FPT University

GRADUATION THESIS

Efficient Multi-Scale for Arbitrary Scene Text Detection for High-Resolution Image

Do Quang Manh
Tran Minh Khoi
Duong Minh Hieu

Bachelor of Aritificial Intelligence

Supervisor: Assoc. Prof. Phan Duy Hung _____

Supervisor's signature

Major: Aritificial Intelligence

University: FPT University

Hoa Lac campus, 8/2023

ACKNOWLEDGMENT

Our earnest gratitude is extended to Assoc. Prof. Phan Duy Hung, our esteemed thesis advisor, whose unwavering support, steadfast guidance, and nurturing mentorship have significantly illuminated and steered our academic odyssey. The sagacity of his domain knowledge and perceptive insights has undeniably proven instrumental in shaping the contours of our intellectual discourse. Our sentiments are deeply rooted in appreciation for the multifaceted contributions he has dedicatedly bestowed upon us.

With genuine appreciation, we acknowledge FPT University for furnishing us with the essential prerequisites, infrastructural facilities, and suitable conduits requisite for the successful culmination of our graduation thesis. The institution's dedication to erudition and research has unequivocally kindled our aspirations. As beneficiaries of this distinguished academic institution, we pride ourselves in being integral to its erudite legacy.

Furthermore, our heartfelt recognition extends to our cherished families and stalwart friends, who have been the pillars of unwavering affection, relentless support, and consistent encouragement. Their unswerving faith in our potential and ceaseless motivational fervor have indomitably fortified our determination to realize our ambitions. It is with perpetual gratitude that we acknowledge their indispensable roles in our achievements, acknowledging their contributions have been vital to our accomplishments.

In summation, an overarching sentiment of gratitude envelops all who have woven themselves into the tapestry of our academic journey. Their respective roles and cumulative contributions bear an inestimable significance that has enriched our academic sojourn. With profound appreciation, we acknowledge each individual's role in aiding us every stride of the way, propelling us to this pivotal juncture in our scholarly endeavors. I would like to express my deep gratitude to everyone who has played a role in my academic journey. Your contributions have been invaluable, and I will always be thankful for all that you have done for me. Thank you for being there for me every step of the way and for helping me reach this important milestone in my life.

ABSTRACT

Many datasets centered around scene text detection have emerged with the progressive evolution of deep learning techniques. These datasets exhibit attributes of high-resolution imagery containing diminutive textual elements, thereby establishing a burgeoning trend in computational tasks. Conventional approaches to mitigate the challenge of small text within these images involve downsizing the image dimensions. However, such a strategy often leads to text obfuscation and perceptual deterioration, consequently undermining performance outcomes. Thus, the employment of substantial models operating on enlarged input scales becomes imperative, albeit demanding significant GPU computational resources and prolonged training durations.

In the context of this investigative inquiry, we introduce "TextFocus," an algorithm designed to harness a multi-scale training strategy optimally and efficiently. Instead of meticulously scrutinizing individual pixels across an image pyramid, the TextFocus algorithm adopts a discerning approach. It endeavors to delineate contextual domains encompassing instances of ground-truth text, referred to as "chips." Subsequently, the algorithm engages in an intricate process of identifying all textual regions within the sampled image. This entails accumulating comprehensive textual insights from each "chip," which are subjected to meticulous post-processing techniques, culminating in deriving definitive outcomes for text detection.

The prowess of TextFocus lies in its capacity to adeptly transmute expansive image samples, boasting dimensions of 4000x4000 pixels, into scaled-down, lower-resolution "chips" measuring 640x640 pixels. This transformation imparts a dual advantage of expediting training procedures and enabling the accommodation of larger batch sizes, with a remarkable upper limit of 50 batches on a solitary GPU, even under conventional scaling paradigms. While the prevailing wisdom dictates an incremental enhancement in outcomes with augmented training dimensions, our approach deviates from this paradigm. Our experimentation illustrates that training on high-resolution scales might not yield optimal performance.

Our implementation employs a ResNet-18 backbone, augmented by a segment-like head architecture. The empirical outcomes showcase a commendable F1 score of 0.828 on the SCUT-CTW1500 dataset [1], alongside a respectable F1 score of 0.611 on the Large CTW dataset [2]. These achievements are coupled with a real-time operational capacity, as substantiated by the acceptable frames per second (FPS) metric.

TABLE OF CONTENTS

CHAPTER 1. INTRODUCTION 1

1.1 Scene text detection 1

1.2 Arbitrary shape text detection 2

1.3 Background and problems of research 3

1.4 Research objectives 5

1.5 Contributions 6

1.6 Organization of Thesis 6

CHAPTER 2. THEORETICAL FOUNDATION 8

2.1 Related works 8

2.1.1 History of arbitrary shape text detection methods 8

2.1.2 Studies that directly inspired our research 9

2.2 Foundational theories 10

2.2.1 Resnet 18 architecture 10

2.2.2 Feature Pyramid Network 11

2.2.3 Encoder-Decoder architecture 12

CHAPTER 3. METHODOLOGY 13

3.1 Data enhancement and preprocesing 13

3.1.1 Alpha hull and Alpha shape 13

3.1.2 Generate new annotations for CTW dataset 15

3.2 Baseline architecture 16

3.2.1 Training pipeline 16

3.2.2 Inference pipeline 16

3.3 Pixel Aggregation Network - PAN 17

3.3.1 Overall PAN's Architecture 17

3.3.2 Reducing Channel Block 18
3.3.3 Feature Pyramid Enhancement Module 19
3.3.4 Feature Fusion Module 20
3.3.5 Pixel Aggregation 21
3.3.6 Loss function 23
3.4 Focus branch 23
3.4.1 Focus Pixels Finding 24
3.4.2 Focus Chips Generation 26
3.4.3 Focus Combination for final results 26
3.5 Implementing TextFocus 27
CHAPTER 4. EXPERIMENTAL RESULTS AND DISCUSSION 30
4.1 Datasets 30
4.1.1 Chinese Text in the Wild (CTW datasets) 30
4.1.2 SCUT-CTW1500 30
4.1.3 ICDAR15 30
4.1.4 TotalText 31
4.1.5 SynthText 31
4.2 Evaluation metrics 32
4.2.1 TIoU-Recall 33
4.2.2 TIoU-Precision 33
4.2.3 Tightness-aware Metric 34
4.3 Implementation detail 34
4.4 Results and analysis 36
4.4.1 The effectiveness and influence of the backbone and Detection branch .. 37
4.4.2 The effectiveness of Focus branch 37
4.4.3 Comparison results 40

CHAPTER 5. CONCLUSION AND FUTURE WORK 42

REFERENCES 48

APPENDIX 50

A. Data 50

B. MORE VISUALIZATIONS 51

LIST OF FIGURES

Figure 1.1 Illustration for the problem of Scene text detection 1
Figure 1.2 Illustration for the challenge of Scene text detection 2
Figure 1.3 Illustration for the challenge of Arbitrary text detection 3
Figure 1.4 Scene Text Detection on SCUT-CTW1500 dataset 4

Figure 2.1 The architecture of Resnet18 model 11
Figure 2.2 The architecture of FPN model 11
Figure 2.3 The architecture of UNet 12

Figure 3.1 Alpha hull and Alpha shape 13
Figure 3.2 Alpha hull and Alpha shape in different alpha values 14
Figure 3.3 Voronoi diagram and Delaunay triangulation 14
Figure 3.4 Idea of generating new boundary for CTW dataset 15
Figure 3.5 TextFocus architecture for training process. 16
Figure 3.6 TextFocus architecture for inference process. 17
Figure 3.7 PAN architecture. 18
Figure 3.8 The reducing channel block. 19
Figure 3.9 The Feature Pyramid Enhancement Module. 20
Figure 3.10 The Feature Fusion Module. 21
Figure 3.11 The architecture of Focus branch. 23
Figure 3.12 Groundtruth for focus branch 24
Figure 3.13 Description of focus branch process. 27
Figure 3.14 The complete architecture of TextFocus. 28
Figure 3.15 Polynomial Learning Rate Scheduler. 29

Figure 4.1 Unreasonable cases 32
Figure 4.2 CPU and GPU consumption of TextFocus on SCUT-CTW1500 . . 39
Figure 4.3 Visualize results on three standard benchmarks 40

Figure A.1 Area of objects of different sizes and backgrounds 50

Figure B.1 Inference pipeline in TextFocus 51

LIST OF TABLES

Table 4.1 The detail information of datasets. 31
Table 4.2 Overall training parameters. 36
Table 4.3 Parameters for groundtruth selection when generate chip for focus branch training. 36
Table 4.4 Results on Results on ICDAR2015, Total-Text, SCUT-CTW1500 datasets 36
Table 4.5 Results on CTW dataset 37
Table 4.6 GPU memory and GFlops per image of methods on SCUT-CTW1500 38

LIST OF ABBREVIATIONS

Abbreviation	Definition
CNN	Convolutional Neural Network
FCN	Fully Convolutional Network
FFM	Feature Fusion Module
FPEM	Feature Pyramid Enhancement Module
FPN	Feature Pyramid Network
GPU	Graphics processing unit
GPU	Computer processing unit
IoU	Intersection over Union
PA	Pixel Aggregation
PAN	Pixel Aggregation Network
Unet	U-shape Network

CHAPTER 1. INTRODUCTION

1.1 Scene text detection

Scene text detection represents an extensively pervasive and pivotal facet within computer vision tasks. This computational undertaking holds paramount significance, finding application across diverse domains of routine existence, encompassing endeavors such as document digitization, bill processing, language translation, and surveillance operations, exemplified by extracting information from credit cards and vehicular license plates as shown in Figure 1.1. Furthermore, the significance of scene text detection extends to encompass text-centric retrieval systems and the specialized realm of text-oriented visual question answering. Despite its manifold potentialities, several impediments preclude its seamless integration into practical contexts.

Figure 1.1: Illustration for the problem of Scene text detection. These images are results from the Canny Text Detection in [3].

Foremost among these challenges is the delicate balance between the imperatives of precision and swiftness in execution. This trade-off between the fidelity of results and the expeditiousness of analysis remains a formidable hurdle in achieving optimal operational outcomes. A second difficulty pertains to the inherently unpredictable and diverse nature of arbitrary text instances that the system encounters as shown in Figure 1.2. The erratic and variegated manifestations of text across various contexts pose intricate conundrums, demanding sophisticated solutions for robust and reliable recognition. Finally, the predicament of grappling with high-resolution imagery housing diminutive textual content amplifies the intricacy of the task. The juxtaposition of extensive visual

data with minute textual elements necessitates specialized methodologies to detect and decipher such intricate components accurately. The triumvirate of challenges involving accuracy-speed equilibrium, arbitrary-text instances, and high-resolution, small-scale text further accentuates the multifaceted nature of the scene text detection problem.

Figure 1.2: This image elucidates the intrinsic unpredictable and heterogeneous characteristics of scale exhibited by textual instances in the wild, through the utilization of bounding boxes. Figure is taken from COCO-Text dataset [4].

The landscape of scene text detection has been significantly influenced by the emergence of Deep Learning-based methodologies, most notably Convolutional Neural Networks (CNNs). These techniques have exhibited substantial promise in this domain, showcasing exceptional proficiency in attaining elevated levels of accuracy and performance. Leveraging the inherent capabilities of these models, particularly their adeptness in comprehending extensive repositories of visual data, holds immense potential. When realized, this potential can alleviate the demands placed upon users, offering a paradigm shift towards a more streamlined and precise approach to text detection.

1.2 Arbitrary shape text detection

The domain of arbitrary shape text detection constitutes a specialized and discerning realm within the expansive landscape of scene text detection and text recognition. In contrast to conventional methodologies for text detection, which predominantly center around the recognition of texts confined to rectangular or quadrilateral shapes, the pursuit of arbitrary shape text detection embarks upon the challenge of localizing and outlining textual constructs that manifest with complex, non-standard, and heterogeneous geometries as shown in Figure 1.3. This encompasses instances wherein textual elements assume contours that are convoluted, skewed, or harmoniously embedded within detailed contextual backdrops. The principal objective resides in the precise delineation of the spatial boundaries intrinsic to such textual instances, facilitating subsequent analytic undertakings and cultivating comprehension. This domain finds resonance across an expansive spectrum of application domains, including but not limited to scene text recognition, document scrutiny, image comprehension, and other cognate areas. The

Figure 1.3: The CTW1500 dataset's images [1] are meticulously curated through manual extraction from the Internet, representing both horizontally aligned text and text instances exhibiting diverse orientations.

necessity for sophisticated methodologies and models is paramount, as they stand as instrumental requisites in effectively surmounting the multifaceted challenges and nuances engendered by the identification and delineation of text in these versatile compositional configurations.

1.3 Background and problems of research

The domain of arbitrary shape text detection has undergone remarkable strides, primarily attributed to integrating deep learning methodologies. These advancements have been highlighted by notable contributions that encompass the adoption of models like the Fully Convolutional Network [5] [6], coupled with encoder-decoder architectures such as UNet [7] [8], UNet++ [9]. These models have exhibited substantial efficacy within the confines of this specialized domain. In recent developments, a class of Transformer-based paradigms, exemplified by models such as TextBPN [10]and DeepSOLO++ [11], has ascended to prominence, showcasing exceptional performance and achieving state-of-the-art outcomes.

A dichotomy exists in addressing the text detection challenge, involving two divergent approaches; one pertains to the analysis of images, while the other encapsulates the investigation of videos. Although video inputs possess the potential to furnish temporal insights into object motions, their utilization mandates intricate and resource-intensive processing procedures. So, the research discourse is often inclined toward image-based

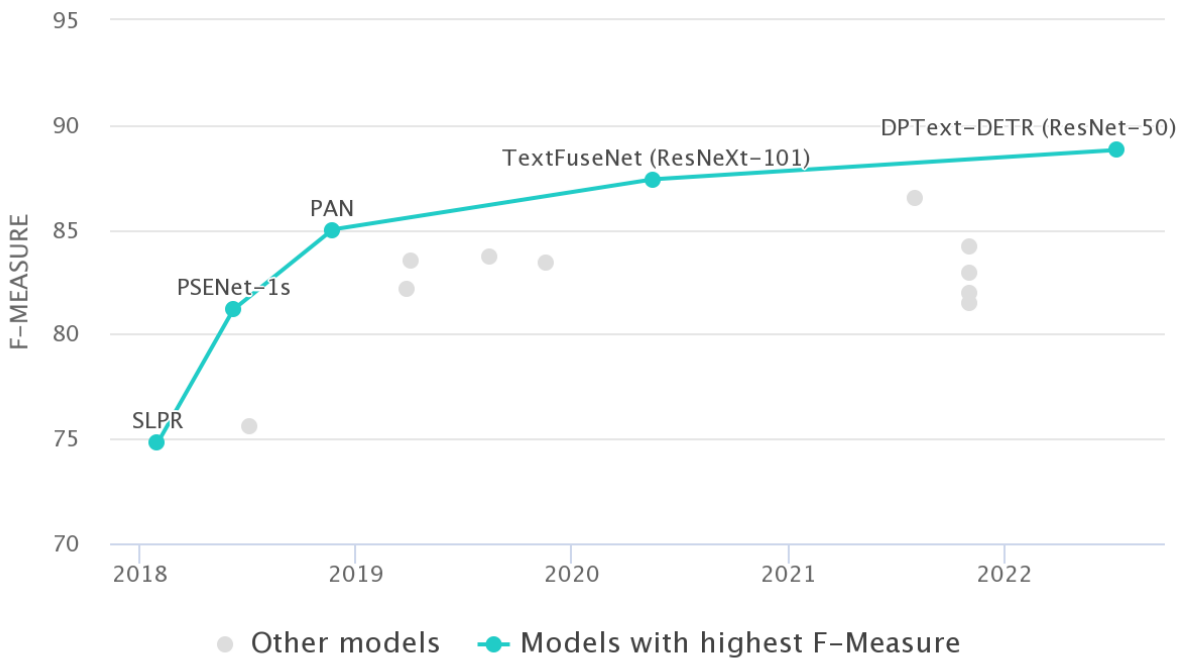


Figure 1.4: The results of some recent model for Scene Text Detection task on SCUT-CTW1500 dataset [1].

detection due to its simplicity and broad applicability.

Recent investigations in the field have predominantly revolved around the quest for novel architectural models. Nevertheless, the incremental enhancements in performance metrics achieved through these approaches have become asymptotic as shown in Figure 1.4. Increasing accuracy by a few percent on public test datasets isn't easy.

A second salient challenge pertains to the nuanced domain of arbitrary text instances. These instances, characterized by their divergence from conventional rectangular configurations, introduce complexities that representation through traditional bounding box annotations.

