Text Detection and Recognition in the Wild

by

Zobeir Raisi

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Doctor of Philosophy
in
Systems Design Engineering

Waterloo, Ontario, Canada, 2022

© Zobeir Raisi 2022

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

External Examiner: Daniel P. Lopresti
Professor, Dept. of Computer Science & Engineering
Lehigh University, Bethlehem, PA, USA

Supervisor: John Zelek
Associate Professor, Dept. of Systems Design Engineering
University of Waterloo

Internal-External Member: Fakhri Karray
Professor, Dept. Electrical and Computer Engineering
University of Waterloo

Internal Member: Paul Fieguth
Professor, Dept. of Systems Design Engineering
University of Waterloo

Internal Member: Nasser Lashgarian Azad
Associate Professor, Dept. of Systems Design Engineering
University of Waterloo

Internal Member: Bryan Tripp
Associate Professor, Dept. of Systems Design Engineering
University of Waterloo

Author’s Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contributions

I hereby declare that I am the sole author of this thesis. This thesis consists in part of the following contributions written for publication as follows:

Zobeir Raisi, Mohamed A. Naiel, Paul Fieguth, Steven Wardell and John Zelek, “Smart Text Reader System for Blind Person using Machine and Deep Learning”, Signal and Image Processing with Machine Learning and Deep Learning Techniques, In Tripathi, S. L., Ghai, D., Chanda, M. (Editors), In Applications based Understanding of Machine and Deep Learning Algorithms for Signal and Image Processing, Wiley-IEEE Press, 2022. [in Press]

This paper is incorporated in Chapter 2 of this thesis.

Zobeir Raisi, Mohamed A. Naiel, Paul Fieguth, Steven Wardell and John Zelek, “2D Positional Embedding-based Transformer for Scene Text Recognition”, CVIS, 2020. [Best Vision Paper Award]

Some parts of this paper are incorporated in Chapter 3 of this thesis.

Zobeir Raisi, Mohamed A. Naiel, Georges Younes, Steven Wardell and John S. Zelek, “Transformer-based Text Detection in the Wild”, CVPR Workshop, 2021.

This paper is incorporated in Chapter 4 of this thesis.

Zobeir Raisi, Georges Younes, and John S. Zelek, “Arbitrary Shape Text Detection using Transformers”, International Conference on Pattern Recognition (ICPR), 2022.

This paper is incorporated in Chapter 5 of this thesis.

Zobeir Raisi, Mohamed A. Naiel, Georges Younes, Steven Wardell and John S. Zelek, “2LSPE: 2D Learned Sinusoidal Positional Encoding using Transformer for Scene Text Recognition”, Computer and Robot Vision (CRV), 2021.
This paper is incorporated in Chapter 6 of this thesis.

Zobeir Raisi and John S. Zelek, “End-to-End Scene Text Spotting at Character Level”, CVIS, 2021. [Best Vision Paper Award]
This paper is incorporated in Chapter 7 of this thesis.

Zobeir Raisi and John S. Zelek, “Occluded Text detection and Recognition in the Wild”, Computer and Robot Vision (CRV), 2022. [Best Computer Vision Paper Award]
This paper is incorporated in Chapter 8 of this thesis.

Abstract

Text detection and recognition (TDR) in highly structured environments with a clean background and consistent fonts (e.g., office documents, postal addresses and bank cheque) is a well understood problem (i.e., OCR), however this is not the case for unstructured environments. The main objective for scene text detection is to locate text within images captured in the wild. For scene text recognition, the techniques map each detected or cropped word image into string. Nowadays, convolutional neural networks (CNNs) and Recurrent Neural Networks (RNN) deep learning architectures dominate most of the recent state-of-the-art (SOTA) scene TDR methods. Most of the reported respective accuracies of current SOTA TDR methods are in the range of 80% to 90% on benchmark datasets with regular and clear text instances. However, those detecting and/or recognizing results drastically deteriorate $\sim 10\%$ and $\sim 30\%$ - in terms of F-measure detection and word recognition accuracy performances with *irregular* or *occluded* text images.

Transformers and their variations are new deep learning architectures that mitigate the above-mentioned issues for CNN and RNN-based pipelines. Unlike Recurrent Neural Networks (RNNs), transformers are models that learn how to encode and decode data by looking not only backward but also forward in order to extract relevant information from a whole sequence. This thesis utilizes the transformer architecture to address the irregular (multi-oriented and arbitrarily shaped) and occluded text challenges in the wild images. Our main contributions are as follows:

(1) We first targeted solving the irregular TDR in two separate architectures as follows:

- In Chapter 4, unlike the SOTA text detection frameworks that have complex pipelines and use many hand-designed components and post-processing stages, we design a conceptually more straightforward and trainable end-to-end architecture of transformer-based detector for *multi-oriented scene text detection*, which can directly predict the set of detections (i.e., text and box regions) of the input image. A central contribution to our work is introducing a loss function tailored to the rotated text detection problem that leverages a rotated version of a generalized intersection over union score to capture the rotated text instances adequately.
- In Chapter 5, we extend our previous architecture to *arbitrary shaped scene text detection*. We design a new text detection technique that aims to better infer n-vertices of a polygon

or the degree of a Bezier curve to represent irregular-text instances. We also propose a loss function that combines a generalized-split-intersection-over union loss defined over the piece-wise polygons.

- In Chapter 6, we show that our transformer-based architecture without rectifying the input curved text instances is more suitable than SOTA RNN-based frameworks equipped with rectification modules for *irregular text recognition* in the wild images. Our main contribution to this chapter is leveraging a 2D Learnable Sinusoidal frequencies Positional Encoding (2LSPE) with a modified feed-forward neural network to better encode the 2D spatial dependencies of characters in the irregular text instances.

(2) Since TDR tasks encounter the same challenging problems (e.g., irregular text, illumination variations, low-resolution text, etc.), we present a new transformer model that can detect and recognize individual characters of text instances in an end-to-end manner. Reading individual characters later makes a robust occlusion and arbitrarily shaped text spotting model without needing polygon annotation or multiple stages of detection and recognition modules used in SOTA text spotting architectures.

- In Chapter 7, unlike SOTA methods that combine two different pipelines of detection and recognition modules for a complete text reading, we utilize our text detection framework by leveraging a recent transformer-based technique, namely Deformable Patch-based Transformer (DPT), as a feature extracting backbone, to robustly read the class and box coordinates of *irregular* characters in the wild images.

(3) Finally, we address the *occlusion* problem by using a multi-task end-to-end scene text spotting framework.

- In Chapter 8, we leverage a recent transformer-based framework in deep learning, namely Masked Auto Encoder (MAE), as a backbone for scene text recognition and *end-to-end scene text spotting* pipelines to overcome the partial occlusion limitation. We design a new multitask End-to-End transformer network that directly outputs characters, word instances, and their bounding box representations, saving the computational overhead as it eliminates multiple processing steps. The unified proposed framework can also detect and recognize arbitrarily shaped text instances without using polygon annotations.

Acknowledgements

First, I would like to thank my supervisor, professor John Zelek, for his immense support without which this would not have been possible. I also would like to thank the Ontario Centres of Excellence (OCE), the Natural Sciences and Engineering Research Council of Canada (NSERC), and ATS Automation Tooling Systems Inc., Cambridge, ON, Canada for supporting this research work.

Dedication

This is dedicated to my wife and kids:
Anas, Ali & Alice.

