Figure 2.10: Example images in ICDAR 2013, ICDAR 2015, ICDAR 2017, ICDAR 2019, CTW1500, Total Text and MSRA-TD500.

There are mainly two different protocol to compute precision and recall, which are *PASCAL* and *DetEval*. The *PASCAL* protocol [92] measures the Intersection over Union (IoU) between prediction and ground-true. In scene text detection, the prediction can be a text region represented by a rectangle bounding box or polygon. The IoU is obtained by computing

$$IoU = \frac{Area(B_p \cap B_g)}{Area(B_p \cup B_g)}, \quad (2.2)$$

where B_p and B_g are the predicted bounding box (polygon) and ground-true bounding box (polygon). After obtaining the IoUs between prediction and ground-true, we compute the true-positives (TP), false-positives (FP) and false-negatives (FN). The TP is defined as the number of predictions that match the ground-true with $IoU > 0.5$. The FP is the number of predictions that do not match with any ground-true (with $IoU < 0.5$). The FN denotes the number of ground-true that are not detected. The precision is computed by

$$P = \frac{TP}{TP + FP}, \quad (2.3)$$

and the recall is computed by

$$R = \frac{TP}{TP + FN}. \quad (2.4)$$

The *DetEval* protocol imposes constraints in computing true-positives. A true-positives in *DetEval* is defined as the number of prediction that match the ground-true with

$$\frac{Area(B_p \cap B_g)}{Area(B_p)} > 0.6 \quad (2.5)$$

and

$$\frac{Area(B_p \cap B_g)}{Area(B_g)} > 0.7. \quad (2.6)$$

Most of the dataset adopt either or both of two evaluation protocols with small modifications. In ICDAR 2003 [93], the recall is calculated in a way similar to IoU, but it is defined as the intersection over minimum bounding box containing both prediction box and ground-true box. The ICDAR 2013 [94] further introduces *one-to-one*, *many-to-one* and *one-to-many* matching based on the *DetEval* protocol. The MSRA-TD500 computes a true-positive by considering both IoU (> 0.5) and rotation difference ($< \frac{\pi}{4}$) between predicted boxes and ground-true boxes.

The performance evaluation in scene text recognition is much simpler than that in scene text detection. The predicted text string is compared to the ground true directly to measure the character-level recognition rate (i.e. how many characters are recognized) or word-level (whether the predicted word is 100% correct).

In summary, both the scene text detection and recognition systems are evolving towards the end-to-end trainable neural networks. For the detection system, various specialized methods are developed in different scenarios, such as long text, multi-orientation, irregular shapes, etc. In terms of recognition, the CNN-plus-RNN paradigm and the attention-based methods become the mainstream. The research of fully end-to-end system which integrates detection and recognition into one neural network model has become popular in the computer vision society and achieved much progress.

Chapter 3

Bottom-up Scene Text Detection: Markov Clustering Networks

3.1 Introduction

Detecting structural objects in an image is a ubiquitous problem in real-world. Powered by the recent advances in Convolutional Neural Networks (CNNs), the object detection system has achieved human-level accuracy with real-time processing capability [25, 17, 18, 19]. Despite the progresses made in general object detection, we still confront problems in detecting objects in a specific application area.

In scene text detection, existing CNN-based methods may fail when producing bounding boxes with extremely large aspect ratio or unsupported orientation [95, 12]. These methods [17, 18, 19] are inspired by the advance of general object detection. A text is treated as an general object whose location will be directly obtained by an anchor-based detection system or further refined by a regional detector. The performance is mainly constrained by the anchor-based detector and easily affected by the variation of text geometry. A higher variation of text geometry leads to a lower matching rate between ground-truth text and anchors than that in general object detection. The reference boxes need to be much carefully designed to ensure sufficient number of matching, which

is realized by diversifying the reference boxes to cover the population of ground-truth boxes. Since each reference box is corresponding to a dedicated classifier and regressor, excessive references with insufficient training samples for each can cause under-trained classifiers and regressors, which greatly degrade performance.

Another factor that potentially degrades the performance is the limited receptive field of CNNs. A demonstration of this problems is shown in Fig. 3.1. The text bounding boxes in Fig. 3.1 (a) are produced by a Faster R-CNN model, which fails to retrieve the text object "hellmann" with a single bounding box. In Fig. 3.1 (b), "hellmann" is detected by an enhanced Faster RCNN with deformable convolution [96] introduced. The deformable convolution provides the capability of dynamically adjusting the receptive field to fit the shape of interested objects. Although the detection result shows slight improvement when handling long text object "hellmann", the deformable convolution enhanced Faster RCNN still fails to generate a correct result. This is because the CNN model with a limited size of receptive field is not be able to perceive the whole scene text object due to its large aspect ratio and varying orientation. In scene text detection, the detection targets could include words and text lines with various shapes. In this case, a fixed receptive field may not be able to capture spatial dependency of a text with large aspect ratio, which results in split bounding boxes for a target. Even though the deformable convolution with adaptable receptive field is adopted, observation at a single location is not fully reliable to predict the objectness and geometry of a text.

To address these issues, an unified framework called Markov Clustering Network (MCN) is proposed for detecting scale-varying and arbitrarily oriented texts. It is an end-to-end trainable model describing both the local correlation and semantic information of an object with *stochastic flow graph*. As shown in Fig. 3.2, equidistant and overlapping regions are considered as nodes of SFG with edges weighted by flow values. Nodes belonging to the same object are strongly connected by the flows and will be grouped together by applying fully paralleled Markov Clustering (MC) on the *stochastic flow*

Figure 3.1: All the detectors are trained with the same data. The ground-truth boxes are labeled in red and the detected boxes are labeled in yellow. (a) Bounding boxes proposed by Faster R-CNN; (b) Bounding boxes proposed by Faster R-CNN with deformable convolution; (c) Bounding boxes proposed by MCN. Rectangles in red are ground-truth boxes and those in yellow are predictions.

graph (SFG). Bounding boxes are produced based on the generated clusters with post-processing.

In contrast with the top-down methods [17, 18, 19], our method predicts bounding boxes in a bottom-up manner. Essentially, the MCN predicts instance-level objectness by merging the dense object predictions according to the local correlation measurements. This framework can naturally detect texts with arbitrary size and orientation, shown in Fig. 3.1 (c). Moreover, there is no reference box matching included in the training flow, which eliminates the under-trained detectors and thus decouples the performance from the ground-truth box statistics. The evaluation on public benchmarks proves that our method is robust to large variation of scale, aspect ratio and orientation. Our method achieves the state-of-art performance with much faster inference. The contribution of this work is summarized as follows:

- A bottom-up method for scene text detection is proposed which assembles local predictions into object bounding boxes by performing Markov Clustering on Stochastic Flow Graph;
- Markov Clustering is regarded as a set of special differentiable neural network layers and an end-to-end training method is developed for learning graph clusters from image data;

Figure 3.2: (a) Input image; (b) Predicted stochastic flow graph by MCN: Nodes correspond to the equidistant overlapping regions in the image and the connections between nodes are referred as flows. The flow intensity is visualized as the width of an edge. A strong flow results in a wide edge while weak flow leads to a narrow edge; (c) Extracted clusters from stochastic flow graph presented in (b) by Markov Clustering; (d) Bounding boxes generated from the clustered nodes in (c).

- The proposed inference process is fully paralleled on GPUs and achieves realtime processing capability with frame rate of 34 FPS, which means to $1.5\times$ speedup when compared with fastest scene text detection algorithm.
- Our method outperforms existing scene text detection methods in detecting arbitrarily oriented text objects, and achieves new state-of-art performance on challenging MSRA-TD500 dataset with precision of 0.88, recall of 0.79 and F-score of 0.83.

3.2 Markov Clustering Networks

3.2.1 Overview

Markov Clustering Network (MCN) is an object detection method based on graph clustering. An $H \times W$ image is translated by MCN into a spatial feature map which will be further constructed into a latticed graph $G(V, E)$ called *stochastic flow graph*. The

nodes V in G correspond to the feature vectors extracted from the overlapping regions of the image. The edges E are weighted by the flow values f_0, f_1, f_2 and f_3 predicted by a Convolutional Neural Network (CNN). They are 2D maps with size of $\frac{H}{U} \times \frac{W}{U}$ denoting the connection intensity or interaction to current node or its three neighbors. Note that U is defined as the stride between nodes. In our prediction framework, the presence of an object is jointly represented by nodes with strong connections to each other, and the background region is represented by isolated nodes. Therefore, detecting an object is equivalent to predicting the flow values and then grouping the nodes according to their connection intensities. Given the flow values predicted by MCN, we extract the objectness by performing Markov Clustering (MC) [97] on G . The strong connected nodes are grouped into clusters representing objects. By mapping the nodes of a cluster back to the input image, the corresponding bounding boxes can be produced by simple post-processing.

3.2.2 Object Representation by Stochastic Flow

The existing object detection methods can be categorized as top-down methods, where the detection relies on coarse global observation of an image [19, 18, 17]. Due to the absence of the local information, these methods usually predict offset of the object size and orientation relative to predefined references (reference boxes) [7, 12]. Designing these references is task-specific which can hardly cover all cases and will degrade the detection robustness. This problem is getting worse in detecting objects with arbitrary aspect ratio and various orientation. If the object geometry is not well-supported by the references, large amounts of failure will occur when detecting these objects.

Our method considers the object detection in a bottom-up manner to solve the problems mentioned above. In our method, as shown in Fig. 3.3, an input image is converted via MCN to a *stochastic flow graph* G with nodes V and directed edges E weighted by stochastic flows f_0, f_1, f_2 and f_3 . For a given node $V(i, j)$, we uses a

Figure 3.3: Markov Clustering Network.

Convolutional Neural Network (CNN) to predict corresponding flows $f_0(i, j)$, $f_1(i, j)$, $f_2(i, j)$ and $f_3(i, j)$. They are positive and sum up to 1. Due to the nature of 2D convolution, the CNN provides a lattice-formed prediction for each flow with a stride U . An object is abstracted as nodes connected by the outgoing flows f_1 , f_2 and f_3 , while the background region is represented by nodes isolated by the self-loop flows f_0 . To be precise, the three outgoing flows f_1 , f_2 and f_3 are respectively pointing the bottom, right and left nodes of the current node. Since the nodes have corresponding spatial relation in the original image, the presence as well as the geometry (size and orientation) of an object can be represented by nodes and their flows, which is insensitive to variation of size and orientation.

From the point of probability, the *stochastic flow graph* is actually modeling the Markov random walk process, where each node denotes a state in a Markov chain and the corresponding directed weighted edges represent the transition probabilities of this state. For a random walk process starting at a given node $V(i, j)$, there exists a stationary distribution (or flow distribution) $P(V|V(i, j))$ describing possible destination nodes of this process. Specifically, the node with maximum value in $P(V|V(i, j))$ is denoted as the attractor of $V(i, j)$. Therefore, the strongly connected nodes can be regarded as nodes

with the same attractor [97]. This interpretation provides us a probabilistic description of flows and clusters. Moreover, it allows us to uniquely represent an instance-level object region with an attractor, which is the fundamental of our detection method.

3.2.3 Detecting Object by Markov Clustering

Based on the probabilistic interpretation and property of *stochastic flow graph*, we apply Markov Clustering (MC) to extract the instance-level object regions. Markov Clustering is an algorithm to identify the strongly connected nodes and group them into clusters. In Markov Clustering, a flow matrix M_0 is constructed from G with entry $M_0(m, n)$ representing the flow value from node $V(i_n, j_n)$ to node $V(i_m, j_m)$. It should be noted that for convenience we use both 1D and 2D notation, alternatively, to index a node. The transformation between 1D notation m and 2D notation (i_m, j_m) can be represented by $m = i_m + \frac{H}{U} \cdot j_m$. The n -th column $M_0(:, n) \in \mathbb{R}^{\frac{H}{U} \cdot \frac{W}{U}}$ of M_0 represents the transition probability of a Markov random walk starting at node V_n , which is denoted as $P_0(V|V(i_n, j_n))$. Markov Clustering is actually computing the stationary distribution $P(V|V(i, j))$ for each node. It consists of a set of iterations including matrix-matrix multiplication and non-linear transformation, which are illustrated as follows:

Expand: Input M_{t-1} , output M_t .

$$M_t = M_{t-1} * M_0 \quad (3.1)$$

Inflate: Input M_t , output M_t .

$$M_t(m, n) = \frac{M_t(m, n)}{\sum_l M_t(l, n)} \quad (3.2)$$

Prune: Input M_t , output M_t .

$$M_t(m, n) = 0, \text{ if } M_t(m, n) < \text{threshold} \quad (3.3)$$

where M_t is the intermediate result at t -th iteration and N is the number of iterations for convergence. The expansion step spreads the flows out of a node to its potential new node. It enhances the flows to the nodes which are reachable by multiple paths. The inflation step and pruning step are meant to regularize the iteration to ensure convergence by introducing a non-linearity into the process, while also have the effect of strengthening intra-cluster flows and weakening the inter-cluster flows [97]. The pseudo-code for Markov Clustering is presented in Algorithm 1.

Algorithm 1 Markov Clustering

```

1: Initialize:
2:  $t = 0$ 
3: for  $m, n$  do
4:    $M_0(m, n) = \text{flow value from node-}m \text{ to node-}n$ 
5: end for
6: while  $t < N$  do
7:    $M_t = \text{Expand}(M_{t-1})$ 
8:    $M_t = \text{Inflate}(M_t)$ 
9:    $M_t = \text{Prune}(M_t)$ 
10: end while
11: Output:  $M_N$  as a clustering.

```

At the start of the process, the outgoing flow distribution of a node is smooth and uniform, and becomes more and more peaked as the iterations are executed. The columns of M_t corresponding to the same cluster will converge to the same one-hot vector. It is reflected on G that nodes within a tightly-linked group will flow to the same attractor at the end, which helps to identify any potential cluster. In addition, Markov Clustering does not require predefined number of clusters, and due to the parallelizability of three operations, Markov Clustering can be fully parallelized on GPUs.

3.2.4 Learning Clustering with Flow Labels

Locating Attractors for Clusters

As illustrated previously, the converged flow matrix M_N describes the flow distribution of possible attractors for each node. Therefore, labeling clusters is equivalent to labeling an

attractor for each node. Defining the nodes within the same bounding box as a cluster, we compute the attractor for this cluster based on the orientation of the ground-truth bounding box. For a left-oblique (right-oblique) text box, we label bottom-left (bottom-right) node as the attractor. The motivation behind is that we want to associate the direction of stochastic flows to the text orientation. This approach gives a consistent ground-truth flow, which encourages the convergence of training. As shown in Fig. 3.4 (a), given a ground-truth bounding box, we firstly compute the coordinates of D , which is the intersection between the major axis and the lower short-side of the bounding box. This step is aiming to coarsely locate the attractor. Second, we draw a horizontal line l_1 that traverses the node with lowest Y -coordinates in the bounding box region, and a vertical line that l_2 traverses the nearest node from D . Finally, the intersection node between l_1 and l_2 is determined as the attractor. To ensure attractor being in a bounding box, we adjust the bounding box size, which may introduce new nodes into it. Notable, for a horizontal text, the lower short-side is specified as the right-side of the ground-truth box, which leads to an attractor located at the bottom-right of the ground-truth bounding box. This labeling strategy encourages MCN to predict a cluster with attractor at the bottom-right of a horizontal text object. It is experimentally observed that most of the horizontal text objects are predicted with consistent flows and a bottom-right attractor, shown in Fig. 3.10 There exist some cases with flows not so consistent, but the flow probability still converges to a unique attractor. The detection performance is actually insensitive to the slight flow ambiguity.