Lastly, the challenge associated with high-resolution images harboring small textual content compounds the barrier of scene text detection in both the training and inference process. The precision of these challenges underscores the multiscale nature of scene text detection.

In the present study, the goal is not to alter or propose the new model architecture. Instead, the emphasis is directed toward refining learning and inference strategies simultaneous with the cultivation of expanded datasets to enhance accuracy and frames per second (FPS) metrics.

1.4 Research objectives

The research objectives of this thesis, Efficient Multi-Scale for Arbitrary Scene Text Detection for High-Resolution Image," are to investigate the current state-of-the-art in arbitrary shape text detection algorithms, identify the limitations of existing methods, and use an improved approach using multiple resolution techniques.

To solve the primary research objective, we thoroughly investigate arbitrary shape text detection of the current methodologies. This investigation seeks to discern existing approaches' inherent limitations, thereby identifying critical areas that necessitate enhancement. From these experiments of previous methods, our algorithm will leverage the benefits of multiple resolutions and multiple scales techniques to overcome the limitations of existing processes, including recognizing small text accurately meanwhile ensure the speed and resources when deploying.

Central to the research objectives is the conception and development of a novel text detection algorithm. This algorithm is engineered to offer a dual advantage: the ability to accurately and expediently detect instances of text within images characterized by multi-resolution attributes. By harnessing the synergies afforded by multiple resolution techniques, our algorithm aims to surmount the deficiencies that often beset current methods, particularly their aptitude to discern diminutive textual elements. Furthermore, this algorithm remains attuned to the imperatives of computational efficiency, ensuring expedited and resource-effective deployment during its operational instantiation.

In tandem with the algorithm's conception, a research goal is evaluating the experiment results of our model on many real datasets. The assessment will be predicated upon various metrics, notably the Intersection Over Union (IOU) metric score and the Tightness-aware Intersect-over-Union (TIOU) metric [12]. These metrics encapsulate performance value, including recall, precision, and the harmonic mean (h-mean), offering a robust framework for comparative analysis.

Moreover, the ambit of this research extends to contribute substantively to the field of image processing and pattern recognition. The insights from the comprehensive exploration of multiple-resolution techniques are fundamental to providing knowledge of the advantages and limitations of multiple-resolution methods for object detection. The implications of this research are potentially affording valuable guidance for researchers and engineers working on optical character recognition (OCR) tasks.

In conclusion, this thesis aims to develop a novel arbitrary shape text detection paradigm, distinguished by its adept utilization of multiple resolution techniques. This methodology is set to pass established benchmarks, thereby contributing to advancing arbitrary shape text detection and the broader horizons of image processing and pat-

tern recognition. The implications of these findings hold the promise of serving as a foundational platform for formulating novel approaches across diverse domains of image detection.

1.5 Contributions

Our main contributions to this study include the following:

1. Applied a novel approach to arbitrary shape text detection called TextFocus, leveraging the power of multiple resolution techniques, that addresses the limitations of existing methods, precisely the trade-off between accuracy and resource consumption concerns intrinsically linked with high-resolution sample training.
2. Synthesized the data by applying the Alpha-shape algorithm to generate new annotations for the CTW dataset [2] .
3. Conducted experiments to demonstrate the effectiveness and efficiency of the proposal solution.

Our algorithm asymptotic existing state-of-the-art methods in terms of recall precision and h-means scores on the same test dataset outperform that method in terms of FPS and resource consumption. Our research provides a promising avenue for further study in text detection.

1.6 Organization of Thesis

The thesis is a comprehensive study that addresses the problem of Arbitrary shape text detection. To achieve this goal, the remaining parts of the thesis are organized as follows:

Chapter 2, entitled "Theoretical Foundation," is an essential section of the thesis that provides the basic theory used in the Scene Text Detection problem. This chapter explores the literature related to the research problem, including direct and indirect studies. The chapter aims to establish a solid foundation for the research by presenting a comprehensive understanding of the theoretical aspects of the problem.

Chapter 3, entitled "Methodology," is a critical section of the thesis that presents the method called Text Focus. This chapter includes the baseline model architecture and the Multi-resolution training and inference strategy. The methodology will be explained in detail to provide a better understanding of the approach taken in this research. This chapter aims to show how the Multi-resolution strategy can be used to improve the accuracy of the Arbitrary Shape Text Detection problem.

Chapter 4, entitled "Experimental Results and Discussion," is an essential section of the thesis that presents the experimental results and discussions. In this chapter, we will deliver the datasets used for training and testing, the evaluation metrics we used

for evaluation, the program implementation details, experimental results with different configurations, and comparisons with current best practices. The chapter aims to provide a comprehensive review of the method and comparisons with existing methods in the field. The experimental results and discussions will be presented to give a better understanding of the performance of TextFocus.

Chapter 5, entitled "Conclusion and Future Work," is the final section of the thesis that gives the conclusion of contribution, the improvements, and the weaknesses of the methods. This chapter provides an overall summary of the research, highlighting the study's main contributions. The chapter will also discuss the limitations and future directions of the method. Finally, some learning cases and future development will be presented to inspire further research.

In conclusion, the thesis aims to comprehensively understand the Arbitrary Shape Text Detection problem and present a novel approach to address this challenge. By submitting the theoretical foundation, methodology, experimental results, and future directions, the thesis aims to contribute to the research in this field and inspire future developments.

CHAPTER 2. THEORETICAL FOUNDATION

In this section, we will present related works, from the first method of Polyp Segmentation problem, to recent studies that directly inspired our research. We will then detail the underlying theories that will be used in this project.

2.1 Related works

2.1.1 History of arbitrary shape text detection methods

Arbitrary shape detection has emerged as a critical research area within scene text detection. The problem of arbitrary-shape text detection is challenging due to the variety of shapes and appearances text can take.

Before deep learning flourished, Connected Component (CC) and Traditional sliding window based had been widely used. Sliding window-based methods [13] [14] involve a multi-scale window over an image and classify the current path to detect objects or features of interest. CC-based modes [15] [16] get the character candidates by extracting CCs. And then, these candidates' CCs are classified as text or non-text.

Recently, deep learning based methods have become popular. These methods can be divided into different groups: regression-based methods, segmentation-based methods, and contour-based methods.

a, Regression-based methods

Regression-based methods [17]–[21] always modify box-regression-based object detection frameworks with word-level and line-level for text instances. However, scene texts frequently exhibit arbitrary orientations accompanied by diverse aspect ratios. To handle this problem, TextBoxes and TextBoxes++ [18] use a series of anchors with different aspect ratios. These methods comprise the text proposal generation stage, with candidate text regions generated, and the bounding box refinement stage, in which candidate text regions are verified and refined to create the final detection result.

Early solutions for polyp segmentation were mainly based on low-level features, such as texture, geometric features, or simple linear iterative clustering superpixels. They are called traditional methods **mamonov2014automated**, **maghsoudi2017superpixel**. However, due to the high similarity between polyps and surrounding tissues, these methods have a high risk of missed or false detection.

b, Segmentation-based methods

Segmentation-based methods [22]–[25] draw inspiration from semantic segmentation to implicitly encode text instances with pixels mask. PSNET[26] employs a progress scale

expansion algorithm for multi-scale segmentation map fusion. PAN [27] and LSAE [28] enhance pixel embeddings for the same text while distinguishing different texts. TextFields [29] employs a deep direction field to link neighboring pixels, generating potential text instances. DB[30] simplifies text detection's postprocessing using differentiable binarization within a segmentation network. These methods emphasize segmentation accuracy as a key determinant of boundary detection quality.

c, Contour-based methods

Contour-based methods [31]–[34] directly model text boundaries to detect arbitrary-shape text. ABCNet [33] and FCENet [34] employ curve modeling (Bezier-Curve and Fourtier-Curve) for text instance contours, accommodation progressive approximation of closed shapes. TextRay [35] introduces the polar system formulation of text contours using a single-shot anchor-free framework to predict geometric parameters and generate simple polygon detections. PCR [36] proposes a progressive contour regression within a top-down detection framework for arbitrary-shape scene text detection. Similar top-down frameworks are used by some methods [31], [32], which regress key points on text contours within text proposals. Relative to segmentation-based methods, considerable room remains for performance and speed enhancement exploration.

d, Other methods

In addition, there are alternative approaches. End-to-end methods [22], [37], [38]integrate text detection and recognition within a single network. These approaches can improve detection performance by leveraging text recognition information [37]. Moreover, Yao et al. [39] predict text corner points for text detection, and Lyu et al. [31] adopt a similar architecture skin to SSD [40] to reconstruct text instances based on predicted corner points.

2.1.2 Studies that directly inspired our research

In response to the challenges mentioned earlier, we introduce our solution named TextFocus, which comprises two main components:

Pixel Aggregation Network (PAN): The initial branch of TextFocus leverages the Pixel Aggregation Network (PAN) [27], an arbitrary shape text detector that can archive a good balance between speed and performance. PAN uses Resnet-18 [41] as a backbone in its architecture and has a segmentation head characterized by its low computational overhead and lightweight nature while upholding a high-performance standard. Furthermore, unlike many other text detectors, PAN predicts the text regions, kernels, and similarity vectors rather than just the instances themselves. Since the chips produced by the focus branch can overlap or split a particular text instance into multiple parts, directly detecting instances is not the best option.

Focused Branch: The second branch introduces the focused branch, a straightforward algorithm that orchestrates the exploration of regions deserving attention on the more extensive scale within the image pyramid. The focus branch searches for plausible text-inclusive areas within the image at each incremental scale, generating chips destined for the subsequent image scale. Noteworthy is that the branch selectively processes a mere 20% of the scale encompassed by the preceding image instead of directly assimilating the entire next-scale appearance. This strategic simplification augments the training process's efficiency, economizes on hardware, memory, and temporal resources, and underscores its efficacy through empirical validation across four challenging datasets: SCUT-CTW1500 [1], ICDAR 2015 [42], TotalText [43], and CTW [2].

These two distinct components are synergistically harmonized to orchestrate a cohesive and synchronized workflow that aligns with our overarching objectives. Notably, our study rigorously curates an advanced pipeline; each block has been precisely examined to generate the final process graph.

The prevalence of high-resolution inputs and intricate annotations within scene text datasets has surged in the landscape of the explosion of deep learning applications. However, these inputs are not suitable for conventional scene text detection. In response, we employ data preprocessing techniques to reformat the original annotations, optimizing their utility as model inputs.

It is imperative to underscore that each chip generated by the focus branch undergoes processing within the model, thereby engendering a direct correlation between chip quantity and processing duration. We have devised a postprocessing methodology that aligns seamlessly with utilizing the model mentioned above components. Specifically, the postprocessing approach accommodates scenarios wherein long text instances are divided into multiple chips, simultaneously mitigating overlapping issues.

2.2 Foundational theories

This thesis contains numerous terms, and providing a detailed explanation of each is outside this study's scope. However, in this section, we will try to present the most crucial theories related to and utilized in this thesis.

2.2.1 Resnet 18 architecture

The convolutional neural network architecture known as ResNet was formulated by Microsoft Research in their publication "Deep Residual Learning for Image Recognition" [41]. The Resnet-18 architecture is a widely used deep learning technique that has been successfully applied to various tasks in computer vision. This specific variant of the ResNet model has been meticulously tailored to cater to the demands of image classification tasks.

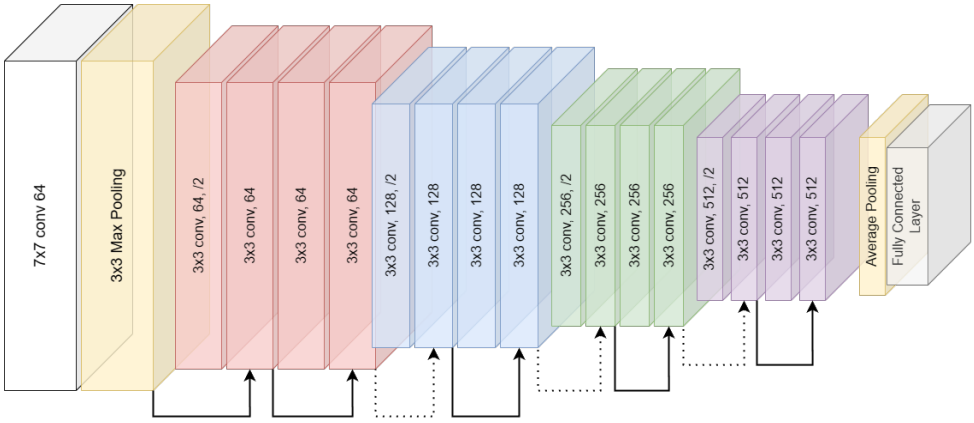


Figure 2.1: The architecture of Resnet18 model.[41]

2.2.2 Feature Pyramid Network

The Feature Pyramid Network (FPN) [44] framework constitutes a comprehensive architectural paradigm that encompasses two fundamental pathways: the bottom-up and the top-down pathways. This innovative design is engineered to facilitate robust feature extraction and semantic comprehension across varying spatial scales within the context of object detection and related computer vision tasks as shown in Figure 2.2.

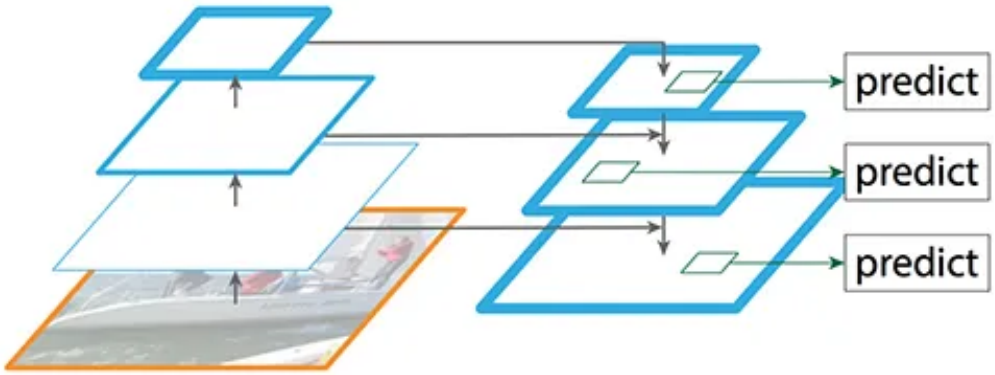


Figure 2.2: The architecture of Feature Pyramid Network model. [44].

2.2.3 Encoder-Decoder architecture

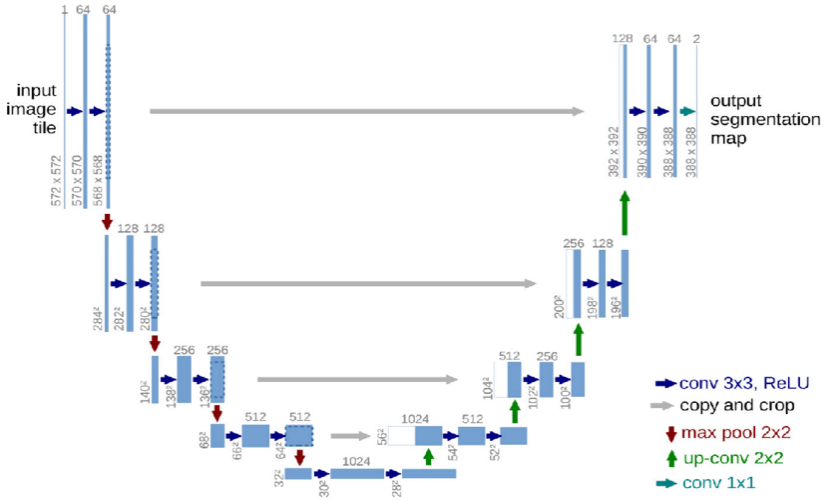


Figure 2.3: The details of the UNet model. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations. [45].

The Encoder-Decoder architecture is a widely used deep learning technique successfully applied to various tasks in computer vision and natural language processing. The architecture consists of two main components, the encoder and the decoder, which transform an input sequence or image into an output sequence or embodiment.

In this chapter, we have provided a detailed presentation of the related works and foundational theories that are relevant to the project. Our objective was to offer a thorough understanding of the underlying theories, which will enable readers to appreciate our TextFocus method, presented in the next chapter. This background information aims to provide a comprehensive context for the TextFocus method, highlighting its unique features and advantages in text detection.

CHAPTER 3. METHODOLOGY

The design of our TextFocus model is a comprehensive end-to-end framework encompassing training and inference phases. This foundational structure of our model embraces an orchestrated between discrete yet harmonizing constituents: the Pixel Aggregation Network (PAN) [27] and the Focus Branch [46]. Furthermore, we introduce a novel method to generate annotations that facilitate the representation of arbitrary shapes for text instances. This method leverages the bounding boxes attributed to each text character and, through the Alpha-shape algorithm, produces the text instances with contours that transcend the confines of conventional geometric. The first section introduces reconstructing novel datasets and the overall architectural arrangement of the TextFocus model. The subsequent explains the details of our training and inference strategy to optimize the operational efficacy of the model.