Table of Contents

List of Tables

List of Figures

List of Symbols

$\mathcal{L}(.)$	Loss function
B	Bernstein Polynomials
S	Smooth Loss function
b	Bounding box
IoU	Intersection over Union
GIoU	Generalized Intersection over Union
PE	Positional Encoding
F	Feature Map
Q	Object queries
P	Control Points
p	Polygon Points
$\mathcal{P}$	2D Learnable Sinusoidal Positional Encoding
A	Attention Score Matrix
C	smallest convex shape

List of Abbreviations

BLSTM	Bidirectional LTSM
CIoU	Complete Intersection over Union
CL	Character-labeled
CNN	Convolutional Neural Network
CTC	Connectionist temporal classification
DPT	Deformable Patch-based Transformer
E2ESTS	End to End Scene text Spotting
FCN	Fully Convolutional Neural Network [3]
FFM	Feature Fusion Module [4]
FFN	Feed-Forward-Neural network
FPEM	Feature Pyramid Enhancement Module [4]
FPN	Feature pyramid networks [5]
IoU	Intersection over Union
GIoU	Generalized Intersection over Union
LSTM	Long Short Term Memory[6]
MAE	Masked Auto Encoders [7]
MJ	MJSynth
NLP	Natural Language Processing

PD	Private Data
PSPNet	Pyramid Scene Parsing Network [8]
RPN	Region Proposal Network [9]
RRN	Recurrent Neural Network
SSD	Single Shot Detector [10]
ST	SynthText
STN	Spatial Transformation Network [11]
STR	Scene Text Recognition
TPS	Thin-Plate Spline
TDR	Text Detection and Recognition
WRA	Word Recognition Accuracy

Chapter 1

Introduction

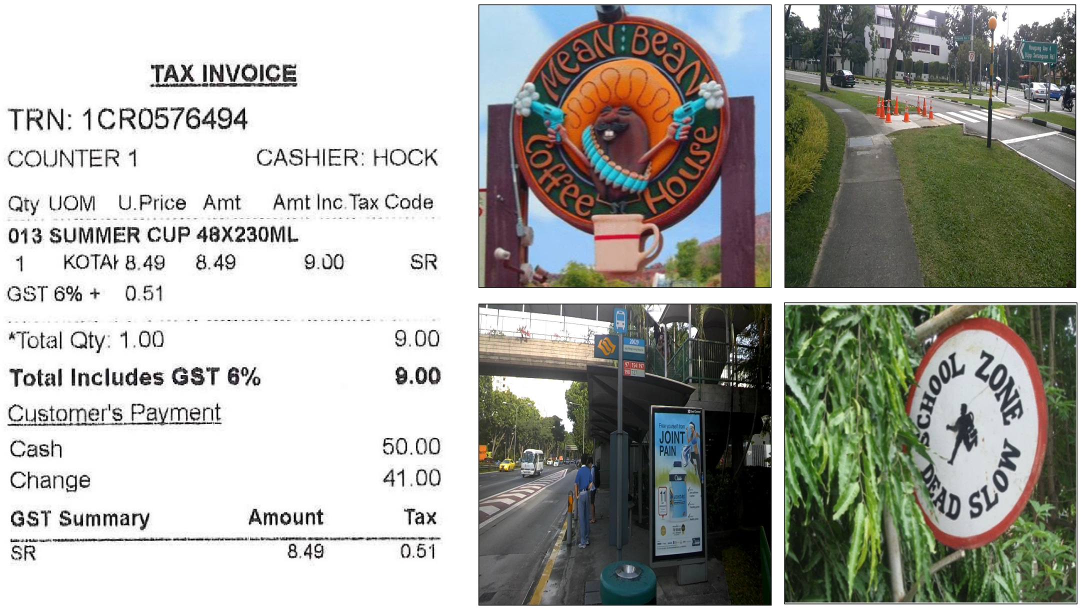
Text is a vital tool for communications and plays an important role in our lives. It can be embedded into documents or scenes as a mean of conveying information [12–14]. Identifying text can be considered as a main building block for a variety of computer vision-based applications, such as robotics [15, 16], industrial automation [17], image search [18, 19], instant translation [20, 21], automotive assistance [22] and analysis of sport videos [23]. Generally, the area of text identification can be categorized into two main categories: (1) identifying text of *scanned printed documents* and (2) text captured in daily scenes (e.g., images with arbitrarily rotated or distorted text captured on urban, rural, highway, indoor / outdoor of buildings, and subject to various geometric distortions, illumination and environmental conditions), where the latter is called *text in the wild* or *scene text*. Figure 1.1 illustrates examples of these two types of text-images. For identifying text of scanned printed documents, Optical Character Recognition (OCR) methods have been widely used [12, 24–26], and have achieved superior performances for reading printed documents with satisfactory resolution; However, these traditional OCR methods face many complex challenges when detecting and recognizing text in the wild [12, 13, 27]. The challenges of detecting and/or recognizing text in images captured in the wild can be categorized as follows:

- **Text diversity:** text can exist in a wide variety of colors, fonts, orientations and languages.
- **Scene complexity:** scene elements contain text on signs, bricks and symbols.
- **Distortion factors:** text is subjected to the effect of image distortion due to several contributing factors such as surface geometry, perspective view, motion blurriness, insufficient camera resolution, capturing angle and partial occlusion [12, 14].
- **Irregular text:** refers to the text with arbitrary shapes that usually have sever orientation and curvature.
- **Occlusion:** text instances are sometimes in situations where an external object/illumination blocks a portion of some characters or when a part of a character is missing.

In the literature, many techniques have been proposed to address the challenges of scene text detection and/or recognition. These schemes can be categorized into *classical machine learning-based*, as in [28–40], and *deep learning-based*, as in [41–67], approaches. A classical approach is often based on combining a feature extraction technique with a machine learning model to detect or recognize text in scene images [31, 68, 69]. Although some of these methods [68, 69] achieved good performance on detecting or recognizing horizontal text [12, 14], they rely on designing hand-crafted features, which limit their performances to handling arbitrarily shaped text instances. these methods typically fail to handle images that contains multi-oriented or curved text [13, 14]. On the other hand, deep-learning based methods have shown effectiveness in detecting and/or recognizing text in adverse situations [13, 46, 57, 67].

1.1 Problem Definition and Challenges

Recent scene text detection and recognition methods have utilized DCNN [9, 10, 73, 74] and RNN frameworks [6, 75], and have achieved promising performances on various challenging



TAX INVOICE

TRN: 1CR0576494

COUNTER 1 CASHIER: HOCK

Qty	UOM	U.Price	Amt	Amt Inc	Tax Code
013 SUMMER CUP 48X230ML					
1	KOTAł	8.49	8.49	9.00	SR
GST 6% +		0.51			

*Total Qty: 1.00	9.00
Total Includes GST 6%	**9.00**
Customer's Payment	
Cash	50.00
Change	41.00

GST Summary	Amount	Tax
SR	8.49	0.51

Figure 1.1: Examples for two main types of text in images: text in a printed document (left column) and text captured in the wild (right column), where sample images are from the public datasets in [70–72].

benchmark datasets [71, 72, 76–88]. However, there are two significant issues that still require more careful studies, which can be summarized as follows:

- **Irregular Text:** although recent methods [4, 46, 54, 57, 58, 89–92] have tried to detect irregular-text, there are still several drawbacks to these methods: (a) the resulted bounding boxes do not minimally encapsulate the text well, and (b) they require a complicated architecture with multiple stages of post-processing. The existing state-of-the-art scene text recognition methods [27, 63–67, 93–95] also perform well when the text in an image is horizontal or nearly horizontal but they fail to correctly recognize the text when text is in arbitrary shapes or geometrically distorted
- **Occluded Text:** Existing methods in scene text detection and recognition rely on the visibility of the target characters in images, however, text affected by heavy occlusion

may significantly undermine the performance of these methods [27, 57, 58, 67]. This failure is often due to the features generated by the current CNNs architectures that have limited robustness to occlusion. This opens the possibilities to either improve the feature extractors and/or the learning models to better handle these sever occlusions.

1.2 Objectives of Thesis

In order to address the problems mentioned above, the objectives of this thesis can be summarized as follows:

- The first goal of this thesis is to design a transformer-based architecture [96] with spatial transformation [97] in order to detect and recognize irregular text in the wild images.