From Attractors to Cluster Labels

The Markov Clustering outputs the stationary distribution $M_N(:, m) \in \mathbb{R}^{\frac{H}{U} \cdot \frac{W}{U}}$ of potential attractors for each node. Thus, the ground-truth label for each node is defined as the target distribution $y_f(i_m, j_m) \in \mathbb{R}^{\frac{H}{U} \cdot \frac{W}{U}}$. As shown in Fig. 3.4 (c), we firstly make 2D mask to record the 1D attractor index $m \in [0, \dots, \frac{H}{U} \cdot \frac{W}{U} - 1]$ for each node. For the nodes within an object region, they share the same attractor index, while for an node corresponding

Figure 3.4: (a) Computing the attractor P_{attr} given a ground-truth bounding box; (b) Adjusting a ground-truth bounding box to include the attractor; (c) Assigning an attractor index for each node; (d) Generating 3D cluster label for each node based on (c).

to the background, it becomes the attractor of itself. Based on the attractor mask, we generate a 3D cluster (flow) label $y_f \in \mathbb{R}^{\frac{H}{U} \times \frac{W}{U} \times (\frac{H}{U} \cdot \frac{W}{U})}$ describing the target stationary distribution for all nodes, which is shown in Fig. 3.4 (d). For specific node $V(i_m, j_m)$ with attractor $V(i_k, j_k)$, the target distribution $y_f(i_m, j_m)$ is a one-hot vector with k -th entry labeled as 1.

Loss Function

Given the converged flow distribution $M_N(:, m)$ for a node $V(i_m, j_m)$ and the target distribution $y_f(i_m, j_m)$, the loss function is represented by a cross-entropy loss between these two distributions:

$$L_f(i_m, j_m) = -y_f(i_m, j_m) \cdot \ln(M_N(:, m)), \quad (3.4)$$

Figure 3.5: Computing graph of Markov Clustering. It takes initial flow matrix M_0 as input and outputs a converged flow M_N^P to compute $C_f(M_0)$.

and the flows of all nodes are globally optimized by minimizing the mean cross-entropy error represented by:

$$C_f = \frac{1}{H/U \cdot W/U} \sum_{i_m} \sum_{j_m} L_f(i_m, j_m) \quad (3.5)$$

Gradients of Markov Clustering

An end-to-end supervised training requires the differentiability of all the operations in a model and the feasibility of labeling the data. In this section, we focus on the differentiability of Markov Clustering. The operations included in Markov Clustering can be treated as special neural network layers, which are differentiable. We visualize the operations with a computing graph which computes the stationary distribution M_N and corresponding cross-entropy loss $C_f(M_0)$ given flow matrix M_0 and target distribution y_f . In Fig. 3.5, each node represents one operation in Markov Clustering, and the directed edges show the data flow throughout the whole clustering process for N iterations. The output data of an operation is marked above the edge and corresponding gradient $g_{(\cdot)}$ is marked below. From the computing graph, gradient of cost function of stochastic flow C_f

respecting M_0 is derived by using the chain rule illustrated below:

$$g_{M_N^P} = \partial C_f / \partial M_N^P, \quad (3.6)$$

$$g_{M_t^I} = g_{M_t^P} \cdot f_P'(M_t^I), \quad (3.7)$$

$$g_{M_t} = g_{M_t^I} \cdot f_I'(M_t), \quad (3.8)$$

$$g_{M_{t-1}^P} = M_0^T \cdot g_{M_t}, \quad (3.9)$$

$$g_{M_0}(t) = g_{M_t} \cdot (M_{t-1}^P)^T, \quad (3.10)$$

$$g_{M_0} = \partial C_f / \partial M_0 = g_{M_0^P} + \sum_{i=1}^N g_{M_0}(i), \quad (3.11)$$

$$f_P'(x) = 1 \text{ if } x > 0, \text{ else } 0, \quad (3.12)$$

$$f_I'(x) = 1. \quad (3.13)$$

The computing graph for C_f composes of a main data path from M_0 through a series of MC iterations to C_f and a set of side paths directly connecting M_0 to the input of expansion node. Therefore, the gradient of C_f respecting M_0 is computed by summing all gradients respecting M_0 input to all expansion node, as illustrated in Equation 3.11. In addition, to simplify the gradient computation, we set the threshold of pruning to be 0, making it be equivalent to a ReLU operation. Thus, the inflation becomes identical mapping with a gradient of 1. This trick will slightly increase the number of iterations for convergence but simplifies the gradient computation, leading to a faster training in general. In this testing phase, the threshold can be turned up for faster convergence.

3.3 Detail Implementation of MCN for Scene Text Detection

3.3.1 Neural Network Architecture

The architecture of MCN, inference flow and training flow are shown in Figure 3.6. An MCN consists of a CNN backbone network inherited from a pretrained VGG-16 model.

Figure 3.6: Implementation detail of MCN for scene text detection.

We remove all the fully-connected layers and output features of the *conv5_3* with $1/16$ resolution. For an input image size of $H \times W$, the *conv5_3* output is of size $H/16 \times W/16$. The *conv5_3* features are respectively fed to a Fore-/Background Subnetwork (FBN) and a Local Correlation Subnetwork (LCN). FBN detects multi-scale objects with a Feature Pyramid Network (FPN) [98] and a 2D-Recurrent Neural Network (2D-RNN). LCN predicts spatial and semantic correlation between adjacent image patches with stride of 16. The objects' presence probability $P \in (0, 1)^{H/16 \times W/16}$ and the local correlation measurements S_1, S_2 and $S_3 \in (0, 1)^{H/16 \times W/16}$ between current image patch and its three neighbors (bottom, right and left) produced by FBN and LCN respectively are translated into four flow maps f_0, f_1, f_2 and $f_3 \in \mathbb{R}^{+H/16 \times W/16}$. A latticed stochastic flow graph is constructed from flow maps which is further described by a flow matrix M_0 . By performing Markov Clustering on the SFG, we can group nodes that belongs to the same object together and generate instance-level bounding boxes based on Principle Component Analysis (PCA).

The MCN is end-to-end trainable with bounding box level labeling. As illustrated in Figure 3.6, the ground-truth bounding boxes are converted to node-wise object mask $y_o \in \{0, 1\}^{H/16 \times W/16 \times 2}$ and flow label $y_f(i_m, j_m) \in \mathbb{R}^{\frac{H}{16} \times \frac{W}{16} \times (\frac{H}{16} \cdot \frac{W}{16})}$, which are used to compute the Object Loss $L_o \in \mathbb{R}^{H/16 \times W/16}$, Object Cost $C_o = \frac{1}{H/16 \cdot W/16} \sum_{i_m} \sum_{j_m} L_o(i_m, j_m)$, Flow Loss $L_f \in \mathbb{R}^{H/16 \cdot W/16}$ and Flow Cost $C_f \in \mathbb{R}$. The total cost $C_{total} \in \mathbb{R}$ is computed by summing C_o and C_f together.

3.3.2 Bounding Box Generation

Given a vertex V_m in a cluster Ψ , we compute their coordinates in the input image $\omega_m = (i_m, j_m) \cdot \mu_{stride} + \mu_{offset}$, where $\mu_{stride} = 16$ and $\mu_{offset} = 8$. Then the bounding box size and orientation of each cluster are computed based on Principle Component Analysis (PCA). Given a set of coordinates $\omega = \{\omega_m | m = 1, 2, \dots\}$ of a cluster, we compute the its eigenvectors θ_1 and θ_2 as well as the corresponding eigenvalues λ_1 and λ_2 . The coordinates of the four corners of the bounding box are computed by:

$$\begin{aligned} c_1 &= A(\lambda_1 \cdot \theta_1 + \lambda_2 \cdot \theta_2) + \phi \\ c_2 &= A(\lambda_1 \cdot \theta_1 - \lambda_2 \cdot \theta_2) + \phi \\ c_3 &= A(-\lambda_1 \cdot \theta_1 - \lambda_2 \cdot \theta_2) + \phi \\ c_4 &= A(-\lambda_1 \cdot \theta_1 + \lambda_2 \cdot \theta_2) + \phi \end{aligned} \tag{3.14}$$

where ϕ is the center of the cluster and A denotes the scaling factor which is set to 1.75.

3.3.3 From Image to Stochastic Flow

Crucially, accurate object detection relies on correct flow prediction. In MCN, the flows f_0, f_1, f_2 and f_3 are the outputs of the Flow Mapping Layer (FML) with regional object probability P and correlation measurement S_1, S_2 and S_3 as inputs. P is generated by the Fore-/Background Network (FBN), while S_1, S_2 and S_3 are output by Local Correlation Network (LCN). Both FBN and LCN are starting at the *conv5_3* of VGG-16 pretrained network.

Fore-/Background Network

As shown in Figure 3.7 (a), the Fore-/Background Network is an FPN-based network [98] with spatial recurrent components and *softmax* output to predict the object score $P \in (0, 1)^{H_{1/16} \times W_{1/16}}$. The output of *conv5_3* is further processed by a Feature Pyramid

Figure 3.7: (a) Architecture of Fore-/Background Network (FBN); (b) Feature Pyramid Network (FPN) fusing feature maps with different resolutions; (c) 2-dimensional Recurrent Neural Network (2D-RNN) encoding contextual representations; (d) Regional objectness classifier predicting presence of an object with stride of 16×16 pixels.

Network (FPN) and a 2-dimensional Recurrent Neural Network (2D-RNN) successively. In FPN shown in Figure 3.7 (b), input with size of $H/16 \times W/16$ is processed by four convolutional blocks with 2×2 pooling layers to obtain additional feature maps with resolution of $1/32$, $1/64$, $1/128$ and $1/256$. These feature maps together with the input are fused to resolution of $1/16$ by deconvolution consisted of layer-wise addition, bilinear upsampling and convolution. By fusing features with different resolution in a pyramid manner, our method have larger capacity to detect multiscale objects with less parameters. Subsequently, the output of FPN is fed to an 2D Recurrent Neural Network (2D-RNN) before region-based classification. We consider a spatial feature map as a 2D sequence which can be directly analyzed by a 2D-RNN. The structure of the proposed 2D-RNN is shown in Figure 3.7 (c). A 2D-RNN is composed of two Bidirectional RNNs (RNN-H and RNN-V), which are applied to the rows and columns of the input feature map independently. As shown in Figure 3.7, the outputs of 2D-RNN is constructed by concatenating two feature maps produced by RNN-H and RNN-V with size of $H_{1/16} \times W_{1/16}$ along depth axis. Finally, a region-based classification is performed on the output feature map by a 2-layer convolutional network with *softmax* output, Figure 3.8 (d).

Figure 3.8: Local Correlation Network (LCN) with four convolution blocks and a softmax layer outputs correlation measurements S_1 , S_2 and S_3 between current anchor and its three neighbors (bottom, right and left).

To predict the semantic and spatial correlation between adjacent subregions, we build another subnetwork with additional four convolutional blocks and a *softmax* classifier starting at *conv5_3*, shown in Figure 3.8. The network outputs three correlation measurements S_1 , S_2 and $S_3 \in (0, 1)^{H_{1/16} \times W_{1/16}}$ representing the semantic and spatial correlation between current anchor and its three neighbors (bottom, right and left) respectively. As the *conv5_3* features is corresponding to subregions of input image with stride of 16, the LCN is actually measuring the correlation among these overlapping subregions. Together with output of objectness network P , S_1 , S_2 and S_3 are mapped to the Stochastic Flow f_0 , f_1 , f_2 and f_3 by Flow Mapping Layer (FML).

Flow Mapping Layer

The Flow Mapping Layer (FML) is point-wise non-linear function with input of P , S_1 , S_2 and S_3 and output of f_0 , f_1 , f_2 and f_3 . The mapping is shown below:

$$f_0 = e^{-\alpha[1-\mu(1-P)] \cdot [S_1^2 + S_2^2 + S_3^2]} \quad (3.15)$$

$$f_1 = (1 - f_0) \cdot \frac{S_1}{S_1 + S_2 + S_3} \quad (3.16)$$

$$f_2 = (1 - f_0) \cdot \frac{S_2}{S_1 + S_2 + S_3} \quad (3.17)$$

$$f_3 = (1 - f_0) \cdot \frac{S_3}{S_1 + S_2 + S_3} \quad (3.18)$$

$$\mu(x) = \frac{1}{1 + e^{-\beta(x-\gamma)}}. \quad (3.19)$$

Here, f_0 is actually the transition probability of *self-loop*, which is controlled by the likelihood of background ($1 - P$) and the correlation measurement between current vertex and its neighbors (S_1 , S_2 and S_3). It is designed to be weak for vertices within the same object region and to be strong for a vertex which corresponds to the background or is just the attractor of a cluster. This behavior is realized by firstly measuring the correlation intensity ($S_1^2 + S_2^2 + S_3^2$) modulated by an *on-off* function $\mu(x)$, and then projecting it to the exponential space. $\mu(x)$ is parameterized by trainable variables β and γ . It takes $1 - P$ as input and produces an on-off signal to control f_0 . It will disables the effect of S_1 , S_2 and S_3 and drive f_0 approaching to 1 when a vertex is in the background region. Accordingly, the values of f_1 , f_2 and f_3 will be small, making all the background vertices to be isolated. In the object region, the correlation intensity S_1 , S_2 and S_3 take control of f_0 since $1 - P$ is small. In this case, f_0 will be large if weak correlation is measured and the vertex will become the attractor of a cluster. Otherwise, the vertices belongs to the same object region will be connected through f_1 , f_2 and f_3 and the flows of a cluster will end at the attractor.

3.4 Experiments

3.4.1 Datasets

We pre-train our model on synthetic scene text dataset SynthText and finetune it on three public scene text detection datasets, namely ICDAR 2013, ICDAR 2015 and MSRA-TD500, using the standard evaluation protocol proposed in [99, 88, 29].

ICDAR 2013 [87] contains mostly horizontal texts with some text slightly oriented. It has been widely adopted for evaluation of scene text detection methods. The dataset consists of 229 training samples and 233 testing samples. Due to there are too few training images, we first use 9000 images from ICDAR 2017 MLT [89] training and validation datasets to train a pre-trained model and then use 229 ICDAR 2013 training

images to fine-tune.

ICDAR 2015 [88] is the Challenge 4 of ICDAR 2015 Robust Reading Competition. This challenge features incidental scene text images taken by Google Glasses without care of positioning, image quality and viewpoint. The dataset exhibits large variation in orientation, scale and resolution. The dataset contains 1000 training images and 500 testing images with word-level annotations. In training, we first train our model using 9000 images from ICDAR 2017 MLT training and validation datasets, then we use 1000 ICDAR 2015 training images and 229 ICDAR 2013 training images to fine-tune our model.

MSRA-TD500 [29] is the first standard dataset that focuses on oriented text. The dataset is also multilingual, including both Chinese and English text. The dataset consists of 300 training images and 200 testing images. Different from ICDAR 2015, MSRA-TD500 is annotated at text line level. Since the number of training images is too few to learn a deep model, we also harness 400 images from HUSTTR400 dataset [100] as training data.

3.4.2 Experiment Details

Our model is pre-trained on SynthText and finetuned on real datasets. It is optimized by the standard SGD algorithm with a momentum of 0.9. Both training and testing images are resized to 512×512 . The batch size is set to 20. In pretraining, the learning rate is set to 10^{-3} for the first 60k iterations, and decayed by a factor of 10 for the rest 30k iterations. The finetuning on public benchmarks runs at learning rate of 10^{-5} with data augmentation proposed in [19]. In testing, the threshold used for Pruning is set to 0.15. Both the training and testing flows are implemented with TensorFlow [101] r1.1 on Dell Precision T7500 workstation with Intel Xeon 5600 processor, 40 GB memory and a NVIDIA GTX 1080 GPU.