3.1 Data enhancement and preprocesing

3.1.1 Alpha hull and Alpha shape

The concept of α hulls was introduced by Edelsbrunner et al. (1983) [47] as a natural generalization of convex hulls. The positive α hull of a set of points, such as $p1, p2, ..., pn$, is defined as the intersection of all closed discs (referred to as α discs) with radius R_α (where $R_\alpha = \frac{1}{a}$) that contain all the points. In contrast, the negative α hull is the intersection of all closed compliments of discs that contain all the points. This is generated by point pairs that can be touched by an empty disc of radius R_α

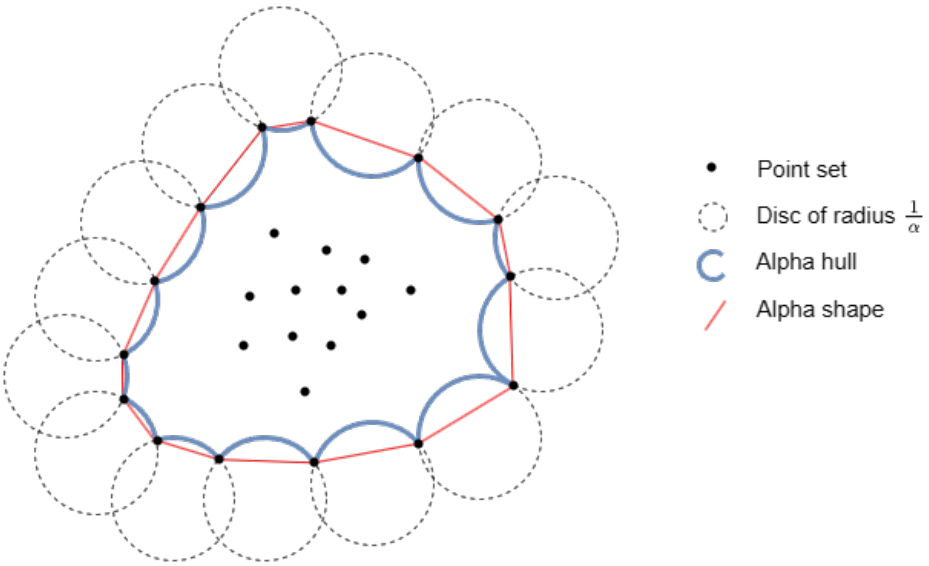


Figure 3.1: Alpha hull and Alpha shape

In computational geometry, alpha shapes find application in the abstraction of the

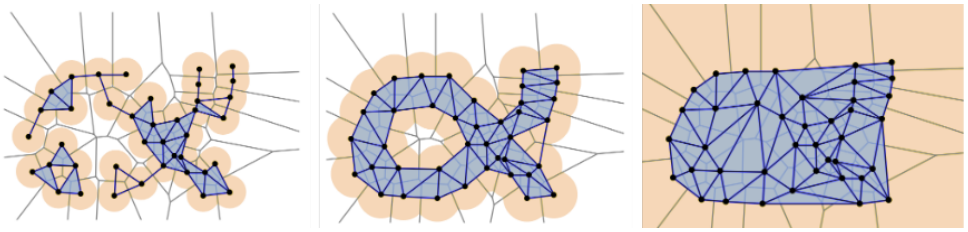
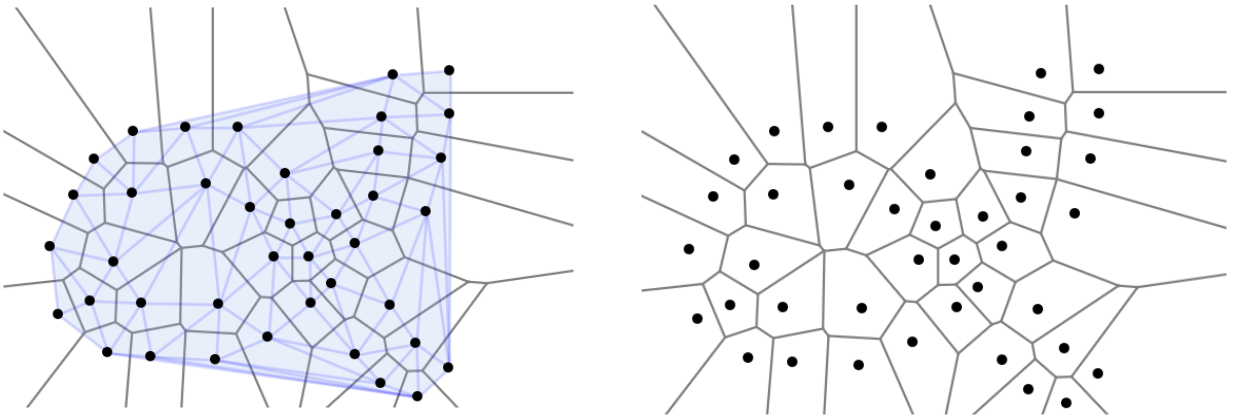


Figure 3.2: From left to right: the α-hull of a set of points sampling the shape of the symbol α in the plane, the α-shape of the same set, and the union of disks of radius α centered at the points



Voronoi diagram of a set of points Voronoi diagram and Delaunay triangulation

Figure 3.3: Voronoi diagram and Delaunay triangulation

convex hull on a finite assemblage of points within the confines of the Euclidean plane. The defining trait of an alpha hull manifests through its arcuate edges, which traverse point pairs, mirroring the contour of a curved disk's periphery. This delineation of the alpha shape materializes after substituting these arcuate edges with linear interconnections bridging the respective point pairs. A graphical depiction of the contrasting attributes between alpha shapes and the convex hull is proffered in Figure 3.1.

Figure 3.1 illustrates this concept, where the dashed circle represents a disc with a radius of $\frac{1}{\alpha}$, the blue arcs notation denotes the boundary of the alpha hull, and the red line delineates the border of the alpha shape between two alpha nodes (also referred to as points in design space). The structure of the alpha shape is dependent on the value of α, meaning that for a given set of points, the shape will differ as α changes in Figure 3.2 . In this work, alpha shapes are used for reliability estimation, as they can be represented as linear boundaries that are well-suited for analytical approaches such as the First Order Reliability Method (FORM).

In the context of this study, the utility of alpha shapes extends to the realm of reliability estimation. This efficacy emanates from their amenable representation as linear boundaries, harmonizing adeptly with analytical methodologies such as the First Order Reliability Method (FORM).

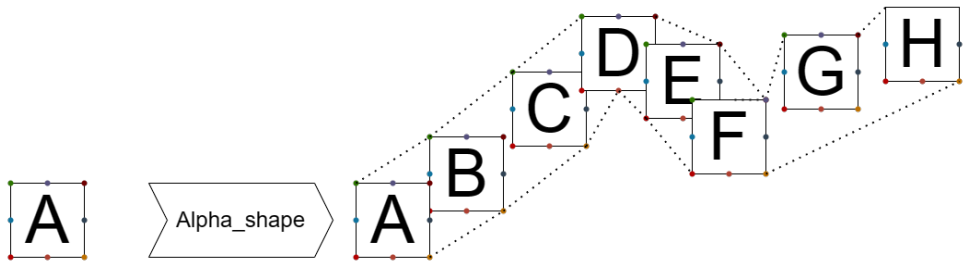


Figure 3.4: Idea of generating new boundary for CTW dataset [2]

The alpha shape emerges as an elegant and computationally efficient construct, intrinsically intertwined with two fundamental geometric structures: the Voronoi diagram and the Delaunay triangulation. A Voronoi region of a specific point, denoted as $p1$, encompasses all points (designated as point i) that exhibit equidistant proximity to both p1 and another distinct point within the set. The Voronoi diagram essentially includes the amalgamation of these Voronoi regions for all constituent points within the collection. A visual representation of the Voronoi diagram corresponding to a designated group of subjects is delineated in Figure 3.3. This diagram vividly illustrates the space partitioning into Voronoi regions, each inextricably associated with its respective point.

3.1.2 Generate new annotations for CTW dataset

For the purposes of training and validation, the present study made use of the Chinese Text in the Wild (CTW) Dataset, as delineated in Yuan et al. [2]. However, it is pertinent to underscore that this dataset exclusively encompassed annotations corresponding to instances of Chinese characters discernible within each individual image. To transcend this inherent limitation and facilitate the incorporation of text instances characterized by arbitrary shapes, a strategic recourse was adopted involving the deployment of the Alpha-Shape algorithm. This algorithm was instrumental in circumventing the aforementioned constraint and served as the mechanism through which delineations of text instance boundaries endowed with arbitrary geometries were synthesized.

The operational sequence of this approach encompassed the systematic computation of the coordinates aligned with all vertices and intermediary midpoints distributed along the edges of the bounding boxes that encapsulate each character instance. These computed coordinates were subsequently harnessed as the fundamental inputs for the Alpha-Shape algorithm. Following this, meticulous fine-tuning of the alpha parameter was undertaken to generate an entirely novel dataset replete with annotations. The values engendered within this dataset were meticulously structured to be readily conducive for delineating borders characterized by arbitrary shapes. This strategic augmentation effectively extended the dataset's capacity to accommodate the heterogeneous array of text lines prevalent across the images under investigation, a visual representation of which is vividly

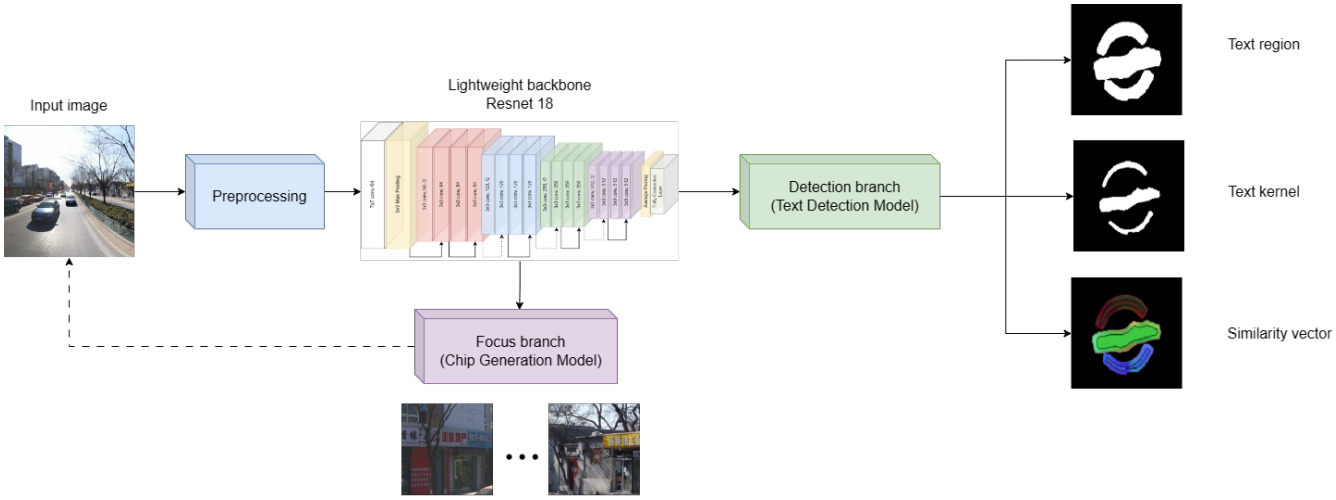


Figure 3.5: TextFocus for training process.

elucidated in Figure 3.4.

3.2 Baseline architecture

In this thesis, we present distinct strategies for the training pipeline and the inference (or detection) pipeline due to their inherent dissimilarity.

3.2.1 Training pipeline

In this thesis, we present distinct strategies for the training pipeline and the inference (or detection) pipeline due to their inherent dissimilarity.

Illustrated in Figure 3.5, the training procedure encompasses the subsequent steps: Before ingestion into the model, input images, exhibiting diverse annotator formats contingent on the dataset, undergo preprocessing to establish uniformity. Post-preprocessing, utilizing a lightweight backbone (ResNet-18) [41], the model handles the processed inputs. It subsequently channels the outputs through dual branches, yielding outcomes encompassing text regions, text kernels, similarity vectors for subsequent textual outcome instances, and focus maps intended to generate higher-level chips. It is pertinent to underscore that the path represented by the dotted line in Figure 3.5 describes that this cycle is entirely distinct from the result of the prediction branch.

3.2.2 Inference pipeline

Given the required time, conducting sequential processing for every sample is impractical, mainly as the chips generated across varying scales encompass the original scale. As depicted in Figure 3.6, the inference procedure is outlined in some steps. The processing phase is the same as the training phase; inputs traverse a lightweight backbone within each scale iteration, yielding feature and focus maps. The focus map serves the purpose of generating chips to be employed in subsequent scales. Upon the cessation of

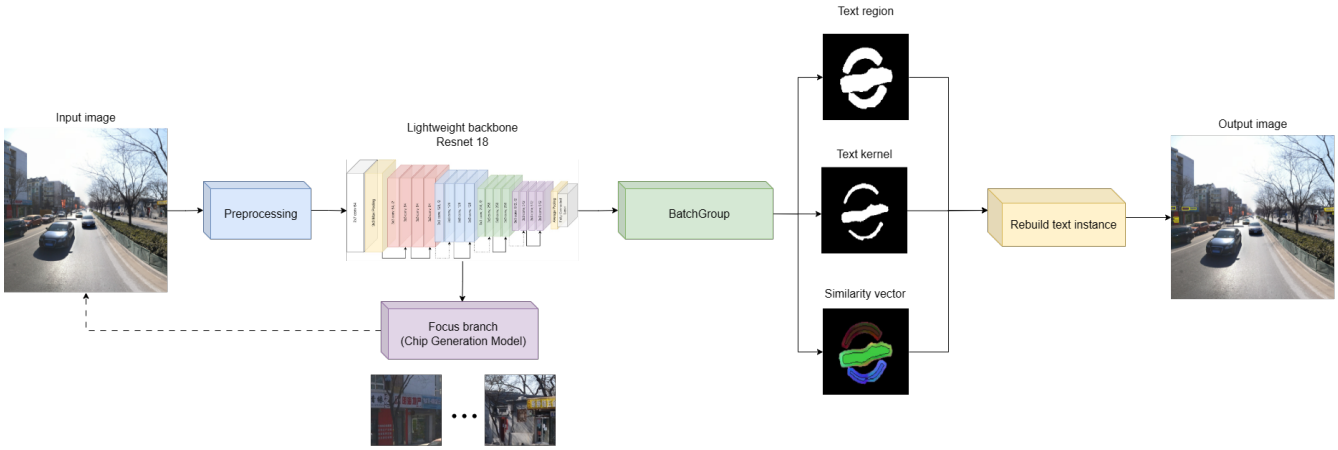


Figure 3.6: TextFocus for inference process.

chip generation, all features are amalgamated into a batch and forwarded to the residual branch. This branch produces the final prediction for the text instance.

More information on each block and approach is provided below.

3.3 Pixel Aggregation Network - PAN

Pixel Aggregation Network (PAN) is an optimized approach by Wang et al [27] for detecting text instances with arbitrary shapes, primarily due to its adept balance between speed and performance. To achieve heightened efficiency, the segmentation network's backbone must be lightweight. Nonetheless, such lightweight architectures often yield features characterized by diminutive receptive fields and limited representation capabilities. To address this challenge, the model was designed with a computationally efficient segmentation head to refine the extracted features. This segmentation head encompasses two pivotal components: the Feature Pyramid Enhancement Module (FPEM) and the Feature Fusion Module (FFM). In this study, we embrace this methodology to address the intricacies of detecting arbitrary-shaped text within images of varying resolutions.

3.3.1 Overall PAN's Architecture

Figure 3.7 illustrates the comprehensive architecture of PAN, wherein the backbone network employs a lightweight model, specifically ResNet-18 [41]. The backbone generates four distinct feature maps through $conv2$, $conv3$, $conv4$, and $conv5$ layers, corresponding to spatial strides of 4, 8, 16, and 32 pixels relative to the input image. To reduce the channel dimension of each feature map to 128 and achieve a more streamlined feature pyramid, a module for channel reduction is implemented utilizing a 1×1 convolutional layer.