  In this proposal, we introduce a new architecture that is able to detect multi-oriented or curved text, encapsulated by quadrilateral boxes or Bezier curve representation, which will overcome the drawbacks of directly deploying a general text detector as in [96] for the scene text detection task. In addition, we aim to simplify the current RNN based scene text recognition architectures [27, 64, 65, 67] by leveraging the transformer [98], and study the effect of the spatial rectification module on the overall recognition accuracy.
- The second goal of this thesis is to propose a direction to allow future techniques to overcome the occlusion limitation by unifying the masked autoencoders [7] with our end-to-end transformer-based text detection and recognition framework. The proposed pipeline can localize the occluders and subsequently focus on the non-occluded characters of the text to make a robust text detection/recognition.

1.3 Contributions

Our main goal in this research work is to design a transformer-based architecture [1, 96] in order to detect and recognize irregular and/or occluded text in the wild images. In order to address the problems mentioned above, the rationale and contributions of this research can be summarized as follows:

1.3.1 Transformer for Scene Text Detection

Rationale: Current scene text detection methods cast text detection as an object detection problem, and their framework is mostly inherited from object detection algorithms. In object detection problems, the goal is to classify all the boxes in the images. Object-detection is a challenging input-to-output mapping problem. This abstract mathematical problem is modeled as a Machine Learning (ML) problem. The ML problem by itself is a proxy, and the specification of it involves introducing new assumptions and approximations. The choices that are made introduce new sub-problems that require solving these new sub-problems, such as: (1) Too many boxes, (2) classification rule is undefined, (3) redundant outputs, and (4) foreground-background imbalance. However, more extensive changes may involve rethinking the machine-learning problem.

One of these types of researches is Detection using a transformer (DETR) [96], which has used different kinds of fundamental substrates. The essential advantage of using a transformer in detection is the using of an element relation modeling mechanism [99]. For this purpose, the transformer uses a set of object queries, which are learned vectors; they interact with each other and with the image features inside the transformer decoder. What differentiates these queries from the classical approach is that they do not have a prior geometric meaning. The category and box are predicted from each query taken from the model without applying a classical non-maximum suppression algorithm. Because the model predicts the output-set directly rather than defining a classification and regression problem on quantized boxes, it avoids label assignment heuristics issues.

By using a transformer for scene text detection: the encoder's multi-head self-attention during training learns how to separate individual words in the scene image by performing the global computations. Also, the decoder typically learns how to attend to different part of characters in the words by using different learnable vectors (so called object queries). After training, the last layer of the decoder is capable of directly predicting the set of detections with an absolute bounding box eliminating the use of any hand-designed components and post-processing like anchor design and non-max suppression [46, 48, 58, 90].

Contribution: Unlike the baseline transformer-based method in [96] that only generates rectangular bounding boxes for detected objects, and therefore, is not designed for handling arbitrary shape detection; we propose a new architecture that is able to detect multi-oriented (Chapter 4) and irregular text by leveraging a prediction head with a polygon or Bezier curve representation (Chapter 5). Thus, our method is more suited to the scene text detection task as it predicts for each text region 20 or 16 control points of a polygon box or a Bezier curve, respectively, which will overcome the drawbacks of directly deploying a general object detector as in [96] that predicts only 4 points of every rectangular box. We also leverage a loss function that better manages the changes in scales and aspect ratios of the detected text regions.

1.3.2 Transformer for Scene Text Recognition

Rationale: Recent recognition methods are mainly based on the combination of a convolutional neural network (CNN) as a feature extractor, with a Recurrent Neural Networks (RNNs) for capturing sequential dependency and producing sequence of characters. The existing RNN-based methods [27, 63–67, 93–95] perform well when the text in an image is horizontal or nearly horizontal but they fail to correctly recognize the text when text is in arbitrary shapes or distorted. The main reason for failures is that RNN-based methods require converted one-dimensional (1D) features and are not designed for recognizing irregular-text instances due to losing the spatial information within two-dimensional (2D) images. Some methods have tried to mitigate the high curvature recognition problem using a rectification module [11] by first rectifying the input image into a normalized image, and then treating recognition as a sequence prediction task. However, rectification causes errors in character recognition due to distortion perspective especially in severe curvature or vertical word images.

Different from RNN based sequence-to-sequence model, the transformer adopts global attention to encode and decode characters inside the text image using a look ahead strategy that does not consider the order of pixels. Transformers have been widely applied to problems with sequential data. Based on the idea that a scene image can be treated as a sequence of characters, we follow the original transformer [1] to design a scene text recognition model in order to recognize the sequence of characters in the image autoregressively.

Contribution: We also utilize a transformer for scene text recognition with some modification to its framework without using any rectification module. For this purpose, we leverage a 2D Learnable Sinusoidal Positional Encoding (2LSPE), in which the frequencies are learned, for scene text recognition. The proposed framework better captures the 2D spatial information of irregular-text characters via text-alignment in the image (Chapter 6). We also propose a new feed-forward-network layer in the encoder module to make it more robust to capturing the features generated by the encoder's self-attention mechanism. In addition, we aim to simplify the current RNN based scene text recognition architectures [27, 64, 65, 67] by leveraging a transformer [98], and study the effect of the spatial rectification module on the overall recognition accuracy.

1.3.3 End-to-End Scene Text Detection and Recognition

For a complete text reading, simultaneous text detection and recognition is required. Since text detection and recognition tasks encounter the same challenging problems (e.g., irregular text, illumination variations, low resolution text, etc.), we present a new end-to-end transformer that can detect and recognize text in the image at the same time. Unlike step-wise detection and recognition, the end-to-end framework will improve the overall speed by eliminating multiple processing steps. Further, the proposed end-to-end transformer offers higher accuracy than previous end-to-end CNN-based approaches [60, 100] (Chapter 7).

1.3.4 Masking Auto-encoders for Occlusion Handling

Since we can view the occluded text as a problem in where its elements are masked, one way to tackle the challenging occluded text problem is to use a masking approach by masking a large portion of the given input during training and reconstructing the missing pixels. Using this approach, we can increase the generalization capability of a given classifier by being able to tackle unseen scenarios. For this purpose, inspired by Masked Autoencoders (MAE) [7], we first leverage a fine-tuned MAE as a backbone to extract more semantic features in a new end-to-end scene text spotting framework. We propose a new multi-task prediction head and loss function that can directly output the class and bounding box coordinates of characters and the bounding box information of arbitrarily shaped word instances (Chapter 8).

Chapter 2

Literature Review

During the past decade, many techniques have been proposed for reading text in images captured in the wild [43, 47, 64, 69, 77]. The process of interpreting text from images can be divided into two serial tasks, namely, *text detection* and *text recognition* tasks. As shown in Fig. 2.1, text detection aims detecting or localizing text regions from images. On the other hand, text recognition task only focuses on the process of converting the detected text regions into computer-readable and editable characters, words, or text-line. In this chapter [101, 102], the conventional and recent algorithms for text detection and recognition will be discussed.

2.1 Text Detection

As illustrated in Figure 2.2, scene text detection methods can be categorized into *classical machine learning-based* [29, 31–33, 41, 69, 80, 103–117] and *deep learning-based* [44–51, 53, 54, 58, 61, 118] methods. In this section, we will review the methods related to each of these categories.