3.4.3 Ablation Studies

MCN v.s. Segment-based Method

In the first experiment, we compare our MCN method with/without segment box regression with a baseline method (Local Links) which also utilizes local correlation to predict to combine multiple bounding boxes. The down-sampling factor is set to $1/16 \times 1/16$. The Local Links predicts fore/background and four local connections between nodes. The instance-level bounding boxes are generated from the segment boxes by finding the maximum connected component on the foreground regions. As shown in Tab. 3.1, our method is overall better than the Local Links with 4% improvement. In Local Links, only the local correlation of between adjacent node are considered. In Local Links, an unexpected mergence between two individual objects will occur if they are close to each other. In contrast, Markov Clustering allow construct a global prediction by assembling correlation information of different nodes, which greatly improves the robustness and flexibility.

MCN v.s. Faster R-CNN

This experiment illustrates the flexibility of MCN in detecting long text objects. The baseline model is a Faster R-CNN trained on public text detection datasets. We assign each anchor with 15 reference boxes and fix this configuration to all datasets. The reference boxes are the combination of 3 scales (16×16 , 64×64 and 256×256) and 5 aspect ratios (1, 2, 5, 10). Similar to the last experiment, the prediction is conducted on a feature map with down-sampling factor of $1/16 \times 1/16$. Tab. 3.2 compares the performance of Faster R-CNN and our MCN method. It shows that the MCN method overall outperforms the Faster R-CNN by a great margin (10%). It is because the performance of Faster R-CNN highly depends on the design of reference boxes. It is scenario specific and may not be able to generalize from one dataset to another. In detail, we visualize some typical examples to demonstrate the advantage of MCN in Fig. 3.9.

Table 3.1: Comparison between MCN and baseline model (Local-links) on public benchmarks.

Datasets	ICDAR 2013			ICDAR 2015			MSRA-TD500		
Methods	P	R	F	P	R	F	P	R	F
Local Links	84.9	81.1	82.9	77.9	64.6	70.6	83.3	70.3	76.2
MCN	88.0	86.5	87.2	80.4	69.9	74.8	87.8	77.8	82.5

Table 3.2: Performance comparison between MCN and Faster R-CNN in detecting scene texts. $\dagger$: VGG-16 + FPN, $\ddagger$: ResNet-50 + FPN

Datasets	ICDAR 2013			ICDAR 2015			MSRA-TD500		
Architectures	P	R	F	P	R	F	P	R	F
Faster R-CNN † [17]	77.8	70.9	74.2	63.7	73.4	68.2	73.1	66.9	69.9
Faster R-CNN ‡ [17]	79.6	71.4	75.2	74.6	64.9	69.4	72.9	67.4	70.0
MCN †	88.0	86.5	87.2	80.4	69.9	74.8	87.8	77.8	82.5
MCN ‡	91.9	83.7	87.6	82.1	76.1	79.0	87.1	80.4	83.6

It is observed that Faster R-CNN may fail when detecting text objects with extremely large aspect ratio, shown in Fig. 3.9 (a). In comparison, MCN shows better robustness in handling long text lines. The *stochastic flow graph* provides a elegant framework to represent the relation between nodes. This inherent information is utilized by MCN to model the long distance dependency, which improve the performance for long text objects.

Profiling the MCN

Fig. 3.10 visualizes the predicted flows by MCN. The input images with three orientations, horizontal, right-oblique and left-oblique are shown in Fig. 3.10 (a). Fig. 3.10 (b) profiles the activation maps including the object region prediction P , link scores S_1, S_2, S_3 , and stochastic flows f_0, f_1, f_2, f_3 . All the activation maps with size of 32×32 originally are upsampled to 512×512 for demonstration. According the activation map of f_0, f_1, f_2 and f_3 , we draw the dominative flows and label the attractor on the input image, which is

Figure 3.9: (a) Prediction results by Faster R-CNN; (b) Prediction results by MCN.

shown Fig. 3.10 (c). In Fig. 3.10 (d), the predicted bounding boxes and the ground-truth bounding boxes are labeled in yellow and red respectively.

On one hand, MCN shows the high accuracy in detecting text objects close to each other. As shown in objectness map P at the first row of Fig. 3.10, regions of multiple text objects merge together and we cannot generate bounding boxes directly from this map. The stochastic flows predicted by MCN captures the instance-level correlation and separate the merged regions into clusters. In some challenging cases with low quality P , the flow-based prediction can maintain good performance since an text object is jointly predicted by multiple nodes and their connections. On the other hand, the MCN method is flexible to handle text objects with different lengths and orientations. The orientation of an object is also represented by all flows within the object region jointly, resulting in a more accurate bounding box generation.

3.4.4 Performance Comparison

Table 3.3 compares our method with the published works on public datasets of scene text detection. On the ICDAR-13 dataset, our method reaches the state-of-art performance with precision of 0.88, recall of 0.87 and F-score of 0.88. On ICDAR-15, a slight

Figure 3.10: Flow profiling of detecting text with different orientations. (a) Input images; (b) Predicted objectness maps P , link maps S_1, S_2, S_3 , and stochastic flows f_0, f_1, f_2, f_3 ; (c) Visualization of Markov Clustering with dominative flows marked in yellow arrows and attractors marked in blue points; (d) Generated bounding boxes (in yellow) based on predicted clusters and according ground-truths labeled in red boxes.

Figure 3.11: Example results on ICDAR 2013 and MSRA-TD 500 datasets. Text instances with multiple scales and orientations are detected. The predicted bounding boxes by our method are labeled in yellow and the ground-truth bounding boxes are labeled in red.

Table 3.3: Localization performance on ICDAR-13, ICDAR-15 and MSRA-TD500.

Dataset	ICDAR-13			ICDAR-15			MSRA-TD500		
Methods	P	R	F	P	R	F	P	R	F
TextFlow [95]	0.85	0.76	0.80	-	-	-	-	-	-
Jaderberg <i>et al.</i> [1]	0.89	0.68	0.77	-	-	-	-	-	-
Zhang <i>et al.</i> [61]	0.88	0.78	0.83	0.71	0.43	0.54	0.83	0.67	0.74
Gupta <i>et al.</i> [59]	0.92	0.75	0.83	-	-	-	-	-	-
Yao <i>et al.</i> [2]	-	-	-	0.72	0.59	0.65	0.77	0.75	0.76
TextBox [7]	0.88	0.83	0.85	-	-	-	-	-	-
CTPN [70]	0.93	0.83	0.88	0.52	0.74	0.61	-	-	-
SegLink [12]	0.88	0.83	0.85	0.72	0.77	0.75	0.86	0.70	0.77
DeepReg [102]	0.92	0.81	0.86	0.82	0.80	0.81	0.77	0.70	0.74
MCN	0.88	0.87	0.88	0.72	0.80	0.76	0.88	0.79	0.83

performance drop is observed as compared to the existing text detection methods. Since most of the text objects are of size smaller than the node density (16×16 pixel), the flows predicted for these objects are weak, leading to inaccurate object detection. But MCN achieves a new state-of-art performance on the MSRA-TD500 dataset. As shown in Table 3.3, MCN outperforms the existing methods by a great margin with precision of 0.88, recall of 0.79 and F-score of 0.83. Different from the ICDAR-13 that consists of only horizontal text objects, MSRA-TD500 contains a large number of oblique and long text samples. The performance improvement in MSRA-TD500 shows that MCN is better at detecting multioriented text objects.

Fig. 3.11 demonstrates the bounding box prediction of MCN. The samples include both English and Chinese with different scales and orientations. The predicted bounding boxes are labeled in yellow and the ground-truths are labeled in red. As shown in Fig. 3.11, MCN detects multilingual text objects with various scales and orientations robustly. The flow clustering framework supports the different bounding box geometry flexibly. Compared with the region proposal based scene text detection algorithms

Table 3.4: Number of iterations, detection performance, and runtime.

	P	R	F	time (ms)
N=1	0.7	0.34	0.46	0.32
N=2	0.72	0.65	0.68	0.51
N=3	0.78	0.71	0.74	0.67
N=4	0.85	0.85	0.85	0.86
N=5	0.88	0.87	0.88	1.05
N=6	0.88	0.87	0.88	1.23
N=7	0.88	0.87	0.88	1.41
N=8	0.88	0.87	0.88	1.60

[70, 12, 102, 7], our method predicts more elaborated instance-level bounding boxes. As for the segmentation based methods [2, 103], our method involves much less heuristic operations. Both the flexibility and data-driven characteristics make MCN be superior to existing scene text detection methods.

3.4.5 Speed

In this section, we analyze the computation time of Markov Clustering. The Markov Clustering algorithm is implemented based on CUDA 8.0 with cuDNN 5 library [104].

We profile the computation time on Table 3.4, as well as according precision, recall and F-score of bounding prediction with different N . In general, the computation time of Markov Clustering increases linearly with the increase of N . The detection performance also increases as N increases, since the flow matrix M_t requires sufficient number of iterations for convergence. Fortunately, it only takes few iterations for convergence to reach the best detection performance. As shown in Table 3.4, MCN reaches its best performance with $N = 5$ and it takes only 0.86 ms to compute the clusters. This computing time is negligible when compared to the whole inference time of over 25 ms.

We also compare the inference speed in FPS with the recently proposed scene text

Table 3.5: FPS comparison on ICDAR-13 with input size of 512×512 .

Methods	P	R	F	FPS
Yao [29]	0.63	0.63	0.60	0.14
Gupta [59]	92.0	75.5	83.0	15
TextBox [7]	0.88	0.83	0.85	20.6
CTPN [70]	0.93	0.83	0.88	14.2
SegLink [12]	0.88	0.83	0.85	20.6
MCN	0.88	0.87	0.88	34

detection methods on ICDAR-13 dataset. As shown in Table 3.5, our method achieves state-of-art performance and outperforms the existing methods with $1.5\times$ speedup. This is owing to flow-based method, which can tolerate inaccurate fore-/background prediction and thus maintains the same performance with less network parameters.

3.5 Conclusion

In this work, we present a novel text detection framework called Markov Clustering Network, which predicts instance-level text boxes in a bottom-up manner. The scene text detection is reconsidered as a graph clustering problem and Markov Clustering is introduced to assemble both global and local information. We further interpret the Markov Clustering as a set of differentiable neural network layers, and develop an end-to-end training method allowing learning graph clustering from image data. MCN shows superiority in the sense of accuracy, robustness and speed. MCN outperforms the existing scene text detection algorithms in detecting multiscale and multioriented text objects. It also achieves $1.5\times$ speedup in comparison with the state-of-art algorithm.

Chapter 4

Enhancing Markov Clustering

Networks: Two-dimensional

Position-aware Attention

4.1 Introduction

MCN provides an data-dependent sub-region aggregation strategy by utilizing the local correlation. Intuitively, the Markov Clustering Network predicts the state of a node by collecting local information from all locations of an image. In other word, the state of a node depends on the state of the others. In the training phase, every node independently computes the cross-entropy loss of the predicted flow probability distribution and propagates the error signal spatially to all of the other. This supervised signal forces a neural network to consider the observations from all locations. However, the classic feed-forward neural network, e.g. CNN, only capturing the context within its receptive field, may not be optimal architecture for the Markov Clustering framework.

In this chapter, a 2-dimensional Position-aware Attention (2D-PAA) mechanism is proposed to enhance the representation capacity of a neural network to support our Markov Clustering framework. The proposed method is compared with the recently

proposed Spatial Recurrent Network [24]. Similar to Spatial Recurrent Network [24], 2D-PAA considers input representations as a 2D sequence. Instead of capturing relative positional information along spatial dimension directly through the sequential structure of Recurrent Neural Networks, 2D-PAA applies positional encoding to make the entire model be aware of the sequence order. 2D-PAA explicitly models correlation between two spatial representations and thereby extends the receptive field of the CNN model by adaptively fusing spatial features in different part of an image. Compared with the spatial recurrent architecture, 2D-PAA is much more easy to parallelize with plenty of contextual information. However, the training cost of 2D-PAA increases rapidly with the number of nodes. To address this issue, an efficient forward/backward strategy is proposed to greatly reduce the computation complexity of 2D-PAA training.

The 2D-PAA is integrated in the Markov Clustering Network and the evaluation on public benchmarks validates the effectiveness of proposed 2D-PAA mechanism. Moreover, the proposed efficient training strategy brings 62% memory reduction and 49% computation reduction. The contribution of this work is summarized as follows:

- A 2D Position-aware Attention mechanism (2D-PAA) is proposed to further enhance the capacity of MCN model
- A randomized training techniques is developed for 2D-PAA layer to reduce the computational complexity during training.
- Our method achieves the state-of-art performance on popular ICDAR 2013 and ICDAR 2015 datasets, and outperforms existing scene text detection method in detecting arbitrarily oriented text objects with new state-of-art performance (P: 89.2, R: 80.6 and F: 85.2) on challenging MSRA-TD500 dataset.

4.2 Two-dimensional Position-aware Attention

4.2.1 2D Self-attention with Position Encoding

It has been demonstrated in [70] that the spatial recurrent structure can greatly improve the performance in detecting horizontal text lines. The idea behind is that text have strong sequential characteristics which can help to reduce false detections caused by ambiguous patterns and recover missed text proposals with weak text information. The RNNs essentially provide a mechanism to selectively fuse spatial representations and thus extend the receptive field. However, the RNNs still have some drawbacks. Firstly, the RNNs are prone to over-fit when the training dataset is small [105]. Secondly, RNNs have poor parallelism due to its temporal dependent structure. Thirdly, RNNs can only encode context along one direction and may not be optimal for orientation-varying scene text objects. In contrast, non-recurrent models, such as attention [77] and self-attention [106] models, has much higher level of parallelism but lack capability to capture sequential characteristics of texts, but they process context information without necessarily considering input elements sequentially. Our solution is applying explicit encoding of position information to capture sequential characteristics of texts.

The 2D Position-aware Attention (2D-PAA) is an extension of self-attention models [107], which considers the pair-wise position relation between spatial features extracted by the preceding CNN. A set of predefined encodings α^{vert} and α^{horz} are used to combine with input spatial features to expose relative position information to the model. As shown in Fig. 4.1, it is applied at the output of CNN with resolution of $\frac{H}{U} \times \frac{W}{U}$. The CNN representation $\mathcal{X} \in \mathbb{R}^{\frac{H}{U} \times \frac{W}{U} \times C_{\mathcal{X}}}$ is treated as a 2-dimensional sequence by 2D-PAA layer, on which $\frac{H}{U} \times \frac{W}{U}$ individually operate and compute a new sequence $\mathcal{Z} \in \mathbb{R}^{\frac{H}{U} \times \frac{W}{U} \times C_{\mathcal{Z}}}$. For

convenience, we use the reshaped version of $\mathcal{X}$ and $\mathcal{Z}$:

$$\mathcal{X}' = \text{Reshape}(\mathcal{X}, [\frac{H}{U} \cdot \frac{W}{U}, C_{\mathcal{X}}]), \quad (4.1)$$

$$\mathcal{Z}' = \text{Reshape}(\mathcal{Z}, [\frac{H}{U} \cdot \frac{W}{U}, C_{\mathcal{Z}}]). \quad (4.2)$$

The output $\mathcal{Z}' \in \mathbb{R}^{\frac{H}{U} \cdot \frac{W}{U} \times C_{\mathcal{Z}}}$ as the weighted sum of the linear transformation of input $\mathcal{X}' \in \mathbb{R}^{\frac{H}{U} \cdot \frac{W}{U} \times C_{\mathcal{X}}}$. The i -th row of $\mathcal{Z}'$ is represented by:

$$z_i = \sum_j e_{ij} \cdot (x_j \cdot W_3 + \alpha_{ij}^{horz} + \alpha_{ij}^{vert}) \quad (4.3)$$

where x_i and x_j are the i and j rows of $\mathcal{X}$, representing corresponding spatial features indexed by 1-D indexes; α_{ij}^{horz} and α_{ij}^{vert} belong to matrices $\alpha^{horz} \in \mathbb{R}^{(2N_h+1) \times C_{\mathcal{Z}}}$ and $\alpha^{vert} \in \mathbb{R}^{(2N_v+1) \times C_{\mathcal{Z}}}$ that encode $(2N_h+1)$ horizontal offsets and $(2N_v+1)$ vertical offsets; $W_3 \in \mathbb{R}^{C_{\mathcal{X}} \times C_{\mathcal{Z}}}$ is the weighted matrix of linear transformation and e_{ij} is an element of weighted sum matrix $\mathcal{E} \in \mathbb{R}^{\frac{H}{U} \cdot \frac{W}{U} \times \frac{H}{U} \cdot \frac{W}{U}}$. Specifically, e_{ij} can be computed by:

$$e_{ij} = \frac{x_i \cdot W_1 \cdot (x_j \cdot W_2 + \alpha_{ij}^{horz} + \alpha_{ij}^{vert})^T}{\sqrt{C_{\mathcal{Z}}}}. \quad (4.4)$$

where W_1 and W_2 represent the transformation matrices for two different forward paths of input $\mathcal{X}$.