The thin feature pyramid undergoes augmentation via a series of n_c cascaded Feature Pyramid Enhancement Modules (FPEMs). These FPEMs are engineered to be cascadable,

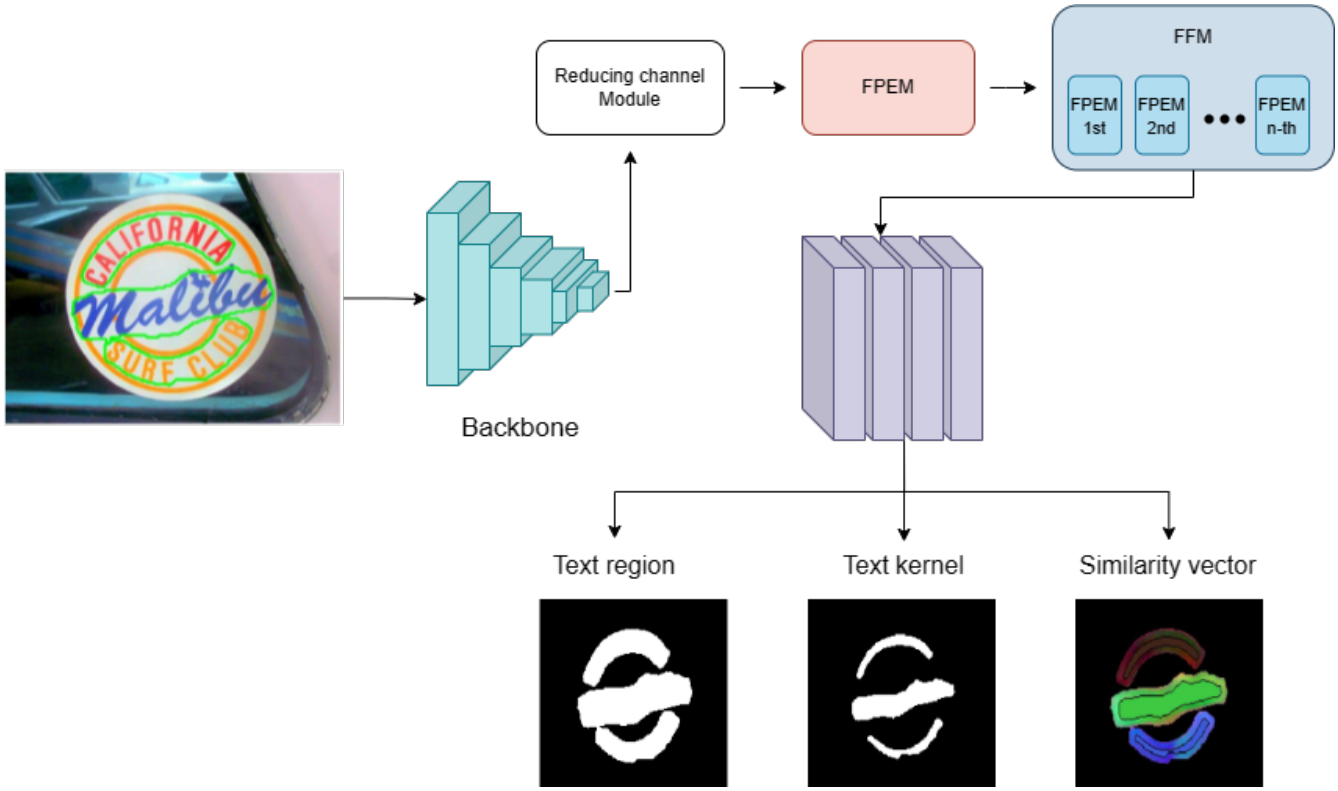


Figure 3.7: The architecture of PAN model.

characterized by their minimal computational overhead, rendering them suitable for integration behind the backbone network. This integration enriches the depth and semantic richness of features across diverse scales. Following each iteration of this enhancement process, an upgraded feature pyramid is generated, leading to n_c augmented feature pyramids: $F1, F2, ..., Fn_c$.

Subsequently, the Feature Fusion Module (FFM) is engaged to amalgamate the feature outputs from the FPEMs situated at varying depths. The outcome of this fusion process culminates in a final segmentation-oriented composite. PAN undertakes the task of text region prediction, enabling a comprehensive depiction of the contours of text instances. Additionally, the model predicts kernels that facilitate the differentiation of distinct text instances. The network anticipates a similarity vector for each text pixel to establish a coherent association between text pixels and kernels from the same text instance. The objective is to minimize the dissimilarity between the similarity vectors of a pixel and its corresponding kernel within the same text instance.

A straightforward and efficient post-processing algorithm is administered to derive the ultimate text instances from the model's predictions.

3.3.2 Reducing Channel Block

Four feature maps are engendered upon subjecting the input image to the ResNet [41] backbone, each distinct in scale. Typically, these feature maps exhibit augmented

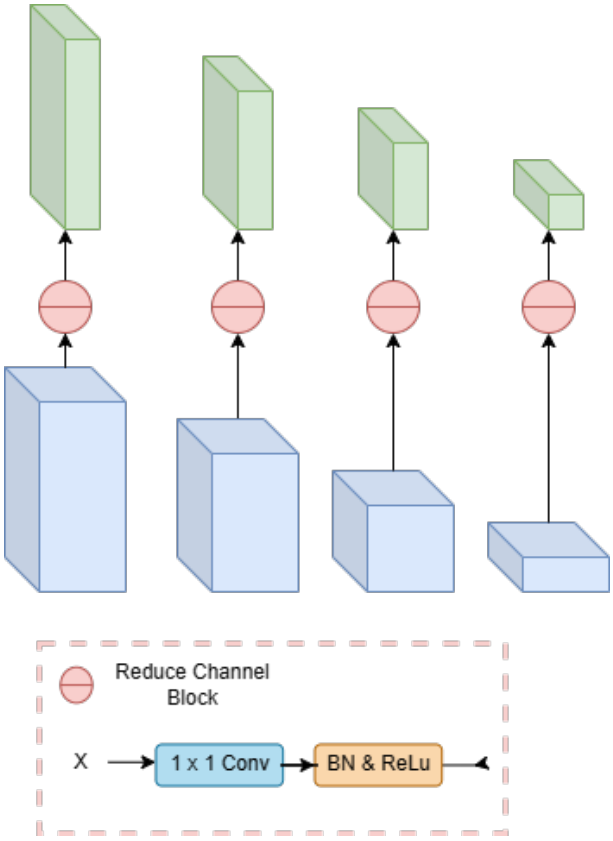


Figure 3.8: The reducing channel block in PAN model.

dimensions with the network's profundity. This phenomenon can precipitate considerable parameter escalation and computational requisites, mainly when larger filter dimensions are employed. To solve this problem, the model uses a 1×1 convolutional layer, concomitant with batch normalization and a rectified linear unit (ReLU) layer. This composite configuration facilitates channel-wise pooling, effectively mitigating the potential performance overhead. Moreover, it substantiates the expeditiousness of both model training and inference; the block's architecture has shown in Figure 3.8.

3.3.3 Feature Pyramid Enhancement Module

The architectural configuration of the Feature Pyramid Enhancement Module (FPEM) resembles the well-recognized U-Net architecture [7], as visually depicted in Figure 3.9. The module encompasses two phases, each contributing to its functionality: up-scale and down-scale enhancement. Within the up-scale enhancement stage, the module iteratively refines the feature maps of the input feature pyramid, progressively augmenting their quality with strides of 32, 16, 8, and 4 pixels, respectively. Conversely, the down-scale enhancement phase operates in reverse, utilizing the feature pyramid engendered by the up-scale enhancement stage as input. Enhancement is executed, proceeding from a stride of 4 pixels to that of 32 pixels. The down-scale enhancement phase bestows upon us the definitive output feature pyramid generated by FPEM.

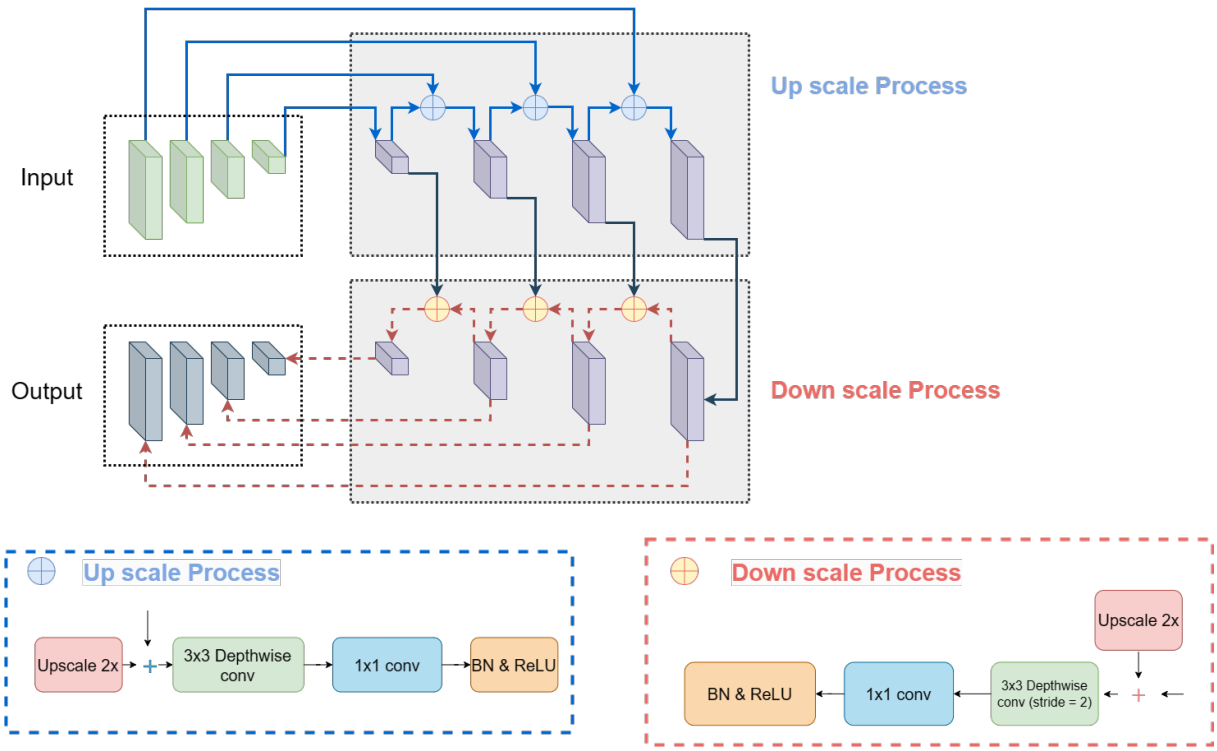


Figure 3.9: The architecture of Feature Pyramid Enhancement Module.

To engineer the fusion point within FPEM, the model employs a 3×3 depthwise convolution, followed by a 1×1 projection, as a departure from the conventional convolutional mechanism. This strategic choice accentuates the efficiency of FPEM by expanding the receptive field. This configuration mitigates computational overhead while optimizing receptive fields and network depth augmentation. Notably, the versatile character of FPEM is underscored by its cascadable nature. Increasing the cascade number, denoted as NC, consequently accentuates the efficacy of feature fusion across disparate scales, concomitantly augmenting the expansiveness of receptive fields intrinsic to the features.

In summary, the Feature Pyramid Enhancement Module (FPEM) entails an architecture similar to the U-Net, underpinning its dual-phase operation: up-scale and down-scale enhancement. The cascading of its cascadable nature empowers the module's adaptability and feature fusion capabilities. FPEM enriches the network's depth and receptive field by adopting specialized convolutional mechanisms, concomitantly minimizing computational intricacies. These attributes collectively contribute to FPEM's prominence as a pivotal component, facilitating advanced feature aggregation and enhancing a receptive lot of the network's features.

3.3.4 Feature Fusion Module

Feature fusion represents an essential stage in exploring the insights embedded within feature maps. This entails merging feature pyramids characterized by distinct depths, thus harnessing low-level and high-level semantic particulars. A straightforward and efficient

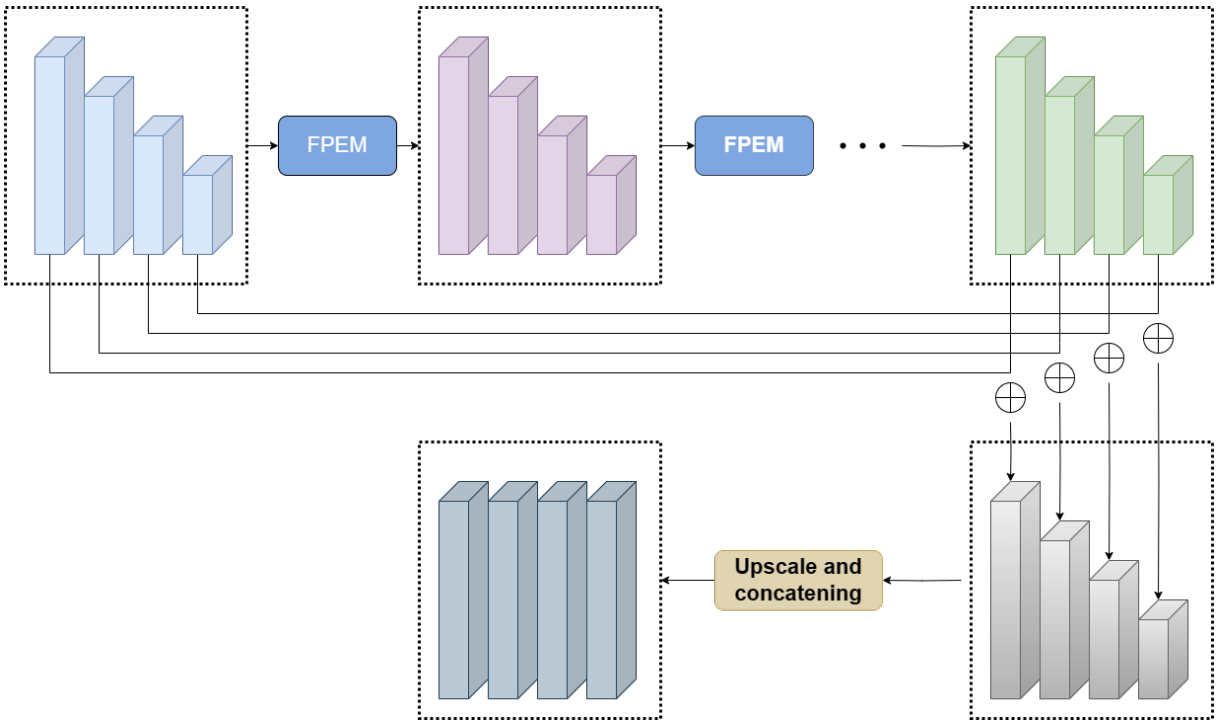


Figure 3.10: The architecture of Feature Fusion Module.

strategy for integrating these divergent feature pyramids involves their upsampling and subsequent concatenation.

In the initial step, the model executes an element-wise summation between the feature maps of corresponding scales extracted from the initial and terminal outputs of the Feature Pyramid Enhancement Modules (FPEMs). The resultant feature maps, arising from this additive operation, undergo an upsampling process and subsequently converge into a culminating feature map marked by a mere 4×128 channels.

This technique strategically serves the dual objectives of preserving information post traversal through multiple layers of the model while concurrently sustaining expeditiousness in training and operational phases when applied on computational devices.

3.3.5 Pixel Aggregation

To address text detection challenges, considering both text regions and kernels is very important. Text regions are instrumental in accurately representing text instances' complete shape. However, the text regions may exhibit overlapping characteristics when dealing with proximate text instances. This inherent overlap necessitates a mechanism to differentiate and demarcate distinct text instances effectively.

In sharp contrast, the utilization of kernels offers a discerning advantage. Text instances can be uniquely delineated through kernels, facilitating the differentiation process. Yet, it is crucial to acknowledge that kernels may not encapsulate the entirety of a text instance's spatial extent. Consequently, a vital augmentation step becomes indispensable, the fusion

of pixels residing within text regions into kernels. This process contributes to the holistic reconstruction of complete text instances.

To accomplish this, the model applies an adaptive algorithm termed Pixel Aggregation (PA), characterized by its learnable attributes. The primary function of PA is to judiciously steer text pixels toward their respective kernels in a guided manner. This algorithm's fundamental principle resonates with clustering methodologies. In this case, text instances can be likened to clusters, wherein the kernels are the same as the centers of these clusters, and text pixels represent the samples to be categorized within the clusters. In this paradigm, the optimization objective centers around minimizing the distance between text pixels and the kernels corresponding to the same text instance. This proximity-based rationale fuels the aggregation process, channeling text pixels toward their designated kernels, ultimately unifying pixel clusters into coherent kernels.