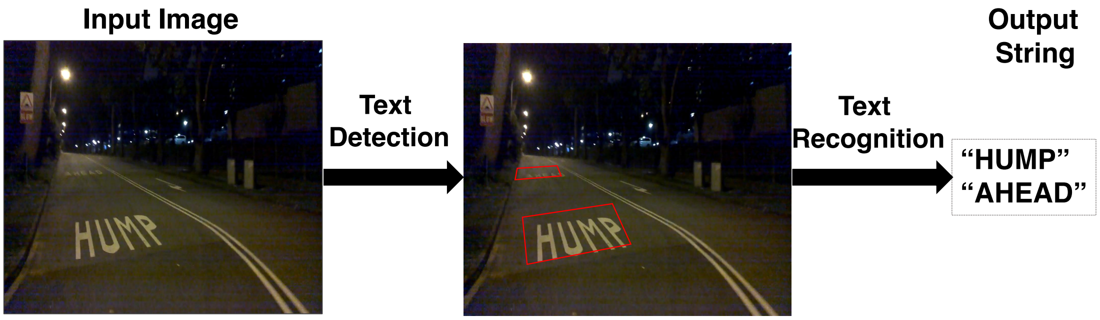


Figure 2.1: General schematic diagram of scene text detection and recognition, where sample image is from the public dataset in [119].

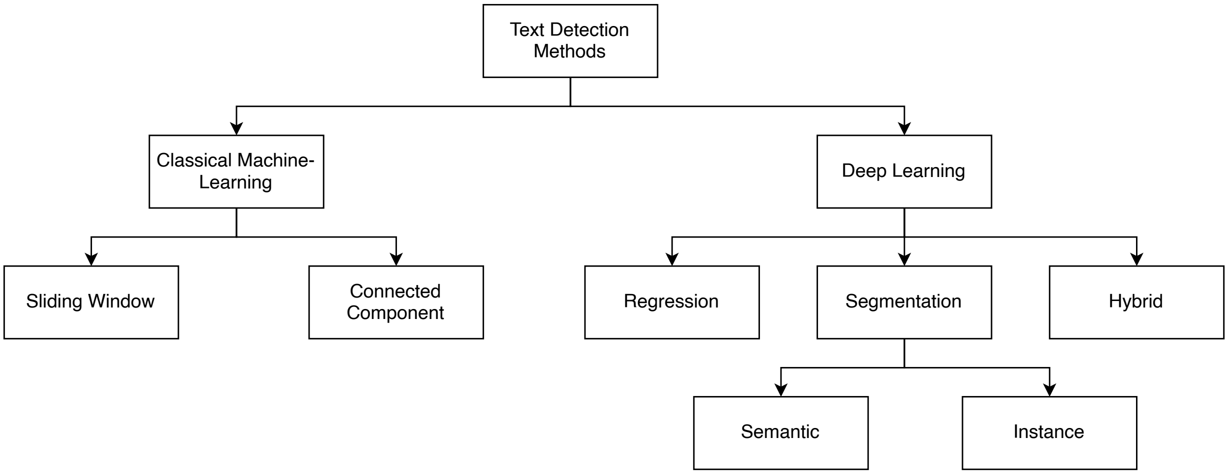


Figure 2.2: General taxonomy for the various text detection approaches.

2.1.1 Classical Machine Learning-based Methods

Traditional methods for scene text detection can be categorized into two main approaches, namely, *sliding-window* and *connected-component* based approaches. In *sliding window-based methods*, such as [28–33], a given test image is used to construct an image pyramid to be scanned over all the possible text locations and scales by using a sliding window of certain size. Then, a certain type of image features (such as histogram of oriented gradients (HOG) [120] as in [31, 121, 122]) are obtained from each window and classified by a classical classifier (such as random ferns [123] as in [31]) to detect text in each window.

Connected-component based methods aim to extract image regions of similar properties (such as color [34–38, 115], and corner points [124]) to create candidate components that can be categorized into text or non-text class by using a traditional classifier (such as SVM [107] and Random Forest [80]). These methods detect characters of a given image and then combine the extracted characters into a word [69, 107, 111, 116] or a text-line [125]. However, the classical-machine learning-based methods [33, 69, 107] perform poorly on some challenging cases like low-contrast images, compact characters and they require complicated rule-based techniques to generalize well on different arbitrarily shaped text instances [126].

2.1.2 Deep Learning-based Methods

The emergence of deep learning [127] has changed the way researchers approached the text detection task and has enlarged the scope of research in this field by far. Since deep learning-based techniques have many advantageous over the classical machine learning-based ones (such as faster and simpler pipeline [128], detecting text of various aspect ratios [118], and offering the ability to be trained better on synthetic data [43]) they have been widely used [49, 50, 129]. Table 2.1 summarizes a comparison among some of the current state-of-the-art techniques in this field.

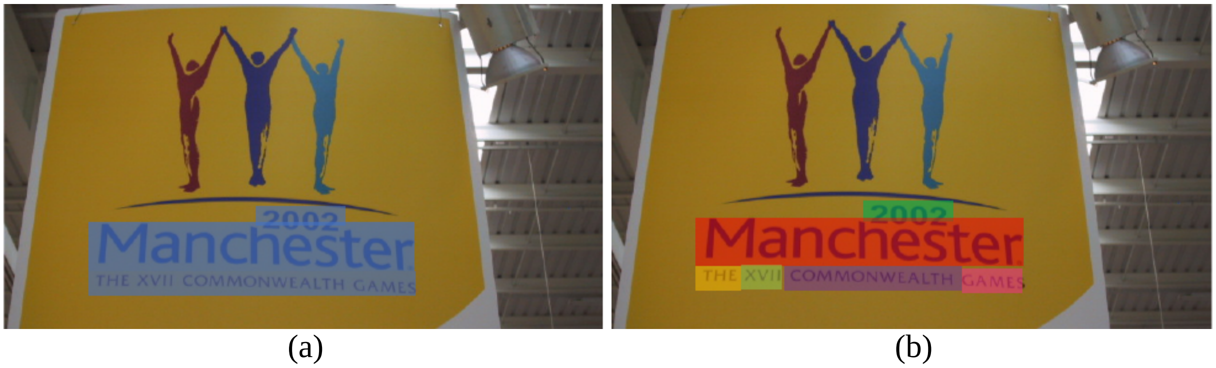
Recent deep learning-based text detection methods [45–49, 61, 118] inspired by object detection pipelines [3, 9, 10, 73, 74] can be categorized into *bounding-box based*, *segmentation-based* and *hybrid* approaches as illustrated in Figure 2.2. *Bounding-box based methods* for text-detection [44–49, 118] regard text as an object and aim to predict the candidate bounding boxes directly. For example, TextBoxes in [47] modified the single-shot descriptor (SSD) [10] kernels by applying long default anchors and filters to handle the significant variation of as-

Table 2.1: Deep learning text detection methods, where W: Word, T: Text-line, C: Character, D: Detection, R: Recognition, RB: Region-proposal-based, SB: Segmentation-based, ST: Synthetic Text, IC15: ICDAR15, IC13: ICDAR13, M500: MSRA-TD500, IC17: ICDAR17MLT, and the rest of the abbreviations used in this table are presented in the list of abbreviation.