4.2.2 Efficient Forward and Backward Pass

In this subsection, we discuss how to efficiently perform training of 2D-PAA. In the forward pass, the most intensive computation would be computing the weighted sum matrix $\mathcal{E}$ with (i, j) -th element represented by Eq. 4.4. Since the offset representation α_{ij}^{horz} and α_{ij}^{vert} differ with different position pairs, one may fail to, in the first glance, reformulate Eq. 4.4 into a matrix multiplication which can be perfectly parallelizable on

Figure 4.1: 2-Dimensional Position-aware Attention.

GPUs.

$$e_{ij} = \frac{1}{\sqrt{C_Z}} x_i W_1 W_2^T x_j^T + \frac{1}{\sqrt{C_Z}} x_i W_1 (\alpha_{ij}^{horz} + \alpha_{ij}^{vert})^T, \quad (4.5)$$

$$= \frac{1}{\sqrt{C_Z}} x_i W_1 W_2^T x_j^T + (\beta_{ij}^{horz} + \beta_{ij}^{vert}). \quad (4.6)$$

In fact, as proposed in [107], Eq. 4.4 can be rewritten in to Eq. 4.5 and then Eq. 4.6, where the first term can be regarded as the (i, j) -th element in matrix $\frac{1}{C_Z} \mathcal{X} W_1 W_2^T \mathcal{X}^T$, and the second and third terms are representing the (i, j) -th element of matrix β^{horz} and β^{vert} illustrated in Eq. 4.7 and Eq. 4.8. Therefore, $\mathcal{E}$ is consisted of three separate matrix multiplication, which can be computed using off-the-shell parallel machine platforms. Moreover, since the offset encodings α_{ij}^{horz} and α_{ij}^{vert} are shared by every pair of position, one can firstly pre-compute all possible output of $(\beta^{horz}$ and $\beta^{vert})$ and then select the according columns as β_{ij}^{horz} and β_{ij}^{vert} in both training and testing phase. The space complexity for this approach is $\mathcal{O}(\frac{H^2 W^2}{U^4} + (2N_{(\cdot)} + 1)C_Z)$.

$$\beta^{horz} = \mathcal{X} \cdot W_1 \cdot \alpha_{horz} \quad (4.7)$$

$$\beta^{vert} = \mathcal{X} \cdot W_1 \cdot \alpha_{vert} \quad (4.8)$$

As the number of offset encoding $N_{(.)}$ is much smaller than the number of spatial representations $\frac{HW}{U^2}$, it will result a great amount of memory reduction.

As for backward pass, directly computing the gradient respective to input $\mathcal{X}$ also has the space complexity of $\mathcal{O}(\frac{H^2W^2}{U^4}C_{\mathcal{Z}})$. To reduce the computation in training, we randomly sample a portion of all position pairs in training phase when computing z_i :

$$z_i = \sum_{j \in \mathcal{S}} e_{ij} \cdot (x_j \cdot W_3 + \alpha_{ij}^{horz} + \alpha_{ij}^{vert}), \quad (4.9)$$

where $\mathcal{S}$ denotes a subset randomly chosen from row index of $\mathcal{X}$ with sampling factor of γ . In the backward pass, only the gradients respective to the input elements contributing output z_i will be computed, leading a computation reduction of space complexity for gradients from $\mathcal{O}(\frac{H^2W^2}{U^4}C_{\mathcal{Z}})$ to $\mathcal{O}(\gamma \frac{H^2W^2}{U^4}C_{\mathcal{Z}})$. As is shown in experiment, the performance can be maintained even if γ is set to 0.4. In addition to memory reduction, random sub-sampling can be seen as a kind of regularization that forces 2D-PAA layer to utilize the entire input sequence to compute the output by dropping out several input signals.

4.3 Detail Implementation

4.3.1 Simplified Network Architecture

The model presented in this chapter is similar to that adopted in the last chapter but with some modification for more elegant implementation and better performance. Similar to the architecture used in the last chapter, the new model consists of a backbone network implemented by tailored ResNet-50 [108], and Feature Pyramid Network (FPN), a fore/background branch and a flow prediction branch. Different from the original MCN with a FPN only contributing to the fore/background prediction, both fore/background branch and flow prediction branch share the same backbone and FPN. This practice brings a slight reduction of computation by reducing number of convolution layers. Further, we

Figure 4.2: Architecture of Feature Pyramid Network applied in MCN model.

find that cascade convolution layers are capable to directly predict the stochastic flows. Therefore, the flow mapping layer is removed in the new architecture to achieve more decent processing workflow.

Backbone and Feature Pyramid Networks

The design of cascaded network consisted of a backbone and a Feature Pyramid Network is similar to that in the last chapter. Differently, higher feature resolution is adopted. The $1/16 \times 1/16$ resolution is used in previous model, while three models with different resolutions ($1/16 \times 1/16$, $1/8 \times 1/8$ and $1/4 \times 1/4$) will be evaluated. The tailored ResNet-50 is applied as the backbone network and the architecture of FPN is illustrated in Fig. 4.2. The number of output channels increases from 128 to 1024. To compare the performance between 2D-RNN and 2D-PAA, these two modules are independently taking the representations as input provided by the same FPN.

2D-RNN

The 2-dimensional Recurrent Neural Network (2D-RNN) built on top of the FPN as the baseline model to be compared with proposed 2D-PAA. It is inspired by work [70] where 1D-RNN is individually applied to rows of an input representation. This architecture only encode an one-dimensional contextual information. We extend this architecture to encode two-dimensional context by stacking two 1D-RNN. As illustrated in Fig. 4.3, these two 1D-RNN layers are successively applied to the rows and columns of their inputs. Each layer is a Bi-directional Recurrent Neural Network (Bi-RNN) with 128 hidden nodes. The Bi-RNN composed of two RNNs encode the inputs in inversed direction independently, and the output sequence is constructed by summing the outputs of these two RNNs.

2D-PAA

The Two-dimension Position-aware Attention is built on top of the FPN to perform dynamic feature selection. The number of hidden nodes of 2D-PAA is set to be 256. The maximum offset encodings N_h and N_v are set to be 32, and we clip the relative position to the maximum offsets. Clipping the maximum distance is based on the hypothesis that relative position information will not be useful beyond a certain distance. Also, clipping enables the model to generalize to sequence lengths not seen during training. As a result, we have 65 vectors with dimension of 1×256 to encode the horizontal offset and another 65 vectors with the same dimension to encode the vertical offset.

4.3.2 Training

Objective

Similar to the training strategy illustrated in Chapter 3, the MCN model enhanced by 2D-PAA is trained by simultaneously minimizing the object cost C_o and the flow cost C_f

Figure 4.3: Architecture of 2D-RNN used for comparison with 2D-PAA.

defined in Section 3.2.4, which is represented by:

$$C = C_o + C_f. \quad (4.10)$$

Training Flow

The training flow includes two steps: first we train the our model on SynthText dataset [59] for 10 epochs, and then adopt the real data for fine-tuning. Following approach proposed in [5], we ignore the blurred text region by labeling them as “DO NOT CARE” during training.

Online Hard Negative Mining

As the negatives take up the most of the training samples, hard negative mining is applied to balance the positive and negative samples. We follow the online hard negative mining approach proposed in [109] and keep the ratio between the negatives and positives 5:1 at most. Hard negative mining is only performed for fore/background prediction.

Data Augmentation

Since the number of real data is limited, it is necessary to apply data augmentation for robustness of deep neural networks. First, we fix the short side of images to 600 pixels and resize the longer side according to the aspect ratio of the original image. For the images with aspect ratio less than 25 : 9, we use the original value to resize the images. Otherwise, the aspect ratio of resized image is fixed to 25 : 9. Next, image are rotated randomly in the range of $[-10^\circ, 10^\circ]$. Then, the heights of images are rescaled with ratio from 0.8 to 1.3 while their widths keep unchanged. Finally, 512×512 images are cropped from the transformed images.

4.3.3 Testing

We adopt single-scale and multi-scale testing to evaluate the performance. In single-scale testing, we fix the short side of an image to 600 and resize it according to its original aspect ratio. In the multi-scale testing, the short side length varies in the range of $\{600, 900, 1200\}$. Additionally, we clip the aspect ratio to 25:9 to avoid extremely long images.

4.4 Experiments

We evaluate the proposed model on three public scene text detection datasets, namely ICDAR 2013, ICDAR 2015, and MSRA-TD500.

4.4.1 Detail Analysis

Effectiveness of 2D-PAA

To evaluate the effectiveness of 2D-PAA in detecting shape-varying texts, we compare it with baseline MCN and its 2D-RNN version proposed in [110] on ICDAR 2013, ICDAR 2015 and MSRA-TD500 datasets. The performance of three models is demonstrated in Tab. 4.1. The baseline MCN achieves F-score of 87.2 on ICDAR 2013, 74.8 on ICDAR 2015 and 82.5 on MSRA-TD500. The 2D-RNN and 2D-PAA implementation outperform the baseline by up to 4.5% and 6.1% in terms of F-score. The precision is greatly improved by the 2D-RNN and 2D-PAA, indicating that the contextual information in spatial domain can effectively remove the false alarm. In the baseline MCN, the flow prediction could fail for an extremely long text object, leading to a box cleavage. The split boxes could be removed by NMS and the rest boxes cannot be fully matched with the ground-truth, resulting in low recall for this image. Both the 2D-RNN and 2D-PAA are capable of improve the flow prediction by including the long distance dependency between text regions, and thus alleviate the cleavage problem to improve recall. As shown

Table 4.1: Comparison between baseline MCN and MCN-2DPAA on ICDAR 2013 and MSRA-TD500.

Datasets	ICDAR 2013			ICDAR 2015			MSRA-TD500		
Methods	P	R	F	P	R	F	P	R	F
MCN	88.0	86.5	87.2	80.4	69.9	74.8	87.8	77.8	82.5
MCN-2DRNN	91.8	84.2	87.8	86.6	73.1	79.3	87.8	80.4	83.9
MCN-2DPAA	92.0	89.6	90.7	88.5	74.5	80.9	88.9	80.6	84.5

in Tab. 4.1, the 2D-PAA outperforms the 2D-RNN by 3% in F-score on ICDAR 2013. In this experiment, we keep the number of model parameters roughly the same. Thus, we believe that the performance gain comes from the better feature selection strategy provided by 2D-PAA.

Impact of γ on 2D-PAA

The training of 2D-PAA layer is computationally intensive, since it needs to evaluate the gradients between any two spatial representations. It takes up large amount of GPU cycles and memories. As illustrated in Sect. 4.2, the computation resources can be saved by randomly sampling a portion of spatial representations for gradient computation, while maintaining the prediction performance. In this experiment, we experimentally analyze the impact of sampling factor γ on training of 2D-PAA in terms of detection performance, memory footprint and computation time.

The achieved precision, recall and F-score with different values of γ is profiled on Fig. 4.4.1 (a). The detection performance increases rapidly with γ in the range from 0.1 to 0.4 and saturates from 0.3. As for the memory footprint and computing time, both increase linearly with the γ . For a training image resized to 512×512 , the space complexity are 283.7 MB and 556.3 MB for forward and backward pass respectively with corresponding computing time of 11.3 ms and 26.5 ms when γ is set to 1. In the optimal configuration with γ set to 0.4, 2D-PAA only consumes 112 MB for forward pass and 222 MB for backward pass. Also, the computing time drops greatly from 11.3 ms to 5.8 ms and

from 26.5 ms to 17.3 ms for forward and backward respectively. As result, the random sub-sampling brings around 62% and 49% reduction in memory usage and computation without harm of detection accuracy.

Impact of Prediction Density

The prediction density is the distance between adjacent predictions. In our work, it refers to the spatial distance between neighborhood nodes in a stochastic flow graph. Higher prediction density (smaller distance accordingly) results in better performance in detecting small text objects but consumes more run-time. The impact of prediction density on detection performance and inference speed is analyzed on the ICDAR 2013 and ICDAR 2015 datasets. We evaluate the baseline MCN and MCN-2DPAA models with 3 down-sampling factors (1/16, 1/8 and 1/4) in terms of accuracy and average runtime. All the testing images are resized to images fixed short-side of 600.

As shown in Tab. 4.2, the frame rate for both dataset decreases with the prediction density (from 35.8 fps to 17.8 fps on ICDAR 2013 and from 22.4 fps to 12.3 fps on ICDAR 2015). With the additional 2D-PAA layer, the MCN-2DPAA runs slower than the baseline MCN but enjoy higher accuracy on both datasets (up to 4.3% f-score improvement on ICDAR 2013 and 6.1% improvement on ICDAR 2015). As illustrated in Fig. 4.2, increasing the prediction density brings different improvement on the ICDAR 2013 and ICDAR 2015. It leads to a slight improvement on ICDAR 2013 (from 87.2 to 88.1 in f-score), since ICDAR 2013 is dominated by texts with middle and large size and improving the prediction density will not necessarily enhance the model capacity. While increasing the detection density leads to remarkable improvement on ICDAR 2015 (from 80.4 to 91.9 in precision and from 69.9 to 83.1 in recall). This is because ICDAR 2015 contains many small text objects, which requires more local details to enhance performance. In conclusion, increasing the prediction density is generally beneficial to the detection performance. But the performance gain depends on the statistics of the object size in a scenario, and one should trade off between the detection accuracy and the

Figure 4.4: The first/second row is the performance on ICDAR 2013/ICDAR 2015 with different γ to train 2D-PAA.

Figure 4.5: Memory usage and computation time of 2D-PAA. First and second rows correspond to forward and backward computation respectively.

Table 4.2: Performance and inference speed with different prediction density.

Datasets	ICDAR 2013				ICDAR 2015			
Methods	P	R	F	FPS	P	R	F	FPS
MCN-16	88.0	86.5	87.2	30.8	80.4	69.9	74.8	22.4
MCN-8	88.7	86.5	87.6	24.3	90.7	80.4	85.2	14.9
MCN-4	88.7	87.6	88.1	17.8	91.9	83.1	87.2	11.3
MCN-2DPAA-16	92.0	89.6	90.7	19.4	88.5	74.5	80.9	16.9
MCN-2DPAA-8	92.2	89.9	91.0	11.9	92.7	81.2	86.6	10.4
MCN-2DPAA-4	93.6	91.2	92.4	6.1	92.9	84.7	88.6	5.2

complexity properly to fit a specific application.