Inherent to aggregating text pixels with their corresponding kernels is minimizing the distance between the text pixel and the kernel of the identical text instance. This proximity-based criterion serves as the foundational principle that guides the aggregation process. During the training phase, this principle materializes through an aggregation loss, denoted as L_{agg} and expressed through Equation 3.1 3.2. This loss function effectively encapsulates the essence of the rule underpinning pixel-to-kernel aggregation.

$$L_{agg} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{|T_i|}\sum_{p\in T_i} ln(\mathcal{D}(p, K_i) + 1) \tag{3.1}$$

with

$$\mathcal{D}(p, K_i) = max(||\mathcal{F}(p) - \mathcal{G}(K_i)|| - \theta_{agg}, 0)^2 \tag{3.2}$$

where the T_i is the i_{th} text instance. The N is the number of text instances. $\mathcal{D}(p, K_i)$ defines the distance between text pixel p and the kernel K_i of text instance T_i. θ_{agg} is a constant, which is set to 0.5. $F(p)$ is the similarity vector of the pixel p. $\mathcal{G}(\cdot)$ is the similarity vector of the kernel K_i, which can be calculated by $\sum_{q\in K_i}\mathcal{F}(q)/|K_i|$.

Moreover, ensuring that the kernels remain distinctly discernible in cluster center differentiation is imperative. This mandates that the kernels associated with disparate text instances exhibit substantial separation. A discrimination loss, denoted as L_{dis} and delineated by Equation 3.3, is harnessed during the training phase to operationalize this criterion. This loss function encapsulates the essence of maintaining adequate kernel separation, ensuring the preservation of distinctiveness among kernels belonging to different text instances.

$$L_{dis} = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j=1}^{N} ln(\mathcal{D}(K_i, K_j) + 1) \tag{3.3}$$

with

$$\mathcal{D}(K_i, K_j) = max(\theta_{dis} - ||\mathcal{G}(K_i) - \mathcal{G}(K_j)||, 0)^2 \tag{3.4}$$

where θ_{dis} which is set to 3 in all our experiments.

The testing phase uses the predicted similarity vector as a guiding mechanism. This vector is a navigational tool to direct the alignment of pixels within the text area's confines, ultimately facilitating their convergence toward the pertinent kernel. This strategic orchestration effectively realizes the pixel-to-kernel alignment process, an integral step within the overall text instance detection framework.

3.3.6 Loss function

The formulation of our PAN model loss function can be expressed as follows:

$$\mathcal{L} = \mathcal{L}_{tex} + \alpha \mathcal{L}_{ker} + \beta(\mathcal{L}_{agg} + \mathcal{L}_{dis}) \tag{3.5}$$

$$\mathcal{L}_{tex} = 1 - \frac{2\sum_i P_{tex}(i) G_{tex}(i)}{\sum_i P_{tex}(i)^2 + \sum_i G_{tex}(i)^2} \tag{3.6}$$

$$\mathcal{L}_{ker} = 1 - \frac{2\sum_i P_{ker}(i) G_{ker}(i)}{\sum_i P_{ker}(i)^2 + \sum_i G_{ker}(i)^2} \tag{3.7}$$

where $P(\cdot)$ and $G(\cdot)$ refer to the value of pixel result and ground truth of the text region respectively. We set $\alpha = 0.5$ and $\beta = 0.25$ in all experiments.

3.4 Focus branch

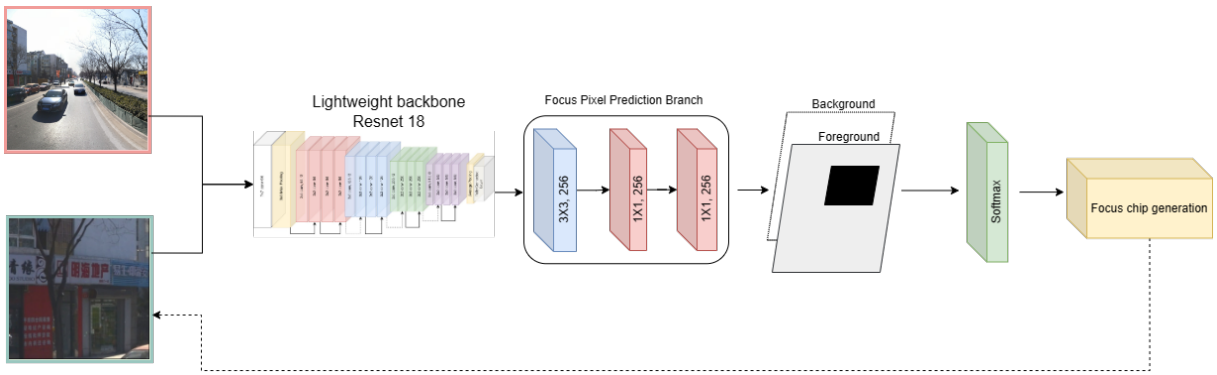
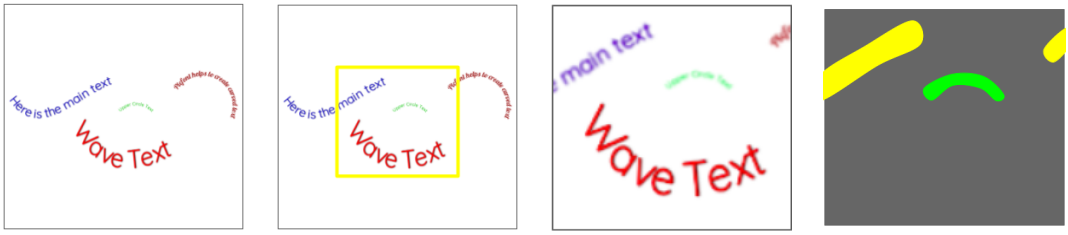


Figure 3.11: The architecture of Focus branch.

To mimic the foveal system of humans, we adopt the autofocus framework [46] with some modifications. The process aims to predict interesting regions in the image that may contain text information and discard remaining background regions that are unlikely to have text information at the following scale. The module zooms and crops from interesting regions when applying the detector at successive scales. Focus branch comprises three main parts: the first predicts FocusPixels through learning; the second produces FocusChips; and the third, FocusCombining, combines segmentation regions from various scales. Since the first component is the only one used in the training phase, all three will be used in the inference phase.

3.4.1 Focus Pixels Finding



(a) (b) (c) (d)

Figure 3.12: All pixels that a particular text instance covers are regarded as Focus pixels if their sizes fall within a certain range and do not deviate too much from the previous scale. When training, it is best to ignore some cases where the area is slightly higher than the upper bound or lower than the lower bound of the area range as mentioned. (a) Previous scale. (b) Chip for the next scale. (c) Current scale. (d) Ground truth for focus map of current scale, where regions with green color are positives (equal 1), yellow color is negatives (equal -1), gray color is background (equal 0)

Focus Pixels are established at the convolutional feature network's granularity from the output of the lightweight backbone. Simple modules process the feature map from the lightweight backbone and produce the focus map, which is then trained to match the interesting ground truth map accordingly. If a feature map pixel overlaps with a small-sized text instance, it is referred to as a focus pixel. In the resized chip, which is input to the network, a text instance is deemed small if the root square value of the text area within the image samples falls within a range (between 3 and 50 in our implementation). Note that in some cases, specific text can be split when generating a chip, and the remaining split text in the chip may not cover enough information of a text instance, so if the area text in the chip is higher, 80% are of the original one, we consider all pixels it covers as focus pixels. In the training process, we mark Focus Pixels as positives. Some pixels that overlap text have a square root of area value less than or slightly more significant than the range we defined for positive pixels (less than five or between 50 and 100). All of those pixels are regarded as negatives. They should be ignored during training since

the network does not have sufficient information to make correct predictions about them at that particular scale. We regarded all remaining pixels as background, has shown in Figure 3.12.

Specifically, given an image of size $W \times H$, and the whole backbone with stride is s, the labels L will be of size $W' \times H'$, where $W' = \left\lceil \frac{W}{s} \right\rceil$ and $H' = \left\lceil \frac{H}{s} \right\rceil$. Since the stride is s , the rach label $l \in L$ corresponds to $s \times s$ pixels in the image. The label l is defined as follows,

$$l = \begin{cases} 1, & IoU(GT, l) > 0, a < \sqrt{GTArea} < b \\ -1, & IoU(GT, l) > 0, \sqrt{GTArea} < a \\ -1, & IoU(GT, l) > 0, b < \sqrt{GTArea} < c \\ 0, & otherwise \end{cases} \tag{3.8}$$

where IoU is intersection over union of the $s \times s$ label block with the groundtruth boundary of text instance. $GTArea$ is the area of the ground truth boundary after scaling. a, b, c are parameters in order to determine which text instance is valid (all pixels considered as 1s), ignored (all pixels considered as -1s), or background (all pixels considered as 0s). a is typically 3, b is 50, c is 100.

For training the network, we use simple modules consisting of two common convolutional layers with a ReLU non-linearity on top of the conv 5 feature map. Ultimately, we use the sigmoid function to classify Focus Pixels as shown in Figure 3.13.

$$F_{conv5}, F_{conv6}, F_{conv7}, F_{conv8} = F_b(I_n) \tag{3.9}$$

$$P = Sigmoid(F_{Focus}(F_{conv5})) \tag{3.10}$$

Given output focus map P with size $W' \times H'$ the loss function is calculated for every pixel as binary classification if the pixel is not ignored,

$$\mathcal{L}_{Focus} = -\sum_{i}^{W'} \sum_{j}^{H'} \sum_{c}^{C} t_{i,j,c} k_{i,j} log(p_{i,j,c}) / \sum_{i}^{W'} \sum_{j}^{H'} k_{i,j}$$

where

$k_{i,j} = 0$ if pixel at position i, j of groundtruth focus map is ignored; $k_{i,j} = 1$ otherwise. $c \in (0, 1)$. $t_{i,j,c} = 1$ if pixel at position i, j of grountruth focus map is c; $t_{i,j,c} = 0$ otherwise. $p_{i,j,c}$ is probability prediction of pixel i, j classified as c.

3.4.2 Focus Chips Generation

Focus branching is used during inference to produce focus maps P, which are then used to predict which pixels in the focus maps will be in the foreground t and have some connected components S. With a size $d \times d$ filter, we dilate every part before merging it. Then, chips were created to contain these connected components. If two chips overlap, the two chips are combined, and the overlapped chips are replaced with the boundary regions surrounding them. Be aware that sometimes, even after dilation, the number of connected components is huge, but each is very small in size. The batch inference is inefficient and takes a long time because so many small chips are produced. Therefore, we eliminate chips whose width or height is below the minimum size k. The process is described in Algorithm 1.

Algorithm 1: Focus Chips Generation

Input: Focus map P, threshold t, dilation constant d, minimum size k
Output: Chips C

1 Transform P into the binary map using threshold t
2 Dilate binary map with a dxd filter
3 Obtain the list of connected components S
4 Generate enclosing chips C for each component in S if the component size is larger than k
5 If chips C overlap, merge these chips
6 **return** Chips C

3.4.3 Focus Combination for final results

An issue encountered in multi-scale inference for text instance detection pertains to situations where a text instance assumes considerable size and elongation at one scale. The consequent creation of chips for the subsequent scale might inadvertently truncate the text instance present in the prior scale. While an alternative detection option could be more accurate, its adoption could result in an elevated count of false positives. This arises from the possibility of the exact text being detected independently in two different scales, yielding two distinct text outputs.

Addressing this complexity necessitates a more nuanced approach than simply selecting the superior outcome or applying the conventional Non-Maximum Suppression (NMS) algorithm to eliminate instances with overlaps. This is attributed to the fact that multiple detections might pertain to the same text instance. As such, the filtered outcome could potentially carry insights about the instance that remain elusive in other detections.

As depicted in Figure 3.13, the outputs from the model do not inherently offer bounding box information akin to prevalent detection models. Instead, integrating region maps and

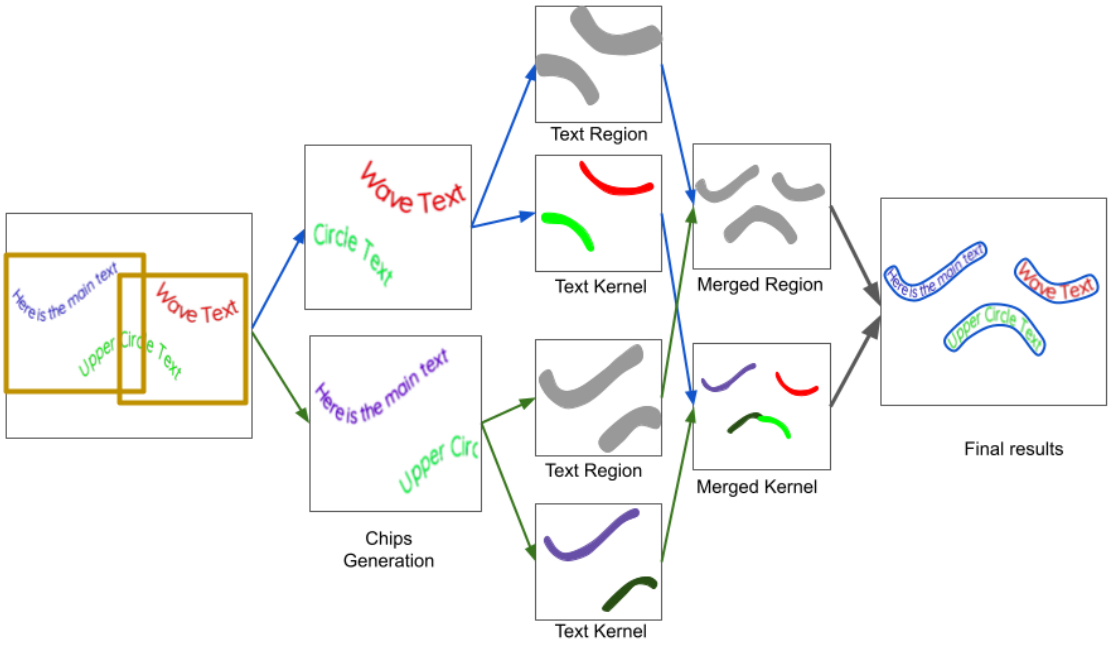


Figure 3.13: Description of focus branch process

kernel maps culminates in generating ultimate results. In contrast to the conventional employment of the NMS technique, a distinctive strategy is adopted wherein all regions and kernels originating from each scale are unified to generate the final regions and kernels.

3.5 Implementing TextFocus

The comprehensive TextFocus pipeline is outlined in Figure 3.14, involving the amalgamation of two distinct branches, each oriented towards specific tasks. Commencing with preprocessing, the input sample characterized by dimensions "W x H" is directed to the lightweight backbone ResNet-18. This stage yields an array of feature maps $[conv5, conv6, conv7, conv8]$ possessing dimensions of $[W/8 \times H/8, W/16 \times H/16, \frac{W}{32} \times \frac{H}{32}, \frac{W}{64} \times \frac{H}{64}]$, respectively. Notably, among these feature maps, the one emanating from conv5 is chosen due to its suitability for defining Focus Pixels. The rationale behind this selection arises from the fact that other feature maps are comparably diminutive in size. This strategic choice offers the advantage of expediting the extraction of focus maps, which subsequently facilitates their utilization during later inference stages.

In the context of the focus branch, a pair of convolutional layers $(3 \times 3$ and $1 \times 1)$ augmented with ReLU non-linearity is incorporated. This design choice channels the feature map emerging from conv5 through these uncomplicated modules. Subsequently, a sigmoid classifier is introduced to predict the presence of Focus Pixels. During the inference phase, the focus maps denoted as P stemming from this branch play a pivotal role in the generation of chips.