Method	Year	IF			Neural Network		Detection	Challenges		Task	Code	Model	Training Datasets	
		BB	SB	Hy	Architecture	Backbone	Target	Quad	Curved			Name	First-Stage	Fine-Tune
Jaderberget *al.*[44]	2014	–	–		CNN	–	W	–	–	D,R	–	DSOL	MJSynth	–
Huang *et al.* [41]	2014	–	–		CNN	–	W	–	–	D	–	RSTD	–	IC11 or IC15
Tian *et al.* [45]	2016	✓	–		Faster R-CNN	VGG-16	T,W	–	–	D	✓	CTPN	PD	IC13
Zhang *et al.* [50]	2016	–	✓		FCN	VGG-16	W	✓	–	D	✓	MOTD	–	IC13, IC15 or M500
Yao *et al.* [51]	2016	–	✓		FCN	VGG-16	W	✓	–	D	✓	STDH	–	IC13, IC15 or M500
Shi *et al.* [118]	2017	✓	–		SSD	VGG-16	C,W	✓	–	D	✓	SegLink	ST	IC13, IC15 or M500
He *et al.* [130].	2017	–	✓		SSD	VGG-16	W	✓	–	D	✓	SSTD	–	IC13 or IC15
Hu *et al.* [131]	2017	–	✓		FCN	VGG-16	C	✓	–	D	–	Wordsup	ST	IC15 or COCO
Zhou *et al.* [46]	2017	✓	–		FCN	VGG-16	W,T	✓	–	D	✓	EAST	–	IC15*, COCO or M500
He *et al.* [129]	2017	✓	–		DenseBox	–	W,T	✓	–	D	–	DDR	–	IC13, IC15 & PD
Ma *et al.* [49]	2018	✓	–		Faster R-CNN	VGG-16	W	✓	–	D	✓	RRPN	M500	IC13 or IC15
Jiang *et al.* [132]	2018	✓	–		Faster R-CNN	VGG-16	W	✓	–	D	✓	R2CNN	IC15 & PD	–
Long *et al.* [53]	2018	–	✓		U-Net	VGG-16	W	✓	✓	D	✓	TextSnake	ST	IC15, M500, TOT or CTW
Liao *et al.* [48]	2018	✓	–		SSD	VGG-16	W	✓	–	D,R	✓	TextBoxes++	ST	IC15
He *et al.* [61]	2018	–	✓		FCN	PVA	C,W	✓	–	D,R	✓	E2ET	ST	IC13 or IC15
Lyu *et al.* [59]	2018	–	✓		Mask-RCNN	ResNet-50	W	✓	–	D,R	✓	MTSpotter	ST	IC13, IC15 or TOT
Liao *et al.* [133]	2018	✓	–		SSD	VGG-16	W	✓	–	D	✓	RRD	ST	IC13, IC15, COCO or M500
Lyu *et al.* [134]	2018	–	✓		FCN	VGG-16	W	✓	–	D	✓	MOSTD	ST	IC13 or IC15
Deng *et al.**[54]	2018	✓	–		FCN	VGG-16	W	✓	–	D	✓	Pixellink*	IC15	IC13, IC15* or M500
Liu *et al.*[60]	2018	✓	–		CNN	ResNet-50	W	✓	–	D,R	✓	FOTS	ST	IC13, IC15 or IC17
Baek *et al.**[57]	2019	–	✓		U-Net	VGG-16	C,W,T	✓	✓	D	✓	CRAFT*	ST	IC13, IC15* or IC17
Wang *et al.**[4]	2019	–	✓		FPEM+FFM	ResNet-18	W	✓	✓	D	✓	PAN*	ST	IC15*, M500, TOT or CTW
Liu *et al.**[58]	2019	–	–	✓	Mask-RCNN	ResNet-50	W	✓	✓	D	✓	PMTD*	IC17	IC13 or IC15*
Xu *et al.* [135]	2019	–	✓		FCN	VGG-16	W	✓	✓	D	✓	Textfield	ST	IC15, M500, TOT or CTW
Liu *et al.** [90]	2019	–	✓		Mask-RCNN	ResNet-101	W	✓	✓	D	✓	MB*	ST	IC15*, IC17 or M500
Wang *et al.** [89]	2019	–	✓		FPN	ResNet	W	✓	✓	D	✓	PSENet*	IC17	IC13 or IC15*

Note: * The method has been considered for evaluation.

pect ratios within text instances. With considering that scene text generally appears in arbitrary shapes, several works have tried to improve the performance of detecting multi-orientated text [46, 48, 49, 118, 129]. For instance, He *et al.* [129] proposed a multi-oriented text detection based on direct regression to generate arbitrary quadrilaterals text by calculating offsets between every point of text region and vertex coordinates. Later, Liao *et al.* [48] extended TextBoxes to TextBoxes++ by improving the network structure and the training process. Textboxes++ replaced the rectangle bounding boxes of text to quadrilateral to detect arbitrary-oriented text. Although bounding-box based methods [46–49, 118] have simple architecture, they require complex anchor design, hard to tune during training, and may fail to deal with detecting curved text.



(a) (b)

Figure 2.3: Semantic vs. instance segmentation. Ground-truth annotations for (a) semantic segmentation, where very close characters are linked together, and (b) instance segmentation. The image comes from the public dataset in [82]. Note, this figure is best viewed in color format.

Segmentation-based methods in [50–56, 58] cast text detection as a *semantic segmentation problem*, which aim to classify text regions in images at the pixel level as shown in Fig. 2.3(a). These methods, first extract text blocks from the segmentation map generated by a FCN [3] or mask regional-convolutional neural network (Mask R-CNN) [74], and then obtain bounding boxes of the text by post-processing. Although these segmentation-based methods [50, 51] perform well on rotated and irregular text, they might fail to accurately separate the adjacent-word instances that tend to connect. To address the problem of linked neighbour characters, Pixellinks [54] leveraged 8-directional information for each pixel to highlight the text margin, and Lyu [55] proposed corner detection method to produce position-sensitive score map. Recently, in [89] a progressive scale expansion network (PSENet) was introduced to find kernels with multiple scales and separate text instances close to each other accurately. However, the method in [89] requires a large number of images for training, which increases the run-time and can present difficulties on platforms with limited resources.

Recently, several works [58, 59, 136, 137] have treated scene text detection as an *instance segmentation problem*, as shown in Fig. 2.3(b), and many of them have applied Mask R-CNN

[74] framework to improve the performance of scene text detection. For example, SPCNET proposed in [137] uses a text context module and a re-score mechanism to suppress false positives. However, these methods [59, 136, 137] have the following drawbacks: Firstly, they suffer from the errors of bounding box handling in a complicated background, where the predicted bounding box fails to cover the whole text image. Secondly, these methods [59, 136, 137] aim at separating text pixels from the background ones, which can lead to many mislabeled pixels at the text borders [58].

Hybrid methods [130, 133, 134, 138] use segmentation-based approach to predict score maps of text and aim as bounding-box based approach to obtain text bounding-boxes through regression. For example, Liu *et al.* [58] proposed a new Mask R-CNN-based framework, namely, pyramid mask text detector (PMTD) for scene text detection, which assigns a soft pyramid label, $l \in [0, 1]$, for each pixel in text instance, and then reinterprets the obtained 2D soft mask into 3D space.

2.2 Text Recognition

Text recognition converts image regions into characters or words, where character classes in the English language often consist of: 10 digits, 26 lowercase letters, 26 uppercase letters, 32 ASCII punctuation marks, and 1 end of sentences (EOS) symbol. When the evaluation metric is case insensitive, only digits and letters are counted, and the rest are removed. However, text recognition models proposed in the literature have used different choices of character classes, which Table 2.2 provides their numbers.

Since the properties of text in the wild images are different from the text in scanned documents, it is challenging to develop an effective text recognition framework based on a traditional machine learning method, such as [76, 139–143], and applying it on these type of scene text images. This is because images captured in the wild tend to include text under various conditions such as images of low resolution [77, 79], lightning extreme [77, 79], environmental conditions [71, 82], and have different number of potential fonts [71, 82, 83], orientation angles [72, 83], languages [85] and lexicons [77, 79]. Researchers proposed different techniques to address these challenging issues, which can be categorized into the *classical machine learning-based* [31, 39, 40, 77, 141, 144] and *deep learning-based* [43, 63–66, 66, 88, 94, 145–154] methods, which in the rest of this section these two methods are discussed.

2.2.1 Classical Machine Learning-based Methods

In the past five decades, *classical machine learning-based* scene text recognition methods [39, 40, 106, 109, 141, 155, 156] have used standard image features, such as HOG [120] and SIFT [157], with SVM [158], k-nearest neighbours [159] classifier, then statistical language models or visual structure prediction applied to prune-out mis-classified characters [12, 160].