4.4.2 Comparison with Existing Methods

Focused Texts

We compare MCN with existing methods in processing focused texts on ICDAR 2013 dataset. Following methods proposed in [5, 10, 3], we report the performance of our method with $1/4 \times 1/4$ down-sampling factor. As shown in Tab. 4.3, the proposed MCN achieves the state-of-art performance. The baseline MCN achieves competitive performance with R^2 CNN [9]. But our method runs faster than R^2 CNN as the two-stage processing flow is slower than the single-stage one. The 2D-PAA version of MCN outperforms the existing methods with precision of 93.6, recall of 91.2 and f-score of 92.4, and the multi-scale testing technique also bring slight performance gain.

Incidental Texts

The performance of MCN method to detect incidental texts is evaluated on ICDAR 2015 dataset. The detection density is set to be stride of 4×4 . Since the ICDAR 2015 is dominated by small texts, our method cannot show much advantage over the existing methods. The baseline MCN with precision of 91.9, recall of 83.1 and F-score of 87.2 performs better than approaches [113, 3, 102, 9] by 4-6% and competitive with FOTS

Table 4.3: Localization performance on ICDAR 2013.

Methods	P	R	F
Jaderberg <i>et al.</i> [1]	89.0	68.0	77.1
TextFlow [95]	85.1	76.0	80.3
CTPN [70]	92.8	73.7	82.2
SegLink [12]	87.7	83.0	85.3
DeepReg [102]	92.0	81.0	86.2
R ² CNN [9]	93.6	82.6	87.7
RRPN [11]	90.2	71.9	80.0
MCLAB-FCN [61]	88.1	77.8	82.7
Jiang et al. [111]	92.2	91.5	91.9
MCN-4	88.7	87.6	88.1
MCN-2DPAA-4	93.6	91.2	92.4
MCN-2DPAA-MS-4	93.8	91.3	92.5

Table 4.4: Localization performance on ICDAR 2015.

Methods	P	R	F
CTPN [70]	74.2	51.6	61.9
SegLink [12]	73.1	76.8	75.0
SSTD [20]	80.2	73.9	76.9
WordSup [112]	80.0	77.0	78.4
DeepReg [102]	82.0	80.0	81.0
EAST [3]	83.3	78.3	80.7
R ² CNN [9]	85.0	80.0	82.4
RRPN [11]	84.0	77.0	80.3
CCFLAB-FTSN [113]	88.7	80.1	84.1
FOTS [5]	91.0	85.1	87.9
MCN-4	91.9	83.1	87.2
MCN-2DPAA-4	92.9	84.7	88.6
MCN-2DPAA-MS-4	93.5	85.0	89.0

Table 4.5: Localization performance on MSRA-TD500.

Methods	P	R	F
Zhang <i>et al.</i> [61]	83.0	67.0	74.1
Yao <i>et al.</i> [2]	77.0	75.0	76.0
SegLink [12]	86.0	70.0	77.2
DeepReg [102]	77.0	70.0	73.3
EAST [3]	87.3	67.4	76.1
RRD [10]	87.0	73.0	79.4
MOSTD [13]	87.6	76.2	81.5
MCN-4	88.5	80.3	84.2
MCN-2DPAA-4	89.2	80.6	85.2

[5]. Introducing 2D-PAA and multi-scale testing bring 1.4% and 0.4% improvement.

Multilingual

The performance of MCN for multilingual detection is evaluated on MSRA-TD500 datasets. As shown in Tab. 4.5, our baseline MCN achieves the state-of-art performance with F-score of 84.2 for baseline MCN and 85.2 for MCN-2DPAA. As shown in Fig. 4.6, the MCN can robustly detect very long text objects without predefined reference boxes. This feature eliminates the heuristic reference box design and allows more flexible text detection workflow for different scenarios. Moreover, the 2D-PAA brings around 1% improvement when compared with the baseline MCN. The adaptive selection provided by 2D-PAA allows the MCN to capture contextual information from larger area, which enhance the performance in detecting long objects.

4.5 Conclusion

In this chapter, we present a 2-dimensional Position-aware Attention (2D-PAA) mechanism to support the Markov Clustering framework. The proposed 2D-PAA can effectively improve performance by introducing explicit, including non-local information when making predictions at each spatial location. Compared with the Spatial Recurrent

Figure 4.6: Example results on ICDAR 2013, ICDAR 2015 and MSRA-TD 500 datasets. Text instances with multiple scales and orientations are detected. The predicted bounding boxes by our method are labeled in yellow and the ground-truth bounding boxes are labeled in red.

Networks using a 2-dimensional recurrent structure to convey information among nodes, the 2D-PAA achieves better performance and is more computationally efficient. Moreover, detailed profiling is provided to show that the proposed training method can effectively reduce cost in terms of computation and memory footprint. The experiments show that the proposed MCN method can effectively predict long text objects, and the 2D-PAA module brings prominent improvement. Also, our method achieves state-of-art performance when compared with the existing text detection methods with faster inference speed in the same prediction density.

Chapter 5

Curve Text Detection: Conditional Spatial Expansion

5.1 Introduction

In the previous chapters, the Markov Clustering Network and 2-dimensional Positionware Attention are presented for multi-oriented text detection. Since the methods proposed in the previous sections are assuming that the texts are in the rectangular shape, their performance will be limited when applying to the scenarios where texts have various shapes and layouts. The performance is dramatically degraded in these new scenarios, and a new method is demanded to tackle the problem of detecting texts with irregular shapes and layouts. To this end, in this chapter, we step further to explore more general scene text detection methods to retrieve arbitrary text.

The main challenges of arbitrary (or curve) text detection come from irregular shapes and highly varying layout. The traditional bounding box representation does not scale well in the curve scenario since one bounding box may cover multiple text objects. Therefore, the recently proposed curve text detection algorithms [21, 114, 115] follow a two-stage detect-and-refine approach to generate elaborated polygons or boundaries. In these methods, a CNN based text detector is applied to locate the regions containing texts,

and then a segmentation or polygon regression algorithm is performed on these regions to produce a tight polygon or boundary. Both methods highly depend on the accurate region proposal provided by the text detector. They prefer a proposed region with only one targeted object included which reduces the ambiguity of the sampled features. Although an oriented box regression is preferable, it often fails in the curve text scenario [11, 9]. The recently proposed curve text detection methods turn to predict horizontal rectangles in the first stage, which is inevitably affected non-targeted texts in the sampled regions.

Specifically, both segmentation and regression-based methods can be disrupted by the unexpected texts included in the same box region. The segmentation based methods can fail to distinguish the targeted text from the others and misclassify the unexpected texts or text-like patterns as positive, shown in Fig. 5.1 (b). The regression-based methods can produce incorrect boundaries by indistinguishably considering all texts as one object. Moreover, as shown in Fig. 5.1 (e), the regression results (produced by predicting an offset of a proposed box region) are highly coupled with previous stage box proposals. A poor box proposal greatly affects the final polygon which degrades the performance.

To tackle the problem mentioned above, a novel Conditional Spatial Expansion (CSE) mechanism is proposed, which acts as a second-stage component applied in the widely adopted two-stage detection workflow. Our method is derived from conditional modeling of dependency between an interior point (called a seed) and the rest parts of a text instance. The CSE can be seen as a conditional prediction process that retrieves an instance-level text region by seeding and expanding. Starting with an arbitrary interior point (seed) of a text region, CSE selectively expands its territory based on location observation of the image patches and the context inferred from merged regions. Compared with the segmentation based method, our CSE is extremely discriminative, especially when texts are close to each other as demonstrated in Fig. 5.1 (c). It provides a controllable approach to extract an expected text region with minimum efforts of post-processing. On the other hand, our CSE is highly flexible since a seed can be specified at any location within the targeted text region. Compared with the polygon regression methods, the

seeding-and-expanding paradigm has less coupling with the previous text detector. Based on a coarse region proposal, our CSE is directly applied to the spatial features produced by the backbone, which preserves all the spatial information and will not be affected by the imperfect region proposals, shown in Fig. 5.1 (f). The experiments show that our method outperforms the existing curve text detection methods on public benchmarks. The contributions of this work are summarized as follows:

- The curve text detection is formulated as a conditional region expansion problem, which initializes a seed within a text region and then progressively retrieves targeted object by region expansion;
- The spatial dependency between the seed and the rest part of an object is modeled by a parameterized Conditional Spatial Expansion mechanism, which allows us to selectively extract a text region indicated by a seed with high area precision;
- Our CSE acts as a second-stage text extractor which can be seamlessly integrated into existing object detection workflows;
- The arbitrariness of seed’s location and high spatial selectivity of our method reduces coupling with the previous detector and thus provide flexible and robust boundary prediction;
- Our method outperforms the existing curve text detection methods on public curve text datasets with F-measurement of 80.2% on Total-Text [91] and 78.4% on CTW-1500 [21].

5.2 Method

5.2.1 Overview

Our method retrieves an instance-level text region by seeding and then expanding. A seed, as shown in Fig. 5.2, uniquely indicates an object and can be arbitrarily initialized at the

Figure 5.1: Problems of existing curve text detection methods: Two region proposals shown in (a) and (d) inevitably include unexpected texts since they are closed to each other. Thus, it causes failures for Mask RCNN based methods and polygon regression methods demonstrated in (b) and (e). Our CSE method demonstrated in (c) and (f) shows strong robustness to this situation and brings significant performance gain.

interior of an object region. Seeding is to select a location within an object from which the corresponding object region is extracted by expanding. As will be illustrated in Sect. 5.2.4, a seed is initialized by an object detector with a predicted box center. Starting with a seed, the expansion is conducted by selectively merging adjacent sub-regions to form a targeted object region. As shown in Fig. 5.2, the sub-regions are abstracted as feature points or nodes, which are sampled from the input image at discrete locations. They are organized as a grid and locally assigned an expanding indicator $y \in \mathbb{R}^5$ to represent the merging direction to neighborhood nodes. Five entries of y denote the probabilities of all possible merging directions (*to – bottom*, *to – right*, *to – left*, *to – top* and *none*). A node will be merged into an existing object region indicated by a seed if its major merging direction is pointing to its neighborhoods that already belong to the object region. An instance-level object boundary can be easily produced by mapping all the positive nodes back to the original image and extracting the contour.

5.2.2 Modeling

The seeding-and-expanding paradigm provides a flexible and controllable way to extract object regions with minimal post-processing. It also reduces the performance coupling with the first-stage detector by allowing arbitrary initialization of a seed node. However, with different seed locations, the dynamics of the region expansion are different. For a specific node, the state of expanding indicator varies with the seed's location and also depends on the expanding indicators of its neighborhoods. Therefore, obtaining expanding indicators should be regarded as a conditional prediction problem.

Fig. 5.2 demonstrates the modeling of conditional prediction problem. A seed node is defined as an arbitrarily chosen inner point of a text object, which is depicted as a black point in Fig. 5.2. Centered at the seed node, a grid is conceptually expanded to cover the whole text object. The rest node in the grid belonging to the same text object are labeled in yellow and the nodes outside the text object are labeled in white. We further define the number of steps from a node to the seed as the minimum number of movements from a node to the seed. Then, we divide the grid into sections using contours, where nodes in the same section have the same number of steps to a seed. We assume that the expanding indicators of nodes P_k within the k -th section (from inside out) are mutually independent and their states depend only on the current spatial feature X_k and the states of nodes within previous sections $P_{k-1}, P_{k-2}, \dots, P_0$. The optimal estimator of Y_k can be represented by

$$\hat{Y}_k = \underset{Y_k}{\operatorname{argmax}} \Pr(Y_k | Y_{k-1}, \dots, Y_0; X_k, \dots, X_0), \quad (5.1)$$

which maximizes the posterior probability of Y_k when observing spatial features $X_{(\cdot)} := \{x(p) | p \in P_{(\cdot)}\}$ and indicators of previous nodes $Y_{(\cdot)} := \{y(p) | p \in P_{(\cdot)}\}$. This conditional modeling allows the region expansion to be adaptable with an arbitrarily initialized seed location. Also, it effectively differentiates the expected object from the others by considering the context derived from a seed. Moreover, the independence assumption

Figure 5.2: Our CSE inside-out explores the expanding indicator of every node and merges nodes with merging direction pointing to nodes that are already in the object region.

among nodes in the same section results in a dendritic Conditional Spatial Expansion process with high-level parallelism.

5.2.3 Conditional Spatial Expansion

To estimate conditional probability illustrated in Eq. 5.1, we develop a highly parameterized Condition Spatial Expansion (CSE) mechanism. Given a seed inside an object region, we construct its neighborhood feature points by expanding a $S \times S$ grid and then sampling the features produced by the backbone network using bilinear interpolation, as shown in Fig. 5.3 (a). Starting with the seed node, our CSE explores every single node inside-out and computes corresponding y and the transition vectors $h_o = [h_o^b, h_o^r, h_o^l, h_o^t]^T$ to its neighborhoods based on the current sampled feature $x \in \mathbb{R}^{d_x}$, local state $c \in \mathbb{R}^d$ and transition vectors $h_i^b, h_i^r, h_i^l, h_i^t \in \mathbb{R}^d$ coming from the adjacent feature points. The transition vectors encode the position-sensitive information, which helps the CSE to be aware of the relative location of the current node to the seed. Depending on the relative position to the seed, the inputs and outputs for the nodes are illustrated in Fig. 5.3 (c). For

Figure 5.3: Given a seed located, depicted as a blue cross within the red region in (a), at the interior of a text region, we expand a grid with $S \times S$ points and sample the feature produced by backbone at these locations using bi-linear interpolation, shown in (a). As demonstrated in (b), the CSE computation starts with the seed and spreads to the adjacent feature nodes. The spreading directions are indicated by the arrows in (b). Each node takes the outputs and the hidden states of previous nodes as input and produces new hidden state and expanding indicator predictions, as illustrated in (c). Finally, the nodes associated with the same seed, depicted as the yellow region in (b), are extracted according to the expanding indicators.

Figure 5.4: Computing Graph inside a Node.

a node in P_k , our CSE only takes the c and h as inputs from P_{k-1} and output new h to P_{k+1} . This constructs an inference process that originated from the seed, which propagates the contextual information among the grid in a dendritic manner. The computation of the nodes in the same section is independent and thus can be fully parallelized on GPUs. The computation complexity is linear to the size of the grid, which is computationally efficient.

Inside a specific node, the computation is illustrated by a computing graph shown in Fig 5.4. Without loss of generality, we denote all possible inputs from neighborhood nodes by $c_i \in \mathbb{R}^{4d}$, $h_i \in \mathbb{R}^{5d}$, and $y_i \in \mathbb{R}^{20}$, which are represented by

$$c_i = [c_i^b, c_i^r, c_i^l, c_i^t]^T, \quad (5.2)$$

$$h_i = [h_i^c, h_i^b, h_i^r, h_i^l, h_i^t]^T, \quad (5.3)$$

$$y_i = [y_i^b, y_i^r, y_i^l, y_i^t]^T, \quad (5.4)$$

where $c_i^{(\cdot)} \in \mathbb{R}^d$, $h_i^{(\cdot)} \in \mathbb{R}^d$ and $y_i^{(\cdot)} \in \mathbb{R}^5$ denotes the local states, transition vectors and the expanding indicators of neighborhood nodes ¹. Here, only parts of c_i , h_i and y_i are available according to the relative position to the seed node ², and the rest will be set to zeros. Particularly, h_i^c is defined to uniquely indicate the seed node. Other than h_i^c of a seed which is learned by backpropagation, we explicitly set the h_i^c of other node to zeros.