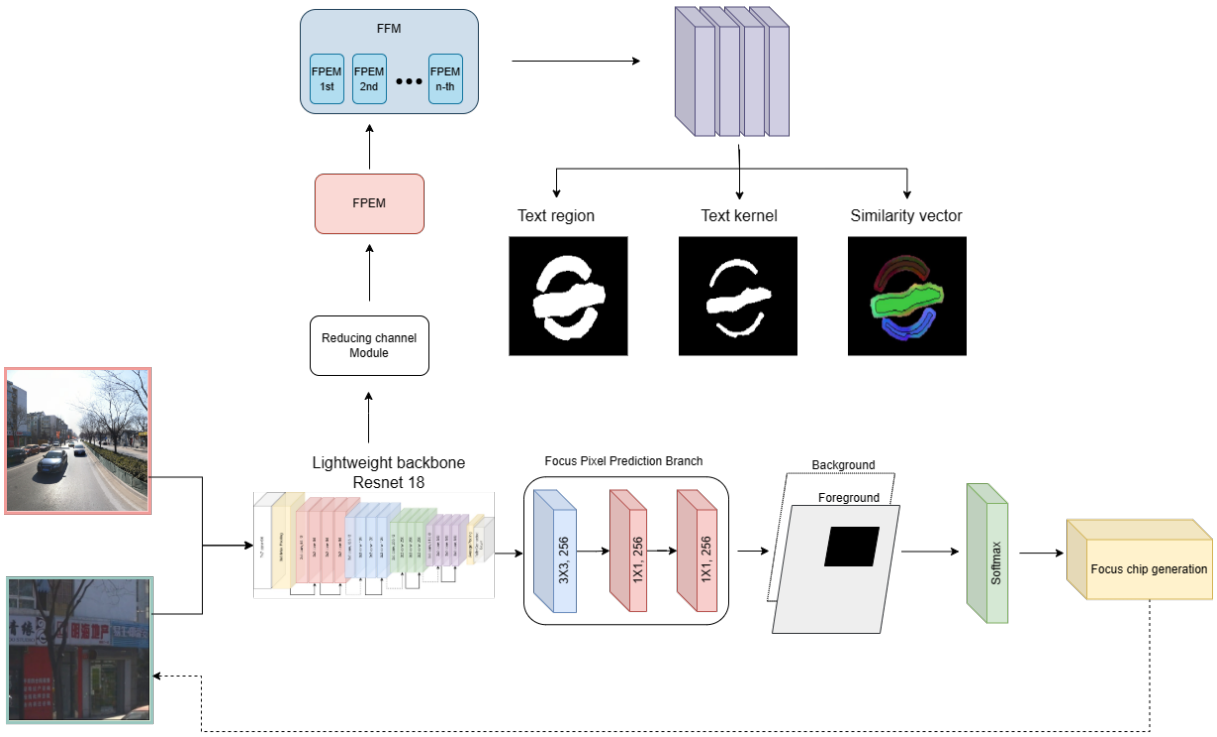


Figure 3.14: The complete architecture of TextFocus

All feature maps from the backbone output for the detection branch are passed through a common 1×1 convolutional layer to obtain a list of feature maps with the same number of channels; in this case, the number of channels is set to 128. Then, to enhance the knowledge contained in feature maps, two cascade modules are combined and used by the FFM module. The outputs of the Text Region, Text Kernel, and Similarity Vector are used in the training phase and produce text instances in the inference phase.

The detection branch processes all feature maps obtained from the backbone output. These feature maps undergo a uniform transformation via a 1×1 convolutional layer, generating a series of feature maps characterized by a consistent channel count of 128. The subsequent step involves the integration of two cascade modules, which are subsequently harnessed by the Feature Fusion Module (FFM) to augment the insights encapsulated within the feature maps. During the training phase, the outcomes stemming from the Text Region, Text Kernel, and Similarity Vector components are utilized, while in the inference phase, these components contribute to generating text instances collaboratively. Our loss function can be formulated as:

$$\mathcal{L} = \mathcal{L}_{tex} + \alpha\mathcal{L}_{ker} + \beta(\mathcal{L}_{agg} + \mathcal{L}_{dis}) + \gamma\mathcal{L}_{Focus} \tag{3.11}$$

where $\mathcal{L}_{tex}$ is the loss of text regions and $\mathcal{L}_{ker}$ is the loss of the kernels. The α, β, γ are used as loss weights to balance the importance of each loss, and we set them to 0.5, 0.25

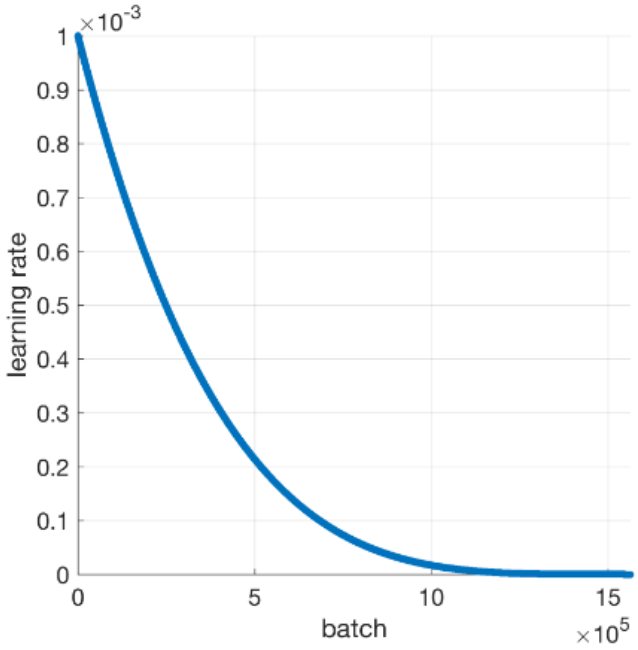


Figure 3.15: The result of Polynomial Learning Rate Scheduler

and 0.5 respectively in all experiments.

We employed the extensive SynthText dataset [48], which comprises approximately 800,000 synthetic images, to demonstrate the substantial improvement achievable in metric outcomes by pretraining text models on this dataset before their subsequent application to the target dataset. Consequently, our approach involves pretraining the SynthText model and transferring the learned weights to the target dataset. For the training phase involving SynthText, we opted for the Adam optimizer algorithm and employed a learning rate of 0.01. A learning rate of 0.001 was adopted for training after transitioning to the target dataset. We incorporated the Polynomial LR Scheduler during training to govern the learning rate progression, as depicted in Figure 3.15.

Throughout the training process, we incorporated a set of straightforward augmentation techniques. These techniques encompassed Random Scale, Horizontal Flip, Random Rotate and Random Crop Padding. This strategic augmentation approach ensured uniform input dimensions for the model. Notably, before being fed into the model, the inputs underwent normalization.

In this chapter, we presented the method TextFocus with a Multi-resolution approach. We will demonstrate the superior efficiency of this method by experimental results in the next chapter.

CHAPTER 4. EXPERIMENTAL RESULTS AND DISCUSSION

4.1 Datasets

The research landscape has witnessed the compilation of numerous datasets aimed at detecting and recognizing text within natural images, significantly propelling the progress of contemporary methodologies. These datasets can be categorized into real-world text datasets [2][1][42][43]and synthetic text datasets [48] [49]. Predominantly featuring Internet images and Google Street View images, the images constituting these datasets encapsulate diverse real-world scenarios.

4.1.1 Chinese Text in the Wild (CTW datasets)

The Chinese Text in the Wild dataset (CTW) [2] constitutes an expansive collection of images meticulously curated to encompass various scenarios wherein Chinese text becomes apparent within uncontrolled, real-world settings. Annotated by experts, CTW features approximately 1 million Chinese characters, stemming from 3,850 distinct ones, across over 30,000 high-resolution street view images. This dataset encapsulates instances of text seamlessly integrated within authentic contexts, encompassing elements like signs, posters, labels, and advertisements. With its deliberate focus on capturing the nuances of Chinese script in diverse contextual landscapes, CTW emerges as a noteworthy resource for propelling advancements in text detection and recognition.

4.1.2 SCUT-CTW1500

The CTW1500 dataset [1] is a specialized image compilation explicitly designed for curved text detection. It encompasses an assemblage of approximately 1,500 images featuring 10,751 bounding boxes. Each image encapsulates instances of curved text found within real-world scenes. Notably, the dataset is meticulously annotated, with each bounding box delineating the precise regions of curved text. Given its meticulous annotation and specialized focus, the CTW1500 dataset is a substantial asset for researchers and practitioners embarking on addressing the intricate complexities and challenges associated with the detection and recognition of curved text within natural environments.

4.1.3 ICDAR15

The International Conference on Document Analysis and Recognition (ICDAR) [42] has assumed a substantial role in orchestrating text detection competitions and fostering the advancement of datasets and algorithms. The ICDAR15 dataset was unveiled in the context of the ICDAR15 Robust Reading Competition, designed to address incidental scene text detection. This dataset encompasses 1,000 training and 500 testing images featuring English text instances.

4.1.4 TotalText

The TotalText dataset [43] constitutes an extensive assemblage of images meticulously compiled to advance scene text detection. This dataset, recently introduced during the ICDAR17 event, comprises 1255 training and 300 testing images. It comprehensively captures diverse instances of text seamlessly integrated within natural scenes, encompassing an array of real-world contexts such as signage, posters, banners, and advertisements. Notably, the dataset stands out due to its comprehensive annotations, which meticulously delineate the text instances' word-level and character-level regions. With its distinctive attributes and comprehensive content, the TotalText dataset emerges as a valuable asset, poised to empower researchers and developers to enhance the efficacy and robustness of scene text detection algorithms, particularly within intricate and varied environmental contexts.

4.1.5 SynthText

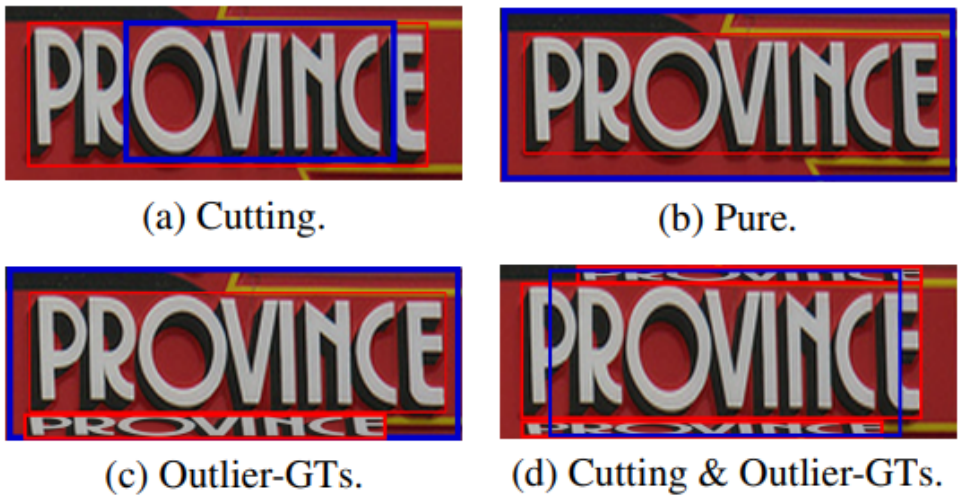
SynthText [48] is a large-scale dataset with around 800,000 synthetic images. These images are created by blending natural images with randomly rendered text. Verisimilarity Image Synthesis Dataset(VISD) [49] contains 10,000 images synthesized with 10,000 background images. Thus, there are no repeated background images for this dataset. Furthermore, we have used the SynthText dataset as a pre-train for all data in our study.

In the context of this study, the evaluation encompasses a comprehensive amount of datasets, where pre-existing annotations of arbitrary shapes characterize the Scut-CTW1500, ICDAR15, and Total Text datasets. In contrast, the Large CTW dataset has novel annotations generated by applying our algorithms. Further diversifying the testing spectrum, the Synth Text dataset is explicitly employed to establish pre-trained models. We use the test datasets above, similar to the previous research and the baseline PAN model. The detail information of datasets as shown in Table 4.1.

The charts below depict the size of the object in the image A. We can see that the area of the things occupies about 10 - 30% of the entire image area. Through this, we can see that the train and test datasets are both high-quality images, and the text has a small space, occupying a small amount compared to the background.

Datasets	Range of resolution(pixel)	Real/synthetic	Annotation level	Language in image
Large CTW [2]	320x240 - 3840x3200	real	word/line	Chinese
Total Text [43]	640x480 - 1920x1080	real	word/line	English
ICDAR15 [42]	300x300 - 2400x2400	real	word/line	Various
Scut-CTW1500 [1]	640x480 - 1920x1080	real	word/line	Chinese
Synth Text [48]	320x240 - 1920x1080	synthetic	word/line	Various

Table 4.1: The detail information of datasets.



(a) Cutting. (b) Pure.

(c) Outlier-GTs. (d) Cutting & Outlier-GTs.

Figure 4.1: Unreasonable cases obtained using recent evaluation metrics. (a), (b), (c), and (d) all have the same IoU of 0.66 against the GT. Red: GT. Blue: detection. [12]

4.2 Evaluation metrics

Given that the primary function of text region detection is to facilitate text recognition, it becomes imperative for bounding boxes representing text regions to accurately encapsulate the entirety of text information while avoiding interference with other instances of text. However, famous evaluation metrics such as the IOU-metric lack consideration for the consequences of segmented ground truth (GT) regions and aberrant GT instances, as shown in Figure 1. Moreover, discerning the adequacy of detection tightness presents challenges. Consequently, the comprehensive representation of the strengths of detection methods still needs to be completed.

The "Tightness-aware Intersection-over-Union (TIoU) Metric" [12] refers to a specialized evaluation measure employed in the context of object detection or localization tasks. Unlike the traditional Intersection-over-Union (IoU) metric, which computes the ratio of the overlapping area to the union of two bounding boxes, the TIoU metric introduces an element of tightness sensitivity. This means that it considers the relative tightness or compactness of the bounding boxes, providing a more nuanced assessment of the spatial alignment between predicted and ground truth object boundaries. The TIoU metric is particularly valuable when dealing with objects of varying sizes and aspect ratios, allowing for a refined evaluation of detection accuracy. This metric is often employed in evaluating object detection models to provide insights into their performance in terms of spatial localization precision while considering the inherent diversity in object shapes and sizes.

4.2.1 TIoU-Recall

At an intuitive level, the scenario where a ground truth (GT) rectangle G_i is intersected by a detection bounding box D_j can lead to erroneous recognition outcomes. While conventional Intersection-over-Union (IoU) metrics are capable of gauging the extent of overlap between G_i and D_j in terms of tightness, they do not possess the capability to assess these situations in a goal-oriented manner, as depicted in Figure 4.1 (a) and (b). In this example, the detections in both (a) and (b) share the same IoU value (0.66) with the ground truth, despite the former failing to encompass a few characters of the GT. To rectify this limitation, the cutting phenomenon can be addressed by introducing a penalty proportional to the intersection area within the GT.

$$C_t = A\left(G_i\right) - A\left(D_j \cap G_i\right), C_t \in \left[0, A\left(G_i\right)\right], \tag{4.1}$$

where $A(\cdot)$ means the area of the region.

The proportion of intersection in G_i is given by:

$$\left(C_t\right) = 1 - x, x = \frac{C_t}{A\left(G_i\right)} \tag{4.2}$$

The final TIoU-Recall is defined as follows:

$$TIoU_{\text{Recall}} = \frac{A\left(G_i \cap D_j\right) * f\left(C_t\right)}{A\left(G_i \cup D_j\right)}. \tag{4.3}$$

Equation 4.3 presents a straightforward yet highly effective approach to addressing the issue of cutting behavior. For instance, in Fig4.1 (a) and (b), the TIoU-Recall values are computed as 0.424 and 0.66, respectively. This reveals that the omission of characters leads to a substantial reduction of 35.8% in the recall performance. This empirical result underscores the significance of the solution in quantifying the impact of cutting behavior on the overall performance of the detection algorithm.

4.2.2 TIoU-Precision

Conversely, a single detection covering multiple ground truths (GTs) can introduce complexities in recognition outcomes. This challenge emerges from the recognition methods' difficulty distinguishing which text instance constitutes the intended target GT, as exemplified in Fig.4.1 (c). To address this issue, a solution involves imposing penalties on such detections to encourage compactness, thus minimizing the influence of outlier GTs. It is worth noting, however, that in cases where outlier GTs are positioned within the target GT region, even an ideal detection bounding box might inadvertently

encompass these outliers. Hence, only outlier GT regions inside the detection bounding box but external to the target GT region are penalized. The union area (Ot) computation pertaining to all eligible outlier GTs is undertaken using Equation .

$$\begin{aligned} O_{t_{ij}} =& A\left(\left(G_1 \cap D_j - G_1 \cap D_j \cap G_i\right) \cup \right. \\ & \ldots \cup \left(G_{i-1} \cap D_j - G_{i-1} \cap D_j \cap G_i\right) \cup \\ & \left(G_{i+1} \cap D_j - G_{i+1} \cap D_j \cap G_i\right) \cup \ldots \cup \\ & \left.\left(G_n \cap D_j - G_n \cap D_j \cap G_i\right)\right), \\ & O_{t_{ij}} \in \left[0, A\left(D_j - D_j \cap G_i\right)\right]. \end{aligned} \tag{4.4}$$

It's worth mentioning that, in cases where each ground truth instance G_n (where $n \neq i$) doesn't intersect with the detection bounding box D_j, it can be straightforwardly disregarded. This selective approach contributes to enhanced computational efficiency. Subsequently, the ratio of intersection within D_j is determined by the following equation:

$$\left(O_t\right) = 1 - x, x = \frac{O_t}{A\left(D_j\right)} \tag{4.5}$$

The TIoU-Precision is defined as follows:

$$TIoU_{\text{Precision}} = \frac{A\left(D_j \cap G_i\right) * f\left(O_t\right)}{A\left(D_j \cup G_i\right)} \tag{4.6}$$

4.2.3 Tightness-aware Metric

The harmonic mean of recall and precision is usually adopted as the primary metric:

$$\text{Hmean} = 2\frac{\text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}} \tag{4.7}$$

where recall and precision are calculated by:

$$\begin{aligned} \text{Recall}_{\text{ ori}} &= \frac{\sum \text{Match}_{gt_i}}{\text{Num}_{gt}}, \\ \text{Precision}_{\text{ ori}} &= \frac{\sum \text{Match}_{dt_j}}{\text{Num}_{dt}}. \end{aligned} \tag{4.8}$$

4.3 Implementation detail

We have put in significant effort and thoroughness in generating data, implementing the TextFocus strategy, training, testing, and analyzing and visualizing results. In this section, we will provide detailed information about the implementation process.