Most classical machine learning-based methods follow a bottom-up approach that classified *characters* are linked up into words. For example, in [31, 77], given a cropped word image, HOG features are first extracted, and then a pre-trained nearest neighbor or SVM classifier is applied on every feature of sliding window to classify the characters of the input word image. Other works adopted a top-down approach, where the *word* is directly recognized from the entire input images, rather than detecting and recognizing individual characters. For example, Almazan *et al.* [161] treated word recognition as a content-based image retrieval problem, where word image and word labels are embedded into an Euclidean space and the embedding vectors are used to match images and labels. However, these methods [31, 33, 77, 144] cannot achieve either an effective recognition accuracy, due to the low representation capability of handcrafted features, or building models that are able to handle text recognition in the wild. Buttom-up approach has proper interpretation since they can locate the position and label of each character.

Table 2.2: Comparison among some of the state-of-the-art of the deep learning-based text recognition methods, where TL: Text-line, C: Character, Seq: Sequence Recognition, PD: Private Dataset, HAM: Hierarchical Attention Mechanism, ACE: Aggregation Cross-Entropy, and the rest of the abbreviations are introduced in the list of abbreviation.

Method	Model	Year	Feature Extraction	Sequence modeling	Prediction	Training Dataset	Irregular recognition	Task	# classes	Code
Wang *et al.* [112]	E2ER	2012	CNN	–	SVM	PD	–	C	62	–
Bissacco *et al.* [33]	PhotoOCR	2013	HOG,CNN	–	–	PD	–	C	99	–
Jaderberg *et al.* [88]	SYNTR	2014	CNN	–	–	MJ	–	C	36	✓
Jaderberg *et al.* [88]	SYNTR	2014	CNN	–	–	MJ	–	W	90k	✓
He *et al.* [166]	DTRN	2015	DCNN	LSTM	CTC	MJ	–	Seq	37	–
Shi *et al.* [64]	RARE	2016	STN+VGG16	BLSTM	Attn	MJ	✓	Seq	37	✓
Lee *et al.* [145]	R2AM	2016	Recursive CNN	LTSM	Attn	MJ	–	C	37	–
Liu *et al.* [65]	STARNet	2016	STN+RSB	BLSTM	CTC	MJ+PD	✓	Seq	37	✓
Shi *et al.* [63]	CRNN	2017	VGG16	BLSTM	CTC	MJ	–	Seq	37	✓
Wang *et al.* [146]	GRCNN	2017	GRCNN	BLSTM	CTC	MJ	–	Seq	62	–
Yang *et al.* [147]	L2RI	2017	VGG16	RNN	Attn	PD+CL	✓	Seq	–	–
Cheng *et al.* [148]	FAN	2017	ResNet	BLSTM	Attn	MJ+ST+CL	–	Seq	37	–
Liu *et al.* [149]	Char-Net	2018	CNN	LTSM	Att	MJ	✓	C	37	–
Cheng *et al.* [150]	AON	2018	AON+VGG16	BLSTM	Attn	MJ+ST	✓	Seq	37	–
Bai *et al.* [151]	EP	2018	ResNet	–	Attn	MJ+ST	–	Seq	37	–
Liao *et al.* [167]	CAFCN	2018	VGG	–	–	ST	✓	C	37	–
Borisyuk *et al.* [66]	ROSETTA	2018	ResNet	–	CTC	PD	–	Seq	–	–
Shi *et al.* [27]	ASTER	2018	STN+ResNet	BLSTM	Attn	MJ+ST	✓	Seq	94	✓
Liu *et al.* [152]	SSEF	2018	VGG16	BLSTM	CTC	MJ	✓	Seq	37	–
Xie *et al.* [153]	ACE	2019	ResNet	–	ACE	ST+MJ	✓	Seq	37	✓
Zhan *et al.* [94]	ESIR	2019	IRN+ResNet,VGG	BLSTM	Attn	ST+MJ	✓	Seq	68	–
Wang *et al.* [154]	SSCAN	2019	ResNet,VGG	–	Attn	ST	✓	Seq	94	–
Wang *et al.* [168]	2D-CTC	2019	PSPNet	–	2D-CTC	ST+MJ	✓	Seq	36	–

Note: * This method has been considered for evaluation.

However, its performance is severely confined by the difficulty of character segmentation and the method usually requires, many labeled training samples for character classifier training (such as PhotoOCR [33]), which is both expensive and time-consuming [162]. Top-down approaches also fail in recognition of the input word image outside of the word-dictionary dataset.

2.2.2 Deep Learning-based Methods

With the recent advances in deep neural network architectures [3, 163–165], many researchers proposed *deep learning-based* text recognition methods [33, 88, 112] to tackle the challenges of recognizing text in the wild. Table 2.2 illustrates a comparison among some of the recent state-of-the-art deep learning-based text recognition methods [27, 63–66, 94, 145–154, 166–168].

The early deep CNN-based character recognition methods [33, 88, 112] require localizing each character, which may be challenging due to the complex background, irrelevant symbols, and the short distance between adjacent characters in scene text images. For word recognition, Jaderberg *et al.* [43] conducted a 90k English word classification task with a CNN architecture Although the [43] showed better word recognition performance in compare to just individual character recognition methods [33, 88, 112], they have two main drawbacks: (1) these methods can not recognize out-of-vocabulary words, (2) deformation of long word images may affect their recognition rate.

Considering that scene text generally appears in the form of a *sequence* of characters, many of recent works [63–65, 94, 148, 150–154, 166] map an input sequence to a variable length output sequence. Inspired by the speech recognition problem, several sequence-based text recognition methods [63, 65, 66, 145, 146, 152, 166] have used *connectionist temporal classification (CTC)* [169] for prediction of character sequences. Fig. 2.4 illustrates three main CTC-based text recognition frameworks that have been used in the literature. In many works [66, 170], CNN models (such as VGG [163], RCNN [165] and ResNet [164]) have been used with CTC as shown in Fig. 2.4(a). For instance, in [66] extracted features from convolutional neural network by are used to predict the feature sequences. Despite reducing the computational complexity, these methods [66, 170] suffered the lack of contextual information and showed a poor performance in terms of scene text recognition accuracy.

For better extracting contextual information, several works [63, 146, 166] used RNN [147] combined with CTC to identify the conditional probability between the predicted and the target sequences (Fig. 2.4(b)). For example, in [63] first a VGG model [171] is employed as a backbone to extract features of input image followed by a bidirectional long-short-term-memory (BLSTM) [6] for extraction of contextual information and then a CTC loss is applied to identify sequence of characters. However, these models [63, 146, 166] are insufficient to recognize irregular text, where characters are arranged on a 2-dimensional (2D) image plane because the CTC-based is

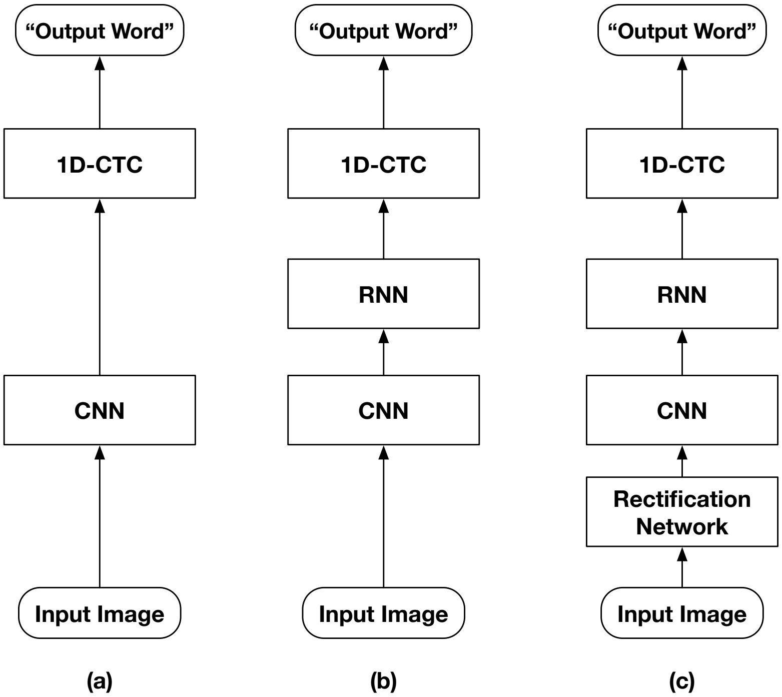


Figure 2.4: Comparison among some of the recent 1D CTC-based scene text recognition frameworks, where (a) baseline frame of CNN with 1D-CTC as in Rosetta [66], (b) adding RNN on the baseline frame as in [63], and (c) adding a Rectification Network on the framework of (b) as in STAR-Net [65].

only designed for 1-dimensional (1D) sequence to sequence alignment and it is hard to to apply it on 2D text recognition problem [153]. Furthermore, in these methods, 2D features of image are converted into 1D features, which may lead to loss of relevant information [168].