From the current observed feature x , as shown in Fig. 5.4, transition vectors h_i and the predicted expanding indicators of neighborhoods y_i , we compute a candidate local state $\tilde{c}$ by

$$\tilde{c} = \tanh(W_c \times s + b_c), \quad (5.5)$$

$$s = [x, y_i, h_i]^T, \quad (5.6)$$

¹The superscripts indicate the relative position to the current node, b-bottom, r-right, l-left, t-top and c-current.

²The non-zero inputs are indicated in Fig. 5.3 (c)

where “ $\times$ ” represents the matrix multiplication, $W_c \in \mathbb{R}^{d \times (d_x + 5d + 20)}$ and $b_c \in \mathbb{R}^d$ denote weights and bias of linear transform before a tanh activation. We apply the gating mechanism [116] to combine the local state c_b , c_r , c_l and c_t from the neighborhoods with the current candidate state $\tilde{c}$ to obtain the local state of current node c , which is formulated as

$$c = \delta(c_b \cdot g_c^b + c_r \cdot g_c^r + c_l \cdot g_c^l + c_t \cdot g_c^t + \tilde{c} \cdot g_{\tilde{c}}), \quad (5.7)$$

where δ denotes the layer normalization operator [117], “ $\cdot$ ” is the element-wise multiplication depicted as “ $\cdot$ ” nodes in Fig. 5.4, and g_b , g_r , g_l , g_t , g_c represent the outputs of gating function which can be further illustrated by

$$g_c = [g_{c_i}^b, g_{c_i}^r, g_{c_i}^l, g_{c_i}^t]^T = \sigma(W_{g_{c_i}} \times s + b_{g_{c_i}}), \quad (5.8)$$

$$g_{\tilde{c}} = \sigma(W_{g_{\tilde{c}}} \times s + b_{g_{\tilde{c}}}). \quad (5.9)$$

Here, $W_{g_{c_i}}$, $b_{g_{c_i}}$ and $W_{g_{\tilde{c}}}$, $b_{g_{\tilde{c}}}$ are defined as the weight matrix and bias to map s into corresponding gating vectors g_{c_i} and $g_{\tilde{c}}$. Since the local state c is essentially the weighted sum of state values of the previous nodes, the values of c increases exponentially with k in our two-dimensional scenario. This significantly harms the numerical stability in both training and testing phases. Thus, the layer normalization technique, depicted in green node in Fig. 5.4, is essential for the CSE to ensure the convergence of training and prevent overflow in testing. Finally, the expanding indicator y and the output transition vectors h_o are derived from the local state c , which are illustrated as follows

$$g_o = \sigma(W_{g_o} \times s + b_{g_o}), \quad (5.10)$$

$$h_o = [h_o^b, h_o^r, h_o^l, h_o^t]^T = \tanh(c) \cdot g_o + b_o, \quad (5.11)$$

$$y = \text{softmax}(W_y \times c + b_y), \quad (5.12)$$

where W_{g_o} and b_{g_o} represent the weight matrix and bias used to produce the corresponding gating signal, and W_y and b_y transform c into logits before feeding to *softmax* activation.

5.2.4 Seeding

In our CSE method, a seed is assumed to be located within an object region. This prerequisite can be easily guaranteed by the first-stage seed initializer implemented by an off-the-self object detection. Here, we adopt detected box centers and shapes by Faster RCNN [17] to decide seed locations and the shapes, and uniformly sample $S \times S$ features using bi-linear interpolation from a region indicated by a bounding box. In fact, our CSE method only requires a seed to be located within an object region with sampling grid coarsely covering the targeted object. Moreover, given a sampling grid, any node within the object region can be specified as a seed. As shown in Sect. 5.4.3, randomly initializing seed location and corresponding grid size does not significantly affect the performance. Therefore, a weaker detector, which is easy to optimize, could be sufficient for CSE to produce satisfactory results.

5.2.5 Optimization

Labeling In the training phase, the ground-truth merging directions are labeled using the strategy illustrated in Fig. 5.5. For each grid in CSE, we first label the nodes within the target ground-truth object region as positive (nodes in yellow) and the rest as negative (nodes in gray). For every positive node in section k , we search its neighborhood positive nodes in section $k - 1$ and label the corresponding merging directions as candidate directions. The candidate directions for a specific positive node are indicated with arrows as shown in Fig. 5.5. One positive node has at most two candidate directions. Among the candidate directions, we only label the one direction with the highest score predicted by neural networks as the final positive merging direction labeled in a solid arrow and the other direction labeled in dotted arrow is not considered in the loss propagation. For the

Figure 5.5: Ground-truth Labeling: All arrows represent the candidate merging directions for nodes, and the corresponding scores predicted by CSE are marked. The candidate directions with highest scores are label as positive directions, which are represented by solid arrows.

seed node, we always label its y_s as positive.

Loss Function We apply the cross-entropy loss to each node to optimize our CSE model, which can be represented by

$$\mathcal{L}_{cse} = \frac{1}{N} \sum_{p \in P} -\ln(y^*(p)), \quad (5.13)$$

where $N = S \times S$ represents the number of nodes in a grid, P denotes a set of all nodes, and y^* is the value of the positive merging direction. Our optimization strategy computes the loss according to the current CSE prediction. Intuitively, it intends to boost the positive candidates which are already strong, which reduces the ambiguity in labeling and speed-up the convergence.

5.3 Implementation Details

5.3.1 System Architecture

Backbone

The system architecture is illustrated in Fig. 5.6. It consists of a backbone network, a faster RCNN and our CSE module. The backbone network is used to encode an image into spatial features, which is composed of a ResNet-34 [118] and a Feature Pyramid Network (FPN) [119]. For the ResNet-34, we remove the fully-connected layers and keep the rest with the output feature dimension of 512 and a down-sampling factor of $\mathcal{Q} = 1/16$. Following the ResNet-34, we apply an FPN with additional 5 down-sampling layers and N_u up-sampling layers. The number of output channels of each down-sampling or up-sampling layer is 512. Notably, each up-sampling layer up-samples an input feature map with a factor of 2, and N_u varies with the targeted benchmarks where the evaluation is conducted. For the curve text benchmarks [91, 21], N_u is set to 5 to produce an output feature map with down-sampling factor of $1/16$. For the non-curve text benchmarks [88], the optimal down-sampling is heuristically set to $1/4$ following the existing approaches [114, 120, 21, 11, 3], and thus two additional up-sampling layers are applied.

Faster RCNN for Seeding

The Faster RCNN [17] is built on top of the backbone to initialize the seeds' locations and corresponding grid sizes. The features computed by the backbone network are first fed to the Region Proposal Network [17] to generate coarse region proposals. In the RPN, 20 anchors with 5 scales (32, 64, 128, 256, 512) and 4 aspect ratios (0.25, 0.5, 1, 2) are defined. In RoI generation, we first select 6k anchors with the highest scores before NMS and output 300 RoIs after that. Here, the IoU threshold is set to be 0.7. The proposed RoIs are subsequently input to the RCNN [25] for fore/background classification and bounding box calibration. In this stage, we suppress the overlapped bounding boxes

using NMS with IoU threshold of 0.5. The reason for setting a high IoU threshold is that the bounding boxes are highly overlapping if their corresponding text object is close to each other. Setting a low IoU threshold, e.g. 0.3, will incorrectly remove these objects. After the second NMS, we remove the box proposals with positive scores lower than 0.7 and feed the resulted box proposals to the CSE module.

CSE

Our CSE takes the output boxes by Faster RCNN to generate a set of $S \times S$ sampling grids. These grids represent the sampling locations on the original images, which will be used to compute the values of sampled features from the output of the backbone based on bilinear interpolation method. Here, S is set to 25. Subsequently, the CSE is applied to the sampled feature points to extract the text region indicated by a seed. Finally, the instance-level boundaries are produced by mapping associated feature points back to the input image and then extracted corresponding contours.

5.3.2 Training

Faster RCNN

The training of Faster RCNN follows the standard training strategy proposed in [17, 109]. The short sides of the input images are fixed to 512 pixels and the aspect ratios are configured to be lower than 1.5. Color distortion and rotation are applied to augment the training images. Since the size of texts' ground-truth boxes is highly varying, the original threshold 0.7 easily leads to a batch overwhelmed by negative samples, which hinders the convergence. We maintain the batch size to be 256 but reduce the positive matching threshold of RPN to 0.6 to maintain a sufficient number of positive anchors. As for the training of RCNN, the batch size is set to 128 and the positive RoI threshold is set to 0.5. Additionally, we apply the Online Hard Example Mining (OHEM) [109] to balance the number of positives and negatives to 1:3.

Figure 5.6: System Architecture.

CSE

A batch to train CSE is configured to have 10 samples with ground-true seeds and grids equally sampled from the positive RoI produced by previous Faster RCNN and the augmentations of the ground-true text boxes. The augmented seeds and grids are constructed by manipulating the grid sizes and the seeds' locations under the constraints that an augmented grid should have at least 0.4 overlapping with the ground-true boxes and a seed should be within the targeted object region. After the RoI augmentation, we compute a ground-true foreground mask for each RoI based on the ground-true polygons provided in the datasets. The ground-true foreground mask will be used to compute the CSE loss for optimization.

5.4 Experiment

5.4.1 Datasets

The datasets used for the experiment are briefly introduced below:

ICDAR-2017 MTL [89] is a large multi-lingual text dataset. It contains 7200 training images and 1800 testing images with text instances from 9 languages. The text regions are constrained to rectangles but can be arbitrarily rotated. In our experiment, all the 9000 images are included to pretrain our model.

MSRA-TD500 [29] is a small dataset composed of multi-lingual, arbitrarily oriented and long text lines. It includes 300 training images and 200 testing images with line-level annotation. The training set is included for pretraining and fine-tuning when evaluating the performance of this dataset.

Total-Text [91] is a newly released benchmark for curve text detection. In addition to rectangle text instances, this dataset especially includes curve text instances. The curve text instances are labeled with polygons with the various number of vertices. We re-sample the vertices and generate a fixed quadrilateral for each text instance as the

ground-truth polygon. The training sets are included in pretraining and also used for fine-tuning when evaluating the performance on this dataset.

CTW-1500 [21] is another curve text dataset released recently. It contains 1500 images in total with 1000 images for training and 500 images for testing. We include the training set to pretrain our method. Before the evaluation on this dataset, we fine-tune the model by only using the training set.

5.4.2 Experiment Details

The experiment is conducted on Tensorflow 1.5.0 [101]. We adopt Faster RCNN driven by ResNet-34 to initialize seed locations and corresponding grids in all experiments. The CSE is implemented and optimized in C++ and accelerated by CUDA. Following the existing training strategies for scene text detection [3, 114, 5], we pretrain our model on a combined dataset. The pretraining dataset consists of over 10k images from full set of ICDAR-17 MLT [89] and the training sets of MSRA-TD500 [29], Total-Text [91] and CTW-1500 [21]. After the pretraining, we fine-tune and evaluate our method on two curve text datasets Total-Text (with 1255 training images and 300 testing images) and CTW-1500 (with 1000 training images and 500 testing images). The model is trained on the combined dataset for 50k iterations and fine-tuned on the datasets to be evaluated. We adopt the Adam optimizer [121] to train the network. In the pretraining phase, the learning rate is fixed to 0.01 for the first 30k iterations and scaled down to 0.002 for the rest iterations. In the fine-tuning, the initial learning rate is set to 0.001 and decays exponentially 0.9 every 5000 iterations. All the experiment is conducted on Dell Alienware with Intel i7 processor, 64GB memory and two NVIDIA GTX 1080 Ti GPUs. The batch size is set to 1 for each of two GPUs in training and only one GPU is used for evaluation.

Figure 5.7: Robustness and Flexibility Analysis: In the first row of each case, the sampling regions are represented by the bounding boxes in yellow and the seed locations are labeled by the cross dash lines. The second row shows the zoom-in of RoIs. The corresponding heatmaps of associate regions are shown in the third row. As shown in (a), our method is very flexible in the seed’s locations. (b) and (c) prove the robustness of our method to extract text from a density text region.

5.4.3 Flexibility and Robustness

In this experiment, we validate the flexibility and robustness of our CSE method qualitatively and quantitatively. In the qualitative experiment, we generate a set of sampling grids (in yellow) with different locations and sizes by randomly manipulating the ground-truth boxes, as shown in the first row of Fig. 5.7. We apply the CSE to the corresponding RoIs shown in the second row of Fig. 5.7, and visualize the extracted text regions by heat maps in the third row of Fig. 5.7. Fig. 5.7 (a) shows the flexibility of our method. Our CSE method can effectively retrieve the text region with different seed’s locations. Even for a text object with large curvature and slim shape, our method can capture all the related sub-regions with high area precision. On the other hand, for a proposed region with many unexpected texts included or even dominated by another text instances (demonstrated in Fig. 5.7 (b) and (c)), our method only extracts associated object regions indicated by the seed. It indicates that our CSE is robust to the ambiguity caused by unexpected objects and can produce satisfactory results even for a poor sampling grid generated by the previous object detector.

In addition to visually investigating our CSE, we quantitatively verify its flexibility

and robustness by rescaling the size of a proposed sampling grid and relocating the seed in a grid. The grid rescaling resizes the height and width of a proposed region proposal by a factor of $\delta_s \geq 1.0$. The seed relocation is applied to a sampling grid to change its seed to a new node, which is still within the targeted object region but has δ_c deviation in Euclidean space from the original seed node. δ_c is normalized by the square root of the original grid area. We study the effects of rescaling and relocation separately by profiling the precisions, recalls, and F-scores on both Total-Text and CTW-1500, and the results are shown in Fig. 5.8. The performance variation respective to rescaling factor δ_s on two datasets is profiled in Fig. 5.8 (a) and (b). The performance is maintained at around 80% on Total-Text and 78% on CTW-1500 for δ_s ranging from 1.0 to 1.5. It slightly drops when δ_s is larger than 1.5 and the F-scores remain above 77% and 73%, respectively. As for seed relocation, the performance variations on Total-Text and CTW-1500 are illustrated in Fig. 5.8 (c) and (d). The performance is not greatly affected by the seed relocation. The F-scores remains 80% and 78% and decrease by only 3% and 4% when δ_c is changed from 0.0 to 1.0. In conclusion, our CSE is robust to randomly initialized seed locations and distorted sampling grids.

5.4.4 Comparing with Mask RCNN

The baseline Mask RCNN method is implemented based on the method proposed in [114]. We remove the text recognition branch and only keep the detection and segmentation branches. For a fair comparison, two methods share the same text detector which is based on Faster RCNN architecture. The quantity results are reported in Tab. 5.1. Our method is overall better than the baseline method. The F-scores of the baseline method on both datasets are 67.5% to 67.8%, respectively. In comparison, our CSE performs much better than the Mask RCNN based method by over 10% with F-score of 80.2% on Total-Text and 77.4% on CTW-1500.

To explore the cause of performance gain, we visually compare the output confidence

Figure 5.8: Performance v.s. rescaling factor δ_s on Total-Text (a) and CTW-1500 (b); Performance v.s. relocation factor δ_c on Total-Text (c) and CTW-1500 (d).

Datasets	Total-Text			CTW1500		
Model	P	R	F	P	R	F
MRCNN	69.2	65.8	67.5	65.1	70.8	67.8
CSE	81.4	79.1	80.2	78.7	76.1	77.4

Table 5.1: Performance Comparison between Mask RCNN based method and our CSE method.