Frameworks and libraries: Our TextFocus strategy is coded using Python programming language and the PyTorch framework. With PyTorch, we can seamlessly read, preprocess, and feed the training data and easily implement the TextFocus strategy during training and inference. PyTorch also includes a Tensorboard tool, which enables users to conveniently track training progress and view training loss, testing loss, and evaluation scores during training. Apart from PyTorch, we utilize several other built-in Python libraries such as OpenCV for image processing, Numpy for matrix computation, Pandas and Matplotlib for result analysis, and other auxiliary libraries such as openmim, mmcv==1.3.1, shapely, pyclipper, editdistance.

Environment: Regarding the environment, for implementation, debugging, data analysis, and data synthesis, we utilize personal computers with the following configuration: Intel Core i7-9300H CPU, 8GB of RAM, and NVIDIA GeForce GTX 1650 Ti GPU with 4GB of VRAM. However, for training, we rely on Google Colab's and Kaggle virtual machines equipped with an Intel Xeon CPU and 40GB of RAM, paired with NVIDIA P100 GPU boasting 40GB VRAM or GPU T4 x2 boasting 30GB VRAM.

Code: We reused a pre-existing codebase from the PAN model [27] [1] and AutoFocus[46] [2] to build our own model. In addition, we developed all of the source code ourselves, including data generation, data loading, training-testing-inference procedures, and visualization.

Hyperparameters: Table 4.2 provides a detailed breakdown of the hyperparameters used during our training process. While most of the parameters remain the same as in the baseline, there are a few changes that we made:

Each dataset contains a predetermined number of training and testing data. Because the attribution data for each dataset varies greatly, since virtually all of the dataset's texts are pretty little, we set the input size for Large CTW to (640×640). We set the input size for other datasets to (320×320) because those texts are large enough to be detectable. Then train the model with a learning rate of 0.001, batch size 32, and num epochs 150. We adopt the Adam optimizer with the same hyper-parameters and use the Polynomial LR Scheduler strategy to adjust the learning rate during training in Table 4.2.

We must choose the ground truth appropriate for each chip produced for the focus branch. As shown in Table 4.3, a text instance is considered positive if the square rootof the area is small enough (greater than a and less than b) or damaging if the room is too large (larger than c). If not, we ignore it because it will confuse the model whenit is being trained.

[1] `https://github.com/WenmuZhou/PAN.pytorch`
[2] `https://github.com/mahyarnajibi/SNIPER`

Parameter	Large CTW Dataset	Other Datasets
Input shape	640,640	320,320
base lr	0.001	0.001
batch size	32	32
num epoch	150	150
optimizer	Adam	Adam
lr scheduler	Polynomial lr	Polynomial lr

Table 4.2: Overall training parameters.

Parameter	Value
do not care low - a	3
small threshold - b	50
do not care high - c	200

Table 4.3: Parameters for groundtruth selection when generate chip for focus branch training.

4.4 Results and analysis

We execute and get results from all experiments with an NVIDIA Tesla P100 GPU 16Gb and one 2.00GHz CPU in a single thread. We evaluate TextFocus on the ICDAR2015 [42], Total-Text [43], SCUT-CTW1500 [1], and Large CTW [2]. Our model, with an input size of (320×320), obtained competitive results of 84.70 F1 Score on the ICDAR2015, 82.05 on the Total-Text, 84.90 on the SCUT-CTW1500 (Table. 4.4). Large CTW is a substantial Chinese text dataset in the wild with a very high-resolution image sample, text instances in the dataset are often tiny according to the overall image. With an input size of (640,640), TextFocus obtained a 61.1 F1 Score with acceptable real-time FPS in Table4.5.

Since the focus branch generates various sizes, we aggregate chips with the same

Method	ICDAR2015 [42]				Total-Text [43]				SCUT-CTW1500 [1]			
	P	R	F1	FPS	P	R	F1	FPS	P	R	F1	FPS
CTPN [50]	74.2	51.6	60.9	3.55	-	-	-	-	60.4	53.8	56.9	3.57
SegLink [51]	73.1	76.8	75.0	-	30.3	23.8	26.7	-	42.3	40.0	40.8	1.35
EAST [20]	83.6	73.5	78.2	-	50.0	36.2	42.0	-	78.7	49.1	60.4	2.52
RRPN [52]	82.0	73.0	77.0	-	-	-	-	-	-	-	-	-
PSENet [26]	84.5	**86.9**	**85.7**	0.8	78.0	**84.0**	80.9	1.95	79.7	84.8	82.2	0.9
TextSnake [53]	84.9	80.4	8.6	0.55	82.7	74.5	78.4	-	67.9	**85.3**	75.6	-
PAN [27]	84.0	81.9	82.9	**12.29**	**83.6**	78.5	80.1	10.11	**86.4**	81.2	83.7	13.11
Ours (320)	**86.1**	74.5	79.9	8.51	82.7	74.1	78.1	**11.13**	84.8	80.9	82.8	**14.21**
Ours (640)	84.3	85.1	84.7	1.92	82.6	81.5	**82.05**	2.12	84.4	83.8	**84.9**	2.45

Table 4.4: Results on ICDAR2015 [42], Total-Text [43], SCUT-CTW1500 [1]. ”P”, “R”, “F” and “FPS” represent the precision, recall, F-measure, and frame per second, respectively.

Method	Large CTW [H-7]			
	R	P	F	FPS
Ours (448)	54.6	52.3	53.4	5.12
Ours (640)	**62.1**	**60.1**	**61.1**	**1.71**

Table 4.5: Results on Large CTW [2]. "P", "R", "F" and "FPS" represent the precision, recall, F-measure, and frame per second, respectively.

size and aspect ratio to achieve a high batch inference throughput. When executing batch inference, we occasionally apply padding, which can slightly alter the number of pixels analyzed per image. Since the number of groups (for size and aspect ratio) can be increased without lowering the batch size, this overhead is negligible for large datasets.

Beyond just achieving efficient and balanced metrics results, TextFocus also aids in efficient training. Chips are generated sequentially for each scale to help input image size to the model always be the same to avoid performance from being impacted by small size input, which helps conserve resources and expedite training.

4.4.1 The effectiveness and influence of the backbone and Detection branch

We adopt PAN[27] as the detecting branch for our model, with ResNet-18[41] serving as the module's neural network. The detection branch of our TextFocus uses an anchor-free text detection strategy, directly getting the text region map and kernel region map combined to now segment the text instances in contrast to text detector models that find regions during the process feature to get the area of interest, through this, to get a bounding box of each text. Furthermore, ResNet-18 is a lightweight backbone; the output of the features cannot describe a specific region in the image sample; the cascaded strategy of the FPEM and FFM modules, which have a low computational cost, allows for a more profound and more expressive expression of features at various scales. The model is very effective and can be used in real-time thanks to its lightweight backbone, cascaded pipeline strategy, and segment-based text detection.

4.4.2 The effectiveness of Focus branch

Saving on resources and computation: Processing a super-resolution image sample for a lightweight backbone cannot learn all the detail in the image for features output, even with features boosting strategy. Additionally, processing high-resolution image samples requires a lot of time and resources. Resource usage and computation increase when the image size increases, as seen in Figures 4.6 and **??**. In our experiments, training the model with 320-pixel images can result in batches that are more than 1.5 times larger than those prepared with 640-pixel photos, taking a total of 10 hours for 150 epochs as opposed to 17 hours for training on an NVIDIA Tesla P100 GPU 16Gb with 640-pixel ideas. However, downscaling the image to process through the model will blur or remove a text instance if

Method	GPU memory (Gb per image)	GFlops per image	F-Score
CTPN	5.2	181.36	56.9
SegLink	2.8	103.09	40.8
EAST	3.1	116.84	60.4
PSENet	5.3	183.43	82.2
TextSnake	2.6	95.62	75.6
PAN	2.5	90.24	83.7
Ours (320)	**1.8**	**24.56**	82.8
Ours (640)	2.7	98.37	**84.9**

Table 4.6: GPU memory and GFlops per image of methods on SCUT-CTW1500[1]. The usage is calculated on the entire inference processes.

it is small, which will worsen performance. The focus branch will zoom in on the exciting regions that may have text instances so the input size of the model doesn't need to be high; each chip generated will be scaled to a specific resolution to process through the model at the following scale, this solve information missing problem of low-resolution and resource, computing, time consumption problem of high-resolution.

Real-time text detection: When resource consumption is low, processing image samples at all resolutions while maintaining an acceptable runtime pipeline is made possible by continuously zooming in the image sample. The Large CTW dataset [2] contains a high resolution of real-world Chinese text, each image sample having a maximum image size of (5000,5000), and most of the text instances are pretty small. Utilizing the focus branch, we train with just 60 epochs and archive results of 61.1 F1 Score with 1.71 FPS. Figure B.1 visualizes the inference pipeline of TextFocus on the Large CTW dataset, tiny text instances of complex images can be detected at higher scales effectively.

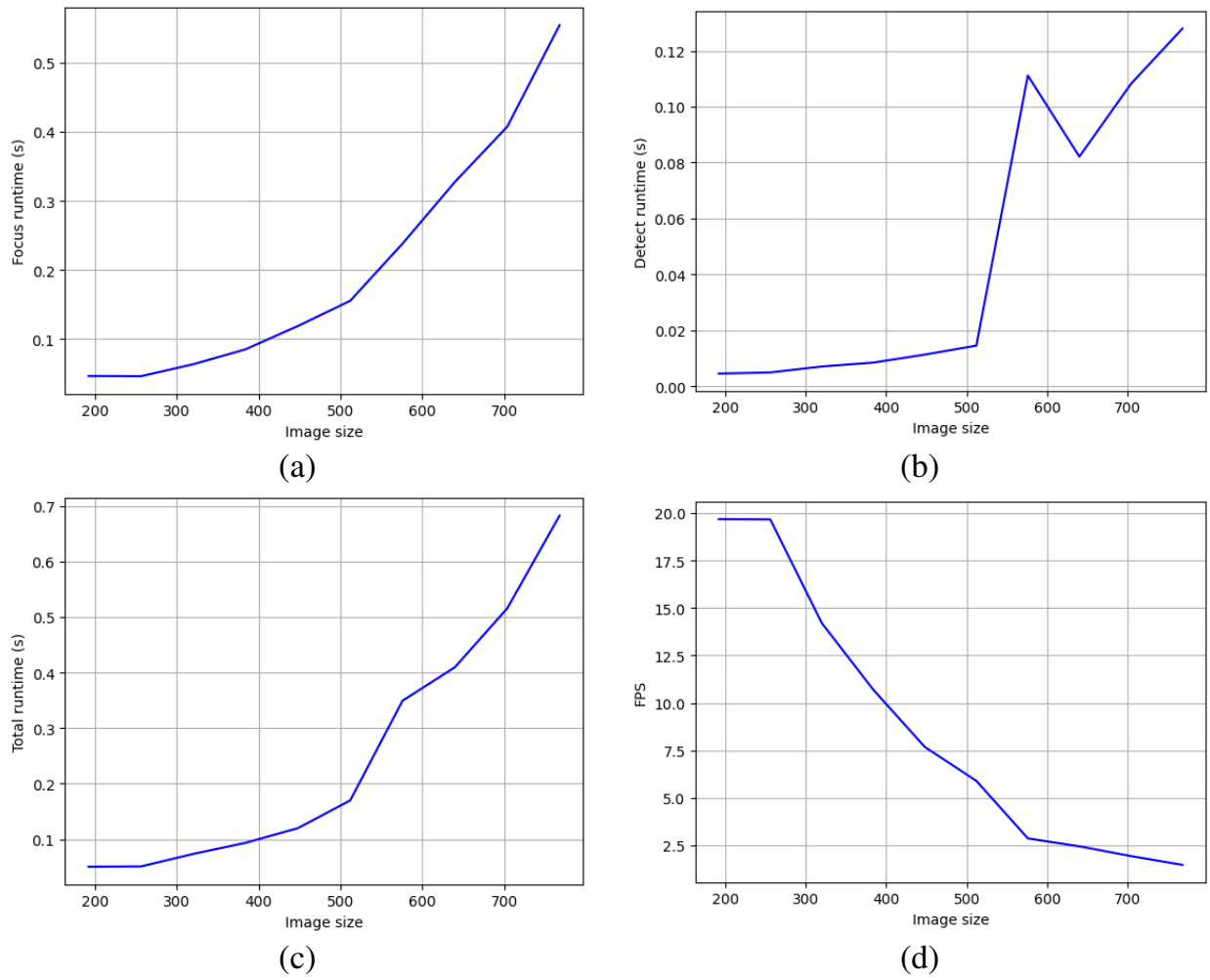


Figure 4.2: Runtime measurement and FPS of TextFocus on SCUT-CTW1500 [1]. The runtime is calculated by getting the mean of each measure for all samples in the dataset. (a) Focus runtime consists of a backbone and a focus branch. (b) Detect runtime consists of detecting branches and post-processing. (c) Total runtime consists of backbone, focus branch, detect branch, and post-processing. (d) FPS is calculated from the total runtime for each image size input

4.4.3 Comparison results

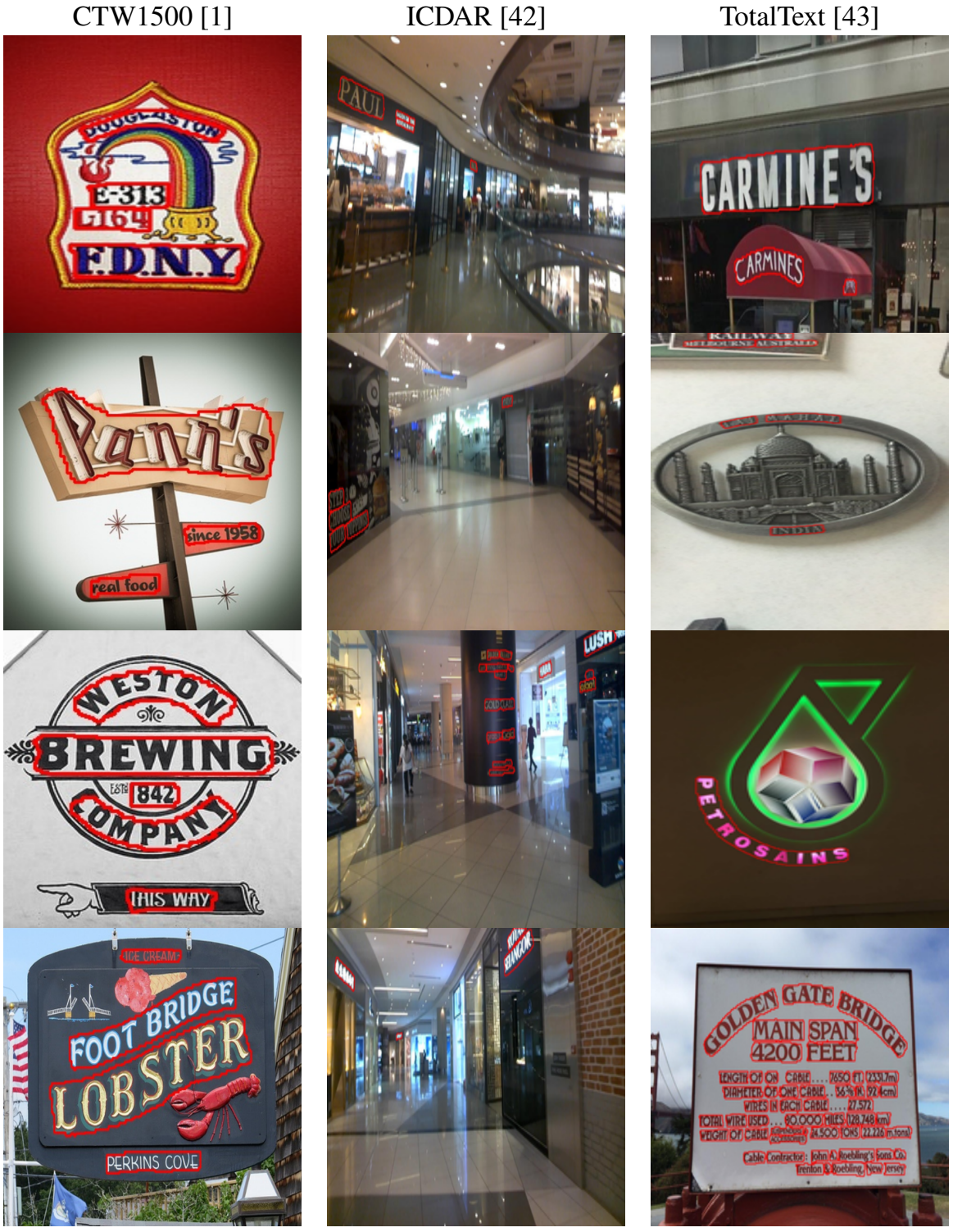


Figure 4.3: Visualize results on three standard benchmarks. (a) Results on SCUT-CTW1500. (b) Results on TotalText. (c) Results on ICDAR2015.