To handle irregular input text images, Liu *et al.* [65] proposed a spatial-attention residue Network (STAR-Net) that leveraged a spatial transform network (STN) [11] for tackling text distortions. It is shown in [65] that the usage of STN within the residue convolutional blocks, BLSTM and CTC framework, shown in Fig. 2.4(c), allowed performing scene text recognition under various distortions.

The *attention mechanism* that was first used for machine translation in [172] is also adopted for scene text recognition [27, 64, 65, 94, 145, 147, 149, 150]. This technique automatically learns implicit attention to enhance in-depth features in the decoding process. Fig. 2.5 illustrates five main attention-based text recognition frameworks that have been used in the literature. For regular text recognition, a basic 1D-attention-based encoder and decoder framework, as presented in Fig. 2.5(a) is used to recognize text images in [145, 173, 174]. For example, Lee and Osindero [145] proposed a recursive recurrent neural network with attention modeling (R2AM), where a recursive CNN is used for image encoding in order to learn broader contextual information, then an attention-based decoder is applied for sequence generation. However, directly training R2AM on irregular text is difficult due to the on-horizontal character placement [175].

Similar to CTC-based recognition methods, for handling irregular text many attention-based methods [27, 67, 93, 94, 149] have used image rectification modules to control distorted text images as shown in Fig. 2.5(b). For instance, Shi *et al.* [27] proposed a text recognition system that combined attention-based sequence and a STN module to rectify text. For this purpose, in [27, 64], a spatial transformer network (STN) is employed first to rectify the irregular text (*e.g.* curved or perceptively distorted), then the text within the rectified image is recognized by a RNN network. However, training a STN-based method without considering human-designed geometric ground truth is difficult, especially, in complicated arbitrary-oriented or strong-curved text images.

The performance of attention-based methods may decline in more challenging conditions, such as images of low-quality and sever distorted text, which may lead to misalignment and attention drift problems [168]. To reduce the severity of these problems, Cheng *et al.* [148] proposed a focusing attention network (FAN) that consists of an attention network (AN) for character recognition and a focusing network (FN) for adjusting the attention of AN. It is shown in [148] that FAN is able to correct the drifted attention automatically, and hence, improve the regular text recognition performance.

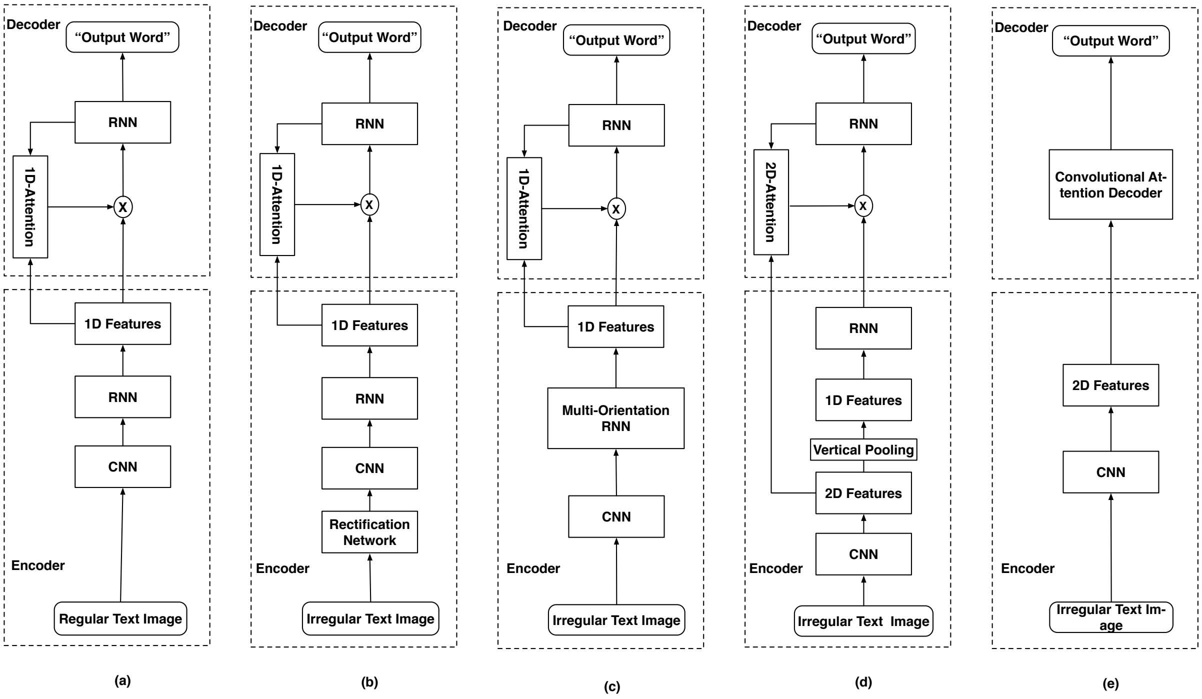


Figure 2.5: Comparison among some of the recent attention-based scene text recognition frameworks, where (a), (b) and (c) are 1D-attention-based frameworks used in a basic model [145], rectification network of ASTER [27], and multi-orientation encoding of AON [150], respectively, (d) 2D-attention-based decoding used in [176], (e) convolutional attention-based decoding used in SRCAN [154] and FACLSTM [177].

Some methods [147, 167, 176] used 2D attention [178], as presented in Fig. 2.5(d), to overcome the drawbacks of 1D attention. These methods can learn to focus on individual character features in the 2D space during decoding, which can be trained using either character-level [147] or word-level [176] annotations. For example, Yang *et al.* [147] introduced an auxiliary dense character detection task using a fully convolutional network (FCN) for encouraging the learning of visual representations to improve the recognition of irregular scene text. The overall pipeline of this method, which uses character level annotation, consist of the following components: a deep CNN for feature extraction, a FCN for dense character detection and a RNN in the final step for recognizing text, where an alignment loss was used to supervise the training of attention model during word decoding. Later, Liao *et al.* [167] proposed a framework called Character Attention FCN (CA-FCN), which models the irregular scene text recognition problem in a 2D space instead of the 1D space as well. In this network, a character attention module [179] is used to predict multi-orientation characters in an arbitrary shape of an image. Nevertheless, this framework requires character-level annotations and cannot be trained end-to-end [59]. In contrast, Li *et al.* [176] proposed a model that used word-level annotations, which enables this model to utilize both real and synthetic data for training without using character-level annotations. Nevertheless, 2-layer RNNs are adopted respectively in both encoder and decoder, which precludes computation parallelization and suffers from heavy computational burden.