Figure 5.9: (a) Examples output by baseline method; (b) Examples output by our CSE method. Compared with the baseline Mask RCNN method, our method shows impressing robustness to the ambiguity caused by adjacent texts and impressing selectivity to the targeted instance.

map produced by two methods on CTW-1500, which is shown in Fig. 5.9. Fig. 5.9 (a) demonstrates the failed examples produced by the baseline method. In these cases, the segmentation is distorted by the adjacent text instance. Parts of the unexpected text instances included in a box cause high activation in a confidence map and corrupt the boundary prediction. In contrast, our CSE is extremely robust in this scenario, since the contextual information captured by CSE helps to eliminate the ambiguity caused by unexpected objects. Moreover, the condition modeling allows our CSE to retrieve long curve text lines with high precision, which is flexible and promising in real applications.

Datasets	Total-Text			CTW1500		
Methods	P	R	F	P	R	F
Poly-Reg	73.8	77.4	75.6	77.1	69.7	73.2
CSE	80.9	80.3	80.6	79.2	76.0	77.6

Table 5.2: Performance Comparison between Polygon Regression based method and our CSE method.

5.4.5 Comparing with Polygon Regression

In this experiment, we compare our method with another baseline based on polygon regression proposed in [21]. The baseline is implemented based on the publicly available source code provided in [21], and is pretrained and fine-tuned with our training strategy. Similar to the previous experiment, the backbone and the RPN are shared among two methods, and the rest parts are implemented based on respective workflows. The results are shown in Tab 5.2. Our method outperforms the baseline method by 5% in terms of F-score on Total-Text, with the precision of 80.9%, recall of 80.3% and F-score of 80.6%. On CTW-1500, our CSE achieves F-score of 77.6 %, which is 4.4% better than the baseline.

We also investigate the causes of performance gain by visualizing the detection results of both methods. As demonstrated in Fig. 5.10, with the same RoI proposals shown in Fig. 5.10 (a), the polygon regression could be corrupted by the other text object which is occasionally included. As can be seen in Fig. 5.10 (b), the baseline model may consider all the texts included in a proposed region as a single object and regress the corresponding boundary. Although this problem can be mitigated by training a more accurate text detector to reduce the unexpected texts, the proposed bounding boxes inevitably cover additional texts due to text’s highly varying shapes and orientations. As shown in Fig. 5.10 (c), the conditional expansion mechanism only merges the sub-regions that are similar to the region indicated by the seed. By exploring the spatial dependency as well as the local information, our CSE method is much more robust than the polygon regression method and can produce more elaborated boundaries.

Figure 5.10: (a) The images with RoIs; (b) The text boundaries output by baseline method; (c) The text boundaries produced by our CSE method. The baseline method is easily affected by the unexpected texts included in the same boxes, while our method shows strong robustness to this situation.

Methods	P	R	F	time (s)
SegLink [12]	30.3	23.8	26.7	-
EAST [3]	50.0	36.2	42.0	-
Mask TextSpotter [114]	69.0	55.0	61.3	-
TextSnake et al. [120]	82.7	74.5	78.4	-
CSE	81.4	79.1	80.2	0.42

Table 5.3: Detection Performance on Total-Text.

Methods	P	R	F	time (s)
SegLink [12]	42.3	40.0	40.8	-
EAST [3]	78.7	49.1	60.4	-
DMPNet [6]	69.9	56.0	62.2	-
CTD [21]	74.3	65.2	69.5	-
CTD+TLOC [21]	77.4	69.8	73.4	-
TextSnake et al. [120]	67.9	85.3	75.6	-
CSE	81.1	76.0	78.4	0.38

Table 5.4: Detection Performance on CTW-1500.

5.4.6 Peer Comparison

We compare our method with the recently proposed methods for curve text detection on curve text benchmarks, Total-Text and CTW-1500. The results are shown in Tab.5.3 and Tab.5.4. Our method creates a new state-of-art performance with a precision of 81.4%, recall of 79.1% and F-score of 80.2% on Total-Text. On CTW-1500 containing both curve texts and long text lines, our method also achieves the state-of-art performance with the precision of 81.1%, recall of 76.0% and F-score of 78.4%. The inference time is 0.42 ms per image and 0.38 ms per image on Total-Text and CTW-1500, respectively. The detection results are demonstrated in Fig. 5.11. It shows that our method can effectively handle curve texts with irregular shapes, highly varying sizes and arbitrary orientations.

Figure 5.11: Detection results on Total-Text and CTW-1500.

Table 5.5: Localization performance on ICDAR 2013.

Methods	P	R	F
CTPN [70]	92.8	73.7	82.2
SegLink [12]	87.7	83.0	85.3
DeepReg [102]	92.0	81.0	86.2
R ² CNN [9]	93.6	82.6	87.7
Mask TextSpotter. [114]	94.1	88.5	91.0
CSE	91.8	90.2	91.0

Table 5.6: Localization performance on ICDAR 2015.

Methods	P	R	F
DeepReg [102]	82.0	80.0	81.0
EAST [3]	83.3	78.3	80.7
R ² CNN [9]	85.0	80.0	82.4
RRPN [11]	84.0	77.0	80.3
Mask TextSpotter. [114]	88.7	80.1	84.1
CSE	92.3	79.9	85.7

Table 5.7: Localization performance on MSRA-TD500.

Methods	P	R	F
DeepReg [102]	77.0	70.0	73.3
EAST [3]	87.3	67.4	76.1
RRD [10]	87.0	73.0	79.4
MOSTD [13]	87.6	76.2	81.5
MCN [110]	88.2	79.2	83.5
CSE	88.5	78.7	83.3

5.5 Performance on Non-curve Text Benchmarks

We evaluate the performance of our method to detect non-curve texts on ICDAR-2015 [88]. As illustrated in Tab. 5.6, our method shows impressive performance on non-curve text datasets containing texts with various shapes and orientations. Our CSE method achieves the state-of-art performance with a precision of 92.3, recall of 79.9 and F-score of 85.7.

5.6 Conclusion

In this work, we analyze the deficiency of the existing curve text detection methods and improve the performance by developing a novel parameterized Conditional Spatial

Expansion (CSE) mechanism. Our method shows strong robustness to the ambiguity caused by close texts with arbitrary shapes and orientations. It is flexible and can extract text regions in a controllable manner. Our CSE method outperforms the existing curve text detection methods.

Chapter 6

Conclusion and Future Works

6.1 Conclusion

In this thesis, we represent our research on detecting scene texts with highly varying shapes and complex layout. Various orientations, large aspect ratio, irregular geometry and complex layout of text object impose a great challenge when using traditional methods to detect scene texts. New detection methods and frameworks are demanded for better performance and generality.

We first consider the scene text detection problem in Chapter 3 from a bottom-up perspective. Uniquely different from the traditional top-down detection methods, the proposed Markov Clustering Network extracts text instances by exploring the potential correlation between pixels. This method represents the objects in an image as clusters, which can be discovered by Markov Clustering without knowing the number of clusters. In addition to developing a method that achieves the state-of-art performance, this research reveals that an object can be jointly represented by multiple local predictions and there exists an efficient method to aggregate these seemingly unrelated cues into an integrated object representation.

Further, in Chapter 4, a 2-dimensional Position-aware Attention (2D-PAA) mechanism is proposed to enhance MCN’s performance. In MCN, an object is jointly indicated

by multiple observations at different locations. The stochastic flows predicted by a neural network are essentially mutual-dependent and explicit information communication among nodes is desired to improve performance. The proposed 2D-PAA acts exactly as an information highway that conveys spatial correlation among nodes. Instead of using recurrent structure to encode sequential context, the positional embedding is introduced in traditional 2-dimensional attention to capture the sequential context of a text with higher computation parallelism. The proposed method effectively enhances the capability to retrieve long texts by utilizing all the representations on the spatial domain selectively. By sub-sampling a portion of input of 2D-PAA, the training cost is greatly reduced without performance loss.

Finally, in Chapter 5, we explore general scene text detection methods to retrieve text with arbitrary shape and layout. A systematic analysis reveals that the irregular shape and layout lead to noisy representations extracted by region-based detection and segmentation algorithms. Multiple text objects included in the same region proposal introduce ambiguity, which results in failures of polygon regression and segmentation. To address this issue, a novel Conditional Spatial Expansion (CSE) mechanism is proposed to improve the performance of arbitrary (curve) text detection. Instead of regarding the curve text detection as a polygon regression or a segmentation problem, we treat it as a region expansion process. Our CSE starts with a seed arbitrarily initialized within a text region and progressively merges neighborhood regions based on the extracted local features by a CNN and contextual information of merged regions. This method can handle texts with various shapes, sizes, and orientations, and can effectively suppress the false-positives coming from text-like textures or unexpected texts included in the same region of interest.

6.2 Future Works

Despite the active studies in this field, there are still some limitations with the proposed methods, and further studies and improvements are demanded. The Markov Clustering Network considers a text instance as a connected region. However, this assumption may not hold for a text object with characters far from each other. A hierarchical Markov Clustering could be the solution where multiple stochastic flow graphs are constructed on feature maps with different resolutions. A cross-hierarchy Markov Clustering algorithm can be developed to discover objects in different granularity. Another research direction could be an extension of the Markov Clustering Network to general instance segmentation. We have demonstrated the potentials to represent a text object in a bottom-up manner. This idea could be extended to a general instance segmentation scenario. The stochastic flow graph constructed by five local flows seems to be insufficient when representing an object with a complex outline. This problem could be solved by introducing a graph with a more diverse flow structure (e.g. a graph with nine local flows).

As for the Conditional Spatial Expansion (CSE), there is much room to improve for the first-stage seed initializer. Currently, a strong object detector such as Faster R-CNN is applied to initialize the seed location and the corresponding grid. Usually, a strong neural network-based object detection required to accurately localize objects with various shapes and sizes has a large number of network parameters. However, our CSE works well given any seed initialized within a text region and thus does not heavily rely on such strong detection to produce a good performance. Instead, a strong detector results in an excessive waste of memory and computation resources. Since the design objective is different from that in general object detection, a new design strategy is required to achieve better performance. One potential solution is to adopt the idea from reinforcement learning to train an initializer with progressive regression. The progressive regression can be seen as a set of actions produced by an agent to refine seed location and grid size. The environment could be the observed region in the current regression step. A positive

reward is assigned if the seed initializer produces a seed within a text with a grid cover the whole text region. The policy gradient method does not impose a strict constraint to the regressor, which could improve the performance of classification.

Appendix A

Publication List

A.1 Conference

- Z. Liu, G. Lin, S. Yang, J. Feng, W. Lin, and W. L. Goh, “Learning markov clustering networks for scene text detection,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018
- Z. Liu, G. Lin, S. Yang, F. Liu, W. Lin, and W. L. Goh, “Towards robust curve text detection with conditional spatial expansion,” *The IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7269–7278, 2019
- Z. Liu, Y. Li, F. Ren, W. L. Goh, and H. Yu, “Squeezedtext: A real-time scene text recognition by binary convolutional encoder-decoder network,” *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018

A.2 Journal

- Z. Liu, W. L. Goh, and G. Lin, “Bottom-up scene text detection with markov clustering networks,” *International Journal on Computer Vision*, 2020

Bibliography

- [1] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, “Reading text in the wild with convolutional neural networks,” *International Journal of Computer Vision*, vol. 116, no. 1, pp. 1–20, 2016.
- [2] 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.
- [3] 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 of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 5551–5560.
- [4] T. He, Z. Tian, W. Huang, C. Shen, Y. Qiao, and C. Sun, “An end-to-end textspotter with explicit alignment and attention,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5020–5029.
- [5] X. Liu, D. Liang, S. Yan, D. Chen, Y. Qiao, and J. Yan, “Fots: Fast oriented text spotting with a unified network,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5676–5685.
- [6] Y. Liu and L. Jin, “Deep matching prior network: Toward tighter multi-oriented text detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 1962–1969.

- [7] M. Liao, B. Shi, X. Bai, X. Wang, and W. Liu, “Textboxes: A fast text detector with a single deep neural network,” in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.
- [8] M. Liao, B. Shi, and X. Bai, “Textboxes++: A single-shot oriented scene text detector,” *IEEE transactions on image processing*, vol. 27, no. 8, pp. 3676–3690, 2018.
- [9] Y. Jiang, X. Zhu, X. Wang, S. Yang, W. Li, H. Wang, P. Fu, and Z. Luo, “R2cnn: Rotational region cnn for orientation robust scene text detection,” *arXiv preprint arXiv:1706.09579*, 2017.
- [10] M. Liao, Z. Zhu, B. Shi, G.-s. Xia, and X. Bai, “Rotation-sensitive regression for oriented scene text detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 5909–5918.
- [11] J. Ma, W. Shao, H. Ye, L. Wang, H. Wang, Y. Zheng, and X. Xue, “Arbitrary-oriented scene text detection via rotation proposals,” *IEEE Transactions on Multimedia*, vol. 20, no. 11, pp. 3111–3122, 2018.
- [12] B. Shi, X. Bai, and S. Belongie, “Detecting oriented text in natural images by linking segments,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2550–2558.
- [13] P. Lyu, C. Yao, W. Wu, S. Yan, and X. Bai, “Multi-oriented scene text detection via corner localization and region segmentation,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7553–7563.
- [14] S. Long, J. Ruan, W. Zhang, X. He, W. Wu, and C. Yao, “Textsnake: A flexible representation for detecting text of arbitrary shapes,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 20–36.

- [15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
- [16] J. Deng, A. Berg, S. Satheesh, H. Su, A. Khosla, and L. Fei-Fei, “Ilsvrc-2012,” 2012.
- [17] S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: Towards real-time object detection with region proposal networks,” in *Advances in neural information processing systems*, 2015, pp. 91–99.
- [18] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.
- [19] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in *European conference on computer vision*. Springer, 2016, pp. 21–37.
- [20] P. He, W. Huang, T. He, Q. Zhu, Y. Qiao, and X. Li, “Single shot text detector with regional attention,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3047–3055.
- [21] L. Yuliang, J. Lianwen, Z. Shuaitao, and Z. Sheng, “Detecting curve text in the wild: New dataset and new solution,” *arXiv preprint arXiv:1712.02170*, 2017.
- [22] S. Zhang, Y. Liu, L. Jin, and C. Luo, “Feature enhancement network: A refined scene text detector,” in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
- [23] H. Sak, A. Senior, and F. Beaufays, “Long short-term memory recurrent neural network architectures for large scale acoustic modeling,” in *Fifteenth annual conference of the international speech communication association*, 2014.

- [24] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, “Long-term recurrent convolutional networks for visual recognition and description,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 2625–2634.
- [25] R. Girshick, “Fast r-cnn,” in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440–1448.
- [26] W. G. Hatcher and W. Yu, “A survey of deep learning: platforms, applications and emerging research trends,” *IEEE Access*, vol. 6, pp. 24 411–24 432, 2018.
- [27] B. Epshtein, E. Ofek, and Y. Wexler, “Detecting text in natural scenes with stroke width transform,” in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 2963–2970.
- [28] W. Huang, Z. Lin, J. Yang, and J. Wang, “Text localization in natural images using stroke feature transform and text covariance descriptors,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 1241–1248.
- [29] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, “Detecting texts of arbitrary orientations in natural images,” in *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2012, pp. 1083–1090.
- [30] C. Yi and Y. Tian, “Text string detection from natural scenes by structure-based partition and grouping,” *IEEE Transactions on Image Processing*, vol. 20, no. 9, pp. 2594–2605, 2011.
- [31] A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, D. J. Wu, and A. Y. Ng, “Text detection and character recognition in scene images with unsupervised feature learning,” in *ICDAR*, vol. 11, 2011, pp. 440–445.