To evaluate the performance of our method for detecting text instance and speed, we compare the TextFocus with other state-of-the-art methods on SCUT-CTW1500, Total Text, and ICDAR2015. In the testing phase, we set the short side of images input to the model to different scales (320, 640). We report the single-scale performance of TextFocus

on these datasets in Table 4.2.

On SCUT-CTW1500, TextFocus-320 with short side 320 achieves the F-measure 82.8 with astonishing speed (14.21 FPS), and TextFocus-640 outperforms all other methods in F-measure by 1.2% while real-time runtime is acceptable (2.45).

The same conclusion can be obtained from Total-Text. TextFocus-320 achieves a competitive F-measure of 78.1 with real-time speed (11.13 FPS), and the best F-measure conducted by TextFocus-640 is 82.05, while the rate can still be acceptable (2.12 FPS).

On ICDAR2015, TextFocus-640 achieves a better F-measure than the quickest method, PAN and has a faster runtime than the best F-measure score one is PSENet (85.7), but TextFocus-640 still achieves a slightly lower result (84.7).

Performance on SCUT-CTW1500, TotalText, and ICDAR2015 shows that the TextFocus is superior in detecting text instances and maintaining real-time speed. We also illustrate several visual results in Figure B, which demonstrate the performance of TextFocus on this dataset.

CHAPTER 5. CONCLUSION AND FUTURE WORK

In this thesis, we have a novel method for arbitrary shape text detection using multiple resolutions. The method is designed to overcome previous methods' limitations and improve text detection accuracy in real-world images. The TextFocus model has been developed with a multiresolution strategy, which can observe the input image at different resolutions, providing more detailed information for recognizing text instances.

The method has been thoroughly researched and analyzed, and extensive experiments have been conducted to validate its performance. The experiments have shown that the method significantly outperforms the baseline model on FPS and TIoU-metric improvement. This demonstrates the validity and advantages of the method in improving the accuracy of arbitrary shape text detection .

However, there are still some limitations in the method that need to be addressed in future works. Firstly, the model has not achieved state-of-the-art performance on some datasets with low-resolution images. This could be improved by developing a more robust model that can handle low-resolution images more effectively. Secondly, the focus branch head in the model has not performed as well as expected, and a brighter, lighter attention head needs to be designed to improve its performance. Thirdly, the computational complexity of the method needs to be reduced to make it more suitable for real-world computer vision applications and pattern recognition tasks.

To address these limitations, future works can focus on several aspects to improve the method. Firstly, incorporating more advanced deep learning techniques, such as transfer learning and attention mechanisms, could further enhance the performance of the method. Secondly, developing a more robust model to handle more multiresolution images more effectively would improve the method's overall performance. Thirdly, creating a more efficient attention head and reducing the computational complexity of the technique would make it more suitable for real-world applications.

Furthermore, the method of creating synthetic data needs to be reviewed. A better way is required to place the text region in challenging positions resembling the real-world scenario. Moreover, an algorithm must be developed to place the text in the work that best suits the background, improving the segmentation's accuracy.

In addition, the method can be extended to other image detection applications, such as mini object detection and OCR. The technique can also be applied to real-time scenarios and integrated with edge devices such as traffic cameras to solve intelligent traffic systems.

In conclusion, this thesis has a new method for arbitrary shape text detection using

multiple resolutions, demonstrating significant accuracy improvements compared to state-of-the-art techniques. The future works outlined in this thesis will further improve the method and contribute to the advancement of text recognition applications. The method can potentially improve the accuracy of text detection and reduce the false positive rate, ultimately improving the understanding of text scenarios on images. The results of this thesis will contribute to the object detection field and pattern recognition.

REFERENCES

[1] 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.

[2] T.-L. Yuan, Z. Zhu, K. Xu, C.-J. Li, T.-J. Mu, and S.-M. Hu, "A large chinese text dataset in the wild," *Journal of Computer Science and Technology*, vol. 34, pp. 509–521, 2019.

[3] H. Cho, M.-C. Sung, and B. Jun, "Canny text detector: Fast and robust scene text localization algorithm," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3566–3573, 2016. [Online]. Available: `https://api.semanticscholar.org/CorpusID:15330792`.

[4] A. Veit, T. Matera, L. Neumann, J. Matas, and S. Belongie, "Coco-text: Dataset and benchmark for text detection and recognition in natural images," *arXiv preprint arXiv:1601.07140*, 2016.

[5] M. Mukhiddinov, "Scene text detection and localization using fully convolutional network," in *2019 International Conference on Information Science and Communications Technologies (ICISCT)*, IEEE, 2019, pp. 1–5.

[6] Y. Cao, S. Ma, and H. Pan, "Fdta: Fully convolutional scene text detection with text attention," *IEEE Access*, vol. 8, pp. 155 441–155 449, 2020.

[7] V. Naosekpam, S. Aggarwal, and N. Sahu, "Utextnet: A unet based arbitrary shaped scene text detector," in *International Conference on Intelligent Systems Design and Applications*, Springer, 2021, pp. 368–378.

[8] K. Fu, L. Sun, X. Kang, and F. Ren, "Text detection for natural scene based on mobilenet v2 and u-net," in *2019 IEEE international conference on mechatronics and automation (ICMA)*, IEEE, 2019, pp. 1560–1564.

[9] P. Shivakumara, A. Banerjee, U. Pal, L. Nandanwar, T. Lu, and C.-L. Liu, "A new language-independent deep cnn for scene text detection and style transfer in social media images," *IEEE Transactions on Image Processing*, 2023.

[10] S.-X. Zhang, C. Yang, X. Zhu, and X.-C. Yin, "Arbitrary shape text detection via boundary transformer," *IEEE Transactions on Multimedia*, 2023.

[11] M. Ye, J. Zhang, S. Zhao, *et al.*, "Deepsolo: Let transformer decoder with explicit points solo for text spotting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 19 348–19 357.

[12] Y. Liu, L. Jin, Z. Xie, C. Luo, S. Zhang, and L. Xie, "Tightness-aware evaluation protocol for scene text detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 9612–9620.

[13] S. Tian, Y. Pan, C. Huang, S. Lu, K. Yu, and C. L. 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.

[14] Z. Zhang, W. Shen, C. Yao, and X. Bai, "Symmetry-based text line detection in natural scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 2558–2567.

[15] L. Sun, Q. Huo, W. Jia, and K. Chen, "A robust approach for text detection from natural scene images," *Pattern Recognition*, vol. 48, no. 9, pp. 2906–2920, 2015.

[16] 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.

[17] J. Ma, W. Shao, H. Ye, *et al.*, "Arbitrary-oriented scene text detection via rotation proposals," *IEEE transactions on multimedia*, vol. 20, no. 11, pp. 3111–3122, 2018.

[18] 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.

[19] 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.

[20] X. Zhou, C. Yao, H. Wen, *et al.*, "East: An efficient and accurate scene text detector," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 5551–5560.

[21] J.-B. Hou, X. Zhu, C. Liu, *et al.*, "Ham: Hidden anchor mechanism for scene text detection," *IEEE Transactions on Image Processing*, vol. 29, pp. 7904–7916, 2020.

[22] 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," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 67–83.

[23] S.-X. Zhang, X. Zhu, J.-B. Hou, C. Yang, and X.-C. Yin, "Kernel proposal network for arbitrary shape text detection," *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[24] J.-B. Hou, X. Zhu, C. Liu, *et al.*, "Detecting text in scene and traffic guide panels with attention anchor mechanism," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 11, pp. 6890–6899, 2020.

[25] X. Zhu, Z. Li, X.-Y. Zhang, C. Li, Y. Liu, and Z. Xue, "Residual invertible spatio-temporal network for video super-resolution," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 33, 2019, pp. 5981–5988.

[26] W. Wang, E. Xie, X. Li, *et al.*, "Shape robust text detection with progressive scale expansion network," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 9336–9345.

[27] W. Wang, E. Xie, X. Song, *et al.*, "Efficient and accurate arbitrary-shaped text detection with pixel aggregation network," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 8440–8449.

[28] Z. Tian, M. Shu, P. Lyu, *et al.*, "Learning shape-aware embedding for scene text detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4234–4243.

[29] Y. Xu, Y. Wang, W. Zhou, Y. Wang, Z. Yang, and X. Bai, "Textfield: Learning a deep direction field for irregular scene text detection," *IEEE Transactions on Image Processing*, vol. 28, no. 11, pp. 5566–5579, 2019.

[30] M. Liao, Z. Wan, C. Yao, K. Chen, and X. Bai, "Real-time scene text detection with differentiable binarization," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, 2020, pp. 11 474–11 481.

[31] X. Wang, Y. Jiang, Z. Luo, C.-L. Liu, H. Choi, and S. Kim, "Arbitrary shape scene text detection with adaptive text region representation," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 6449–6458.

[32] Y. Wang, H. Xie, Z.-J. Zha, M. Xing, Z. Fu, and Y. Zhang, "Contournet: Taking a further step toward accurate arbitrary-shaped scene text detection," in *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 753–11 762.

[33] Y. Liu, H. Chen, C. Shen, T. He, L. Jin, and L. Wang, "Abcnet: Real-time scene text spotting with adaptive bezier-curve network," in *proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 9809–9818.

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

[35] F. Wang, Y. Chen, F. Wu, and X. Li, "Textray: Contour-based geometric modeling for arbitrary-shaped scene text detection," in *Proceedings of the 28th ACM international conference on multimedia*, 2020, pp. 111–119.

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

[37] H. Wang, P. Lu, H. Zhang, *et al.*, "All you need is boundary: Toward arbitrary-shaped text spotting," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 34, 2020, pp. 12 160–12 167.

[38] W. Feng, W. He, F. Yin, X.-Y. Zhang, and C.-L. Liu, "Textdragon: An end-to-end framework for arbitrary shaped text spotting," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 9076–9085.

[39] 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.

[40] W. Liu, D. Anguelov, D. Erhan, *et al.*, "Ssd: Single shot multibox detector," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*, Springer, 2016, pp. 21–37.

[41] 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.

[42] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, *et al.*, "Icdar 2015 competition on robust reading," in *2015 13th international conference on document analysis and recognition (ICDAR)*, IEEE, 2015, pp. 1156–1160.

[43] C. K. Ch'ng and C. S. Chan, "Total-text: A comprehensive dataset for scene text detection and recognition," in *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, IEEE, vol. 1, 2017, pp. 935–942.

[44] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[45] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, Springer, 2015, pp. 234–241.

[46] M. Najibi, B. Singh, and L. S. Davis, "Autofocus: Efficient multi-scale inference," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 9745–9755.

[47] H. Edelsbrunner, D. G. Kirkpatrick, and R. Seidel, "On the shape of a set of points in the plane," in *IEEE Transactions on Information Theory*, 1983. [Online]. Available: `https://api.semanticscholar.org/CorpusID:6983029`.

[48] 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.

[49] F. Zhan, S. Lu, and C. Xue, "Verisimilar image synthesis for accurate detection and recognition of texts in scenes," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 249–266.

[50] Z. Tian, W. Huang, T. He, P. He, and Y. Qiao, "Detecting text in natural image with connectionist text proposal network," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14*, Springer, 2016, pp. 56–72.

[51] 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.

[52] R. Nabati and H. Qi, "Rrpn: Radar region proposal network for object detection in autonomous vehicles," in *2019 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2019, pp. 3093–3097.

[53] 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.

APPENDIX

A. Data

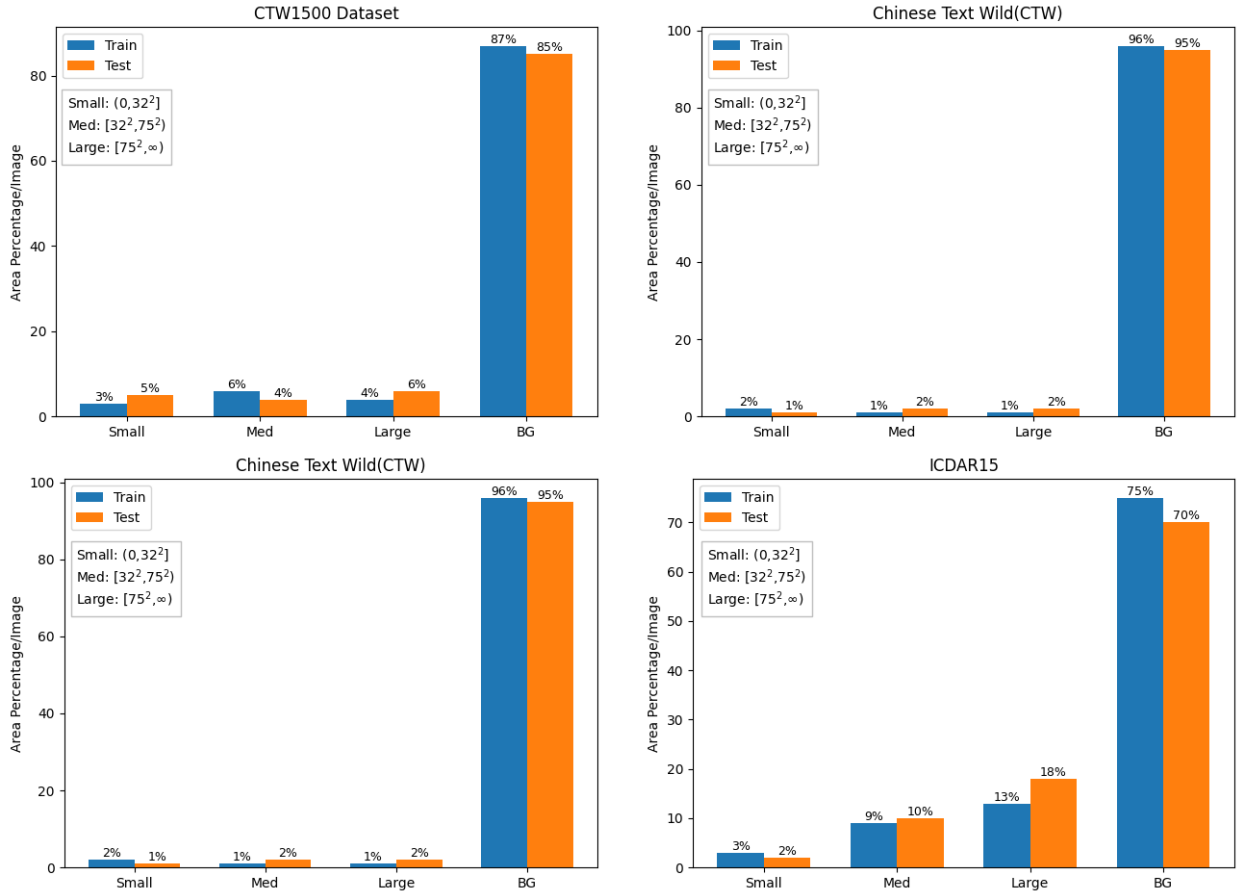


Figure A.1: Area of objects of different sizes and backgrounds in the SCUT-CTW1500[1] , Total Text[43], ICDAR2015[42], and Large CTW[2]. Things are divided based on their site (in pixels) into small, medium, and large.

B. MORE VISUALIZATIONS

Figure B.1: Inference pipeline in TextFocus. All the results are processed on the Large CTW dataset [2]. Focus Pixels are masked as pink, interesting region chips are shown in yellow in the second and fourth rows. Text instances detected in each chip are shown in purple in the first and third rows, final detection results are shown in red in the last row. As can be seen, high resolution image samples contain small text instances and some of them can be detected in the higher scale.