- [32] J.-J. Lee, P.-H. Lee, S.-W. Lee, A. Yuille, and C. Koch, “Adaboost for text detection in natural scene,” in *2011 International Conference on Document Analysis and Recognition*. IEEE, 2011, pp. 429–434.
- [33] K. Wang, B. Babenko, and S. Belongie, “End-to-end scene text recognition,” in *2011 International Conference on Computer Vision*. IEEE, 2011, pp. 1457–1464.
- [34] L. Neumann and J. Matas, “A method for text localization and recognition in real-world images,” in *Asian Conference on Computer Vision*. Springer, 2010, pp. 770–783.
- [35] C. Shi, C. Wang, B. Xiao, Y. Zhang, S. Gao, and Z. Zhang, “Scene text recognition using part-based tree-structured character detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 2961–2968.
- [36] C. Yao, X. Bai, B. Shi, and W. Liu, “Strokelets: A learned multi-scale representation for scene text recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 4042–4049.
- [37] J. A. Rodriguez-Serrano, A. Gordo, and F. Perronnin, “Label embedding: A frugal baseline for text recognition,” *International Journal of Computer Vision*, vol. 113, no. 3, pp. 193–207, 2015.
- [38] A. Gordo, “Supervised mid-level features for word image representation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 2956–2964.
- [39] J. Almazán, A. Gordo, A. Fornés, and E. Valveny, “Word spotting and recognition with embedded attributes,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 12, pp. 2552–2566, 2014.

- [40] M. Busta, L. Neumann, and J. Matas, “Fasttext: Efficient unconstrained scene text detector, 2015,” in *IEEE International Conference on Computer Vision (ICCV)*, vol. 1, no. 9.
- [41] T. Quy Phan, P. Shivakumara, S. Tian, and C. Lim Tan, “Recognizing text with perspective distortion in natural scenes,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 569–576.
- [42] A. Mishra, K. Alahari, and C. Jawahar, “An mrf model for binarization of natural scene text,” in *2011 International Conference on Document Analysis and Recognition*. IEEE, 2011, pp. 11–16.
- [43] T. Wakahara and K. Kita, “Binarization of color character strings in scene images using k-means clustering and support vector machines,” in *2011 International Conference on Document Analysis and Recognition*. IEEE, 2011, pp. 274–278.
- [44] Z. Zhiwei, L. Linlin, and T. C. Lim, “Edge based binarization for video text images,” in *2010 20th International Conference on Pattern Recognition*. IEEE, 2010, pp. 133–136.
- [45] Q. Ye, W. Gao, W. Wang, and W. Zeng, “A robust text detection algorithm in images and video frames,” in *Fourth International Conference on Information, Communications and Signal Processing, 2003 and the Fourth Pacific Rim Conference on Multimedia. Proceedings of the 2003 Joint*, vol. 2. IEEE, 2003, pp. 802–806.
- [46] S. Nomura, K. Yamanaka, O. Katai, H. Kawakami, and T. Shiose, “A novel adaptive morphological approach for degraded character image segmentation,” *Pattern Recognition*, vol. 38, no. 11, pp. 1961–1975, 2005.
- [47] P. P. Roy, U. Pal, J. Llados, and M. Delalandre, “Multi-oriented and multi-sized touching character segmentation using dynamic programming,” in *2009 10th*

- International Conference on Document Analysis and Recognition*. IEEE, 2009, pp. 11–15.
- [48] P. Shivakumara, S. Bhowmick, B. Su, C. L. Tan, and U. Pal, “A new gradient based character segmentation method for video text recognition,” in *2011 International Conference on Document Analysis and Recognition*. IEEE, 2011, pp. 126–130.
- [49] K. Sheshadri and S. K. Divvala, “Exemplar driven character recognition in the wild,” in *BMVC*, 2012, pp. 1–10.
- [50] X. Chen, J. Yang, J. Zhang, and A. Waibel, “Automatic detection and recognition of signs from natural scenes,” *IEEE Transactions on image processing*, vol. 13, no. 1, pp. 87–99, 2004.
- [51] D. Karatzas and A. Antonacopoulos, “Text extraction from web images based on a split-and-merge segmentation method using colour perception,” in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, vol. 2. IEEE, 2004, pp. 634–637.
- [52] A. Mishra, K. Alahari, and C. Jawahar, “Scene text recognition using higher order language priors,” 2012.
- [53] S. Wachenfeld, H.-U. Klein, and X. Jiang, “Recognition of screen-rendered text,” in *18th International Conference on Pattern Recognition (ICPR’06)*, vol. 2. IEEE, 2006, pp. 1086–1089.
- [54] J. J. Weinman, E. Learned-Miller, and A. Hanson, “Fast lexicon-based scene text recognition with sparse belief propagation,” in *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, vol. 2. IEEE, 2007, pp. 979–983.
- [55] D. Zhang and S.-F. Chang, “A bayesian framework for fusing multiple word knowledge models in videotext recognition,” in *2003 IEEE Computer Society*

- Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, vol. 2. IEEE, 2003, pp. II–II.
- [56] N. Dalal and B. Triggs, “Histograms of oriented gradients for human detection,” 2005.
- [57] P. F. Felzenszwalb and D. P. Huttenlocher, “Pictorial structures for object recognition,” *International journal of computer vision*, vol. 61, no. 1, pp. 55–79, 2005.
- [58] L. Neumann and J. Matas, “On combining multiple segmentations in scene text recognition,” in *2013 12th International Conference on Document Analysis and Recognition*. IEEE, 2013, pp. 523–527.
- [59] A. Gupta, A. Vedaldi, and A. Zisserman, “Synthetic data for text localisation in natural images,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2315–2324.
- [60] L. Kang, Y. Li, and D. Doermann, “Orientation robust text line detection in natural images,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014, pp. 4034–4041.
- [61] Z. Zhang, C. Zhang, W. Shen, C. Yao, W. Liu, and X. Bai, “Multi-oriented text detection with fully convolutional networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 4159–4167.
- [62] D. He, X. Yang, C. Liang, Z. Zhou, A. G. Ororbi, D. Kifer, and C. Lee Giles, “Multi-scale fcn with cascaded instance aware segmentation for arbitrary oriented word spotting in the wild,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 3519–3528.

- [63] Z. Liu, G. Lin, S. Yang, J. Feng, W. Lin, and W. L. Goh, “Learning markov clustering networks for scene text detection,” *arXiv preprint arXiv:1805.08365*, 2018.
- [64] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, “Dssd: Deconvolutional single shot detector,” *arXiv preprint arXiv:1701.06659*, 2017.
- [65] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.
- [66] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask r-cnn,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.
- [67] D. Deng, H. Liu, X. Li, and D. Cai, “Pixellink: Detecting scene text via instance segmentation,” in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
- [68] Y. Wu and P. Natarajan, “Self-organized text detection with minimal post-processing via border learning,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5000–5009.
- [69] D. He, X. Yang, W. Huang, Z. Zhou, D. Kifer, and C. L. Giles, “Aggregating local context for accurate scene text detection,” in *Asian Conference on Computer Vision*. Springer, 2016, pp. 280–296.
- [70] Z. Tian, W. Huang, T. He, P. He, and Y. Qiao, “Detecting text in natural image with connectionist text proposal network,” in *European conference on computer vision*. Springer, 2016, pp. 56–72.
- [71] C. Wang, F. Yin, and C.-L. Liu, “Scene text detection with novel superpixel based character candidate extraction,” in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 929–934.

- [72] X. Zhu, Y. Jiang, S. Yang, X. Wang, W. Li, P. Fu, H. Wang, and Z. Luo, “Deep residual text detection network for scene text,” in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 807–812.
- [73] D. Wu, R. Wang, P. Dai, Y. Zhang, and X. Cao, “Deep strip-based network with cascade learning for scene text localization,” in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 826–831.
- [74] Y. Zhou, Q. Ye, Q. Qiu, and J. Jiao, “Oriented response networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 519–528.
- [75] F. Wang, L. Zhao, X. Li, X. Wang, and D. Tao, “Geometry-aware scene text detection with instance transformation network,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1381–1389.
- [76] A. Graves, M. Liwicki, H. Bunke, J. Schmidhuber, and S. Fernández, “Unconstrained on-line handwriting recognition with recurrent neural networks,” in *Advances in neural information processing systems*, 2008, pp. 577–584.
- [77] M.-T. Luong, H. Pham, and C. D. Manning, “Effective approaches to attention-based neural machine translation,” *arXiv preprint arXiv:1508.04025*, 2015.
- [78] B. Shi, X. Bai, and C. Yao, “An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 11, pp. 2298–2304, 2016.
- [79] Y. Gao, Y. Chen, J. Wang, and H. Lu, “Reading scene text with attention convolutional sequence modeling,” *arXiv preprint arXiv:1709.04303*, 2017.

- [80] X.-C. Yin, X. Yin, K. Huang, and H.-W. Hao, “Robust text detection in natural scene images,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 5, pp. 970–983, 2013.
- [81] Z. Cheng, F. Bai, Y. Xu, G. Zheng, S. Pu, and S. Zhou, “Focusing attention: Towards accurate text recognition in natural images,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5076–5084.
- [82] F. Bai, Z. Cheng, Y. Niu, S. Pu, and S. Zhou, “Edit probability for scene text recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1508–1516.
- [83] C. L. Zitnick and P. Dollár, “Edge boxes: Locating object proposals from edges,” in *European conference on computer vision*. Springer, 2014, pp. 391–405.
- [84] P. Dollár, R. Appel, S. Belongie, and P. Perona, “Fast feature pyramids for object detection,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 36, no. 8, pp. 1532–1545, 2014.
- [85] C. Bartz, H. Yang, and C. Meinel, “See: towards semi-supervised end-to-end scene text recognition,” in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
- [86] H. Li, P. Wang, and C. Shen, “Towards end-to-end text spotting with convolutional recurrent neural networks,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 5238–5246.
- [87] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L. G. i Bigorda, S. R. Mestre, J. Mas, D. F. Mota, J. A. Almazan, and L. P. de las Heras, “Icdar 2013 robust reading competition,” in *2013 12th International Conference on Document Analysis and Recognition*. IEEE, 2013, pp. 1484–1493.

- [88] 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 *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*. IEEE, 2015, pp. 1156–1160.
- [89] “Icdar 2017 robust reading competition,” <http://u-pat.org/ICDAR2017/index.php>.
- [90] “Overview - icdar2019 robust reading challenge on arbitrary-shaped text.”
- [91] C. K. Ch’ng and C. S. Chan, “Total-text: A comprehensive dataset for scene text detection and recognition,” in *Document Analysis and Recognition (ICDAR), 2017 14th IAPR International Conference on*, vol. 1. IEEE, 2017, pp. 935–942.
- [92] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes challenge: A retrospective,” *International journal of computer vision*, vol. 111, no. 1, pp. 98–136, 2015.
- [93] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, R. Young, K. Ashida, H. Nagai, M. Okamoto, H. Yamamoto *et al.*, “Icdar 2003 robust reading competitions: entries, results, and future directions,” *International Journal of Document Analysis and Recognition (IJDAR)*, vol. 7, no. 2-3, pp. 105–122, 2005.
- [94] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L. G. i Bigorda, S. R. Mestre, J. Mas, D. F. Mota, J. A. Almazan, and L. P. De Las Heras, “Icdar 2013 robust reading competition,” in *2013 12th International Conference on Document Analysis and Recognition*. IEEE, 2013, pp. 1484–1493.
- [95] S. Tian, Y. Pan, C. Huang, S. Lu, K. Yu, and C. Lim Tan, “Text flow: A unified text detection system in natural scene images,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 4651–4659.
- [96] J. Dai, H. Qi, Y. Xiong, Y. Li, and G. Zhang, “Deformable convolutional networks.”

- [97] S. M. Van Dongen, “Graph clustering by flow simulation,” Ph.D. dissertation, 2001.
- [98] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” *arXiv preprint arXiv:1612.03144*, 2016.
- [99] C. Wolf and J.-M. Jolion, “Object count/area graphs for the evaluation of object detection and segmentation algorithms,” *International Journal of Document Analysis and Recognition (IJDAR)*, vol. 8, no. 4, pp. 280–296, 2006.
- [100] C. Yao, X. Bai, and W. Liu, “A unified framework for multioriented text detection and recognition,” *IEEE Transactions on Image Processing*, vol. 23, no. 11, pp. 4737–4749, 2014.
- [101] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, “Tensorflow: Large-scale machine learning on heterogeneous distributed systems,” *arXiv preprint arXiv:1603.04467*, 2016.
- [102] W. He, X.-Y. Zhang, F. Yin, and C.-L. Liu, “Deep direct regression for multi-oriented scene text detection,” *arXiv preprint arXiv:1703.08289*, 2017.
- [103] Y. Li and J. Ma, “A unified deep neural network for scene text detection,” in *International Conference on Intelligent Computing*. Springer, 2017, pp. 101–112.
- [104] J. Nickolls, I. Buck, M. Garland, and K. Skadron, “Scalable parallel programming with cuda,” *Queue*, vol. 6, no. 2, pp. 40–53, 2008.
- [105] S. Semeniuta, A. Severyn, and E. Barth, “Recurrent dropout without memory loss,” *arXiv preprint arXiv:1603.05118*, 2016.
- [106] T. Shen, T. Zhou, G. Long, J. Jiang, S. Pan, and C. Zhang, “Disan: Directional self-attention network for rnn/cnn-free language understanding,” in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

- [107] P. Shaw, J. Uszkoreit, and A. Vaswani, “Self-attention with relative position representations,” *arXiv preprint arXiv:1803.02155*, 2018.
- [108] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
- [109] A. Shrivastava, A. Gupta, and R. Girshick, “Training region-based object detectors with online hard example mining,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 761–769.
- [110] Z. Liu, G. Lin, S. Yang, J. Feng, W. Lin, and W. L. Goh, “Learning markov clustering networks for scene text detection,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.
- [111] F. Jiang, Z. Hao, and X. Liu, “Deep scene text detection with connected component proposals,” *arXiv preprint arXiv:1708.05133*, 2017.
- [112] H. Hu, C. Zhang, Y. Luo, Y. Wang, J. Han, and E. Ding, “Wordsup: Exploiting word annotations for character based text detection,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017.
- [113] Y. Dai, Z. Huang, Y. Gao, and K. Chen, “Fused text segmentation networks for multi-oriented scene text detection,” *arXiv preprint arXiv:1709.03272*, 2017.
- [114] P. Lyu, M. Liao, C. Yao, W. Wu, and X. Bai, “Mask TextSpotter: An End-to-End Trainable Neural Network for Spotting Text with Arbitrary Shapes,” *ArXiv e-prints*, Jul. 2018.
- [115] F. Zhan and S. Lu, “Esir: End-to-end scene text recognition via iterative image rectification,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

- [116] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
- [117] J. L. Ba, J. R. Kiros, and G. E. Hinton, “Layer normalization,” *arXiv preprint arXiv:1607.06450*, 2016.
- [118] “Resnet-34,” <https://www.kaggle.com/pytorch/resnet34>.
- [119] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” in *CVPR*, vol. 1, no. 2, 2017, p. 4.
- [120] S. Long, J. Ruan, W. Zhang, X. He, W. Wu, and C. Yao, “TextSnake: A Flexible Representation for Detecting Text of Arbitrary Shapes,” *ArXiv e-prints*, Jul. 2018.
- [121] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.
- [122] Z. Liu, G. Lin, S. Yang, F. Liu, W. Lin, and W. L. Goh, “Towards robust curve text detection with conditional spatial expansion,” *The IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7269–7278, 2019.
- [123] Z. Liu, Y. Li, F. Ren, W. L. Goh, and H. Yu, “Squeezedtext: A real-time scene text recognition by binary convolutional encoder-decoder network,” *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.
- [124] Z. Liu, W. L. Goh, and G. Lin, “Bottom-up scene text detection with markov clustering networks,” *International Journal on Computer Vision*, 2020.