To address these computational cost issue of 2D-attention-based techniques [147, 167, 176], in [177] and [154] the RNN stage of 2D-attentions techniques were eliminated, and a convolution-attention network [180] was used instead, enabling irregular text recognition, as well as fully parallel computation and accelerate the processing speed. Fig. 2.5(e) shows a general block diagram of this attention-based category. For example, Wang *et al.* [154] proposed a simple and robust convolutional-attention network (SRACN), where convolutional attention network decoder is directly applied into 2D CNN features. SRACN does not require to convert input images to sequence representations and directly can map text images into character sequences.

End-to-end methods [59–62] usually combine the detection and recognition modules and train them simultaneously. This approach aims to improve the detection performance by leveraging the recognition module. Unlike two-stage methods (step-wise) [43, 47, 48, 87], which detect and recognize text in two separate frameworks, the input of end-to-end methods is an image with ground-truth labels, and the output is a recognized text with its bounding box. For instance, Li *et al.* proposed an end-to-end trainable framework that used Faster-RCNN [9] for detection, and long short term memory (LSTM) attention mechanism for recognition. However, the main drawback of this model is that it is only applicable to horizontal text. FOTS [60] introduced RoIRotate

Table 2.3: Comparison among some of the recent text detection and recognition datasets.

Dataset	Year	# Detection Images			# Recognition words		Orientation			Properties		Task	
		Train	Test	Total	Train	Test	H	MO	Cu	Language	Annotation	D	R
IC03* [76]	2003	258	251	509	1156	1110	✓	–	–	EN	W,C	✓	✓
SVT* [77]	2010	100	250	350	–	647	✓	–	–	EN	W	✓	✓
IC11 [78]	2011	100	250	350	211	514	✓	–	–	EN	W,C	✓	✓
IIIT 5K-words* [79]	2012	–	–	–	2000	3000	✓	–	–	EN	W	–	✓
MSRA-TD500 [80]	2012	300	200	500	–	–	✓	✓	–	EN, CN	TL	✓	–
SVT-P* [81]	2013	–	238	238	–	639	✓	✓	–	EN	W	✓	✓
ICDAR13* [82]	2013	229	233	462	848	1095	✓	–	–	EN	W	✓	✓
CUT80* [83]	2014	–	80	80	–	280	✓	✓	✓	EN	W	✓	✓
COCO-Text* [84]	2014	43686	20000	63686	118309	27550	✓	✓	✓	EN	W	✓	✓
ICDAR15* [71]	2015	1000	500	1500	4468	2077	✓	✓	–	EN	W	✓	✓
ICDAR17 [85]	2017	7200	9000	18000	68613	–	✓	✓	✓	ML	W	✓	✓
TotalText [72]	2017	1255	300	1555	–	11459	✓	✓	✓	EN	W	✓	✓
CTW-1500 [86]	2017	1000	500	1500	–	–	✓	✓	✓	CN	W	✓	✓
SynthText [87]	2016	800k	–	800k	8M	–	✓	✓	✓	EN	W	✓	✓
MJSynth [88]	2014	–	–	–	8.9M	–	✓	✓	✓	EN	W	–	✓

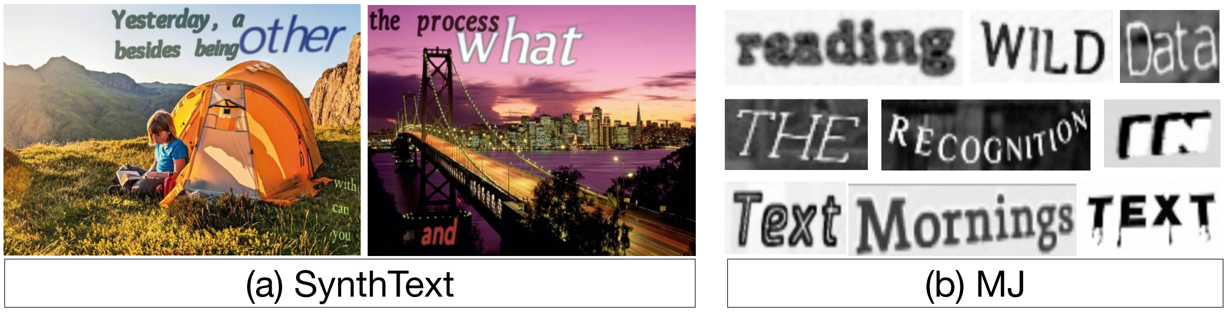
Note: * This dataset has been considered for evaluation. H: Horizontal, MO: Multi-Oriented, Cu: Curved, EN: English, CN: Chinese, ML:Multi-Language, W: Word, C: Character, TL: Textline D: Detection, R: Recognition.

to share convolutional features between detection and recognition module for better detection of both horizontal and multi-oriented text in the image. End-to-end text detection and recognition methods benefit the recognition results for improving the precision of detection, and some of these methods achieved superior performances in horizontal and multi-orientation text datasets. However, the high-performance detection of arbitrary-shape text such as curved or irregular text is still an open research problem.

2.3 Benchmark Datasets

In the field of text detection and recognition, several datasets have been introduced [71, 72, 76–88].

The two synthetic datasets, including Synth-Text [87] and MJ-Synth [88] datasets, are only used for pre-training of text detection and recognition models. There are also real-world datasets



(a) SynthText

(b) MJ

Figure 2.6: Sample images of synthetic datasets are mainly used for pre-training of (a) text detection and (b) recognition models in the wild. Images are taken from publicly available datasets in [87, 88]. All the above images are taken from publicly available benchmark datasets.

utilized extensively for evaluating the performance of detection schemes. We can categorize these datasets as follows:

1. **Horizontal-text datasets**: including ICDAR13 [82] and Coco-Text [119], which are annotated using rectangular bounding boxes. Some images of this datasets are shown in Figure 2.7.
2. **Multi-oriented text datasets:** including ICDAR15 [71], ICDAR17 [85], and MSRA-TD500 [181] that are annotated with quadrilateral and rotated-rectangle bounding boxes. Figure 2.8 illustrates some sample images of this datasets.
3. **Arbitrarily-shaped text datasets:** including Total-Text [72] and CTW-1500 [86] that mainly have curved and irregular text instances with multiple-vertices of polygon annotation. We have shown some images of these datasets in Figure 2.9.

There are also cropped-word datasets [71, 76, 77, 79, 81–84] designed to evaluate the scene text recognition algorithms. These datasets can be categorized into regular-text and irregular-text datasets, as shown in Figure 2.10. Table 2.3 also compares some of the recent text detection and recognition datasets, and the rest of this section presents a summary of each of these datasets.

2.3.1 MJSynth

The *MJSynth* [88] dataset is a synthetic dataset that specifically designed for scene text recognition. This dataset includes about 8.9 million word-box gray synthesized images, which have been generated from the Google fonts and the images of ICDAR03 [182] and SVT [77] datasets. All the images in this dataset have annotated in word-level ground-truth and 90k common English words have been used for generating of these text images.

2.3.2 SynthText

The *SynthText in the Wild* dataset [183] contains 858,750 synthetic scene images with 7,266,866 word-instances, and 28,971,487 characters. Most of the text instances in this dataset are multi-oriented and annotated with word and character-level rotated bounding boxes, as well as text sequences They are created by blending natural images with text rendered with different fonts, sizes, orientations and colors. This dataset has been originally designed for evaluating scene text detection [183], and leveraged in training several detection pipelines [57]. However, many recent text recognition methods [27, 94, 150, 153, 168] have also combined the cropped word images of the mentioned dataset with the MJSynth dataset [88] for improving their recognition performance.

2.3.3 ICDAR03

The *ICDAR03* dataset [182] contains horizontal camera-captured scene text images. This dataset has been mainly used by recent text recognition methods, which consists of 1,156 and 110 text instances for training and testing, respectively. In this work, we have used the same test images of [67] for evaluating the state-of-the-art text recognition methods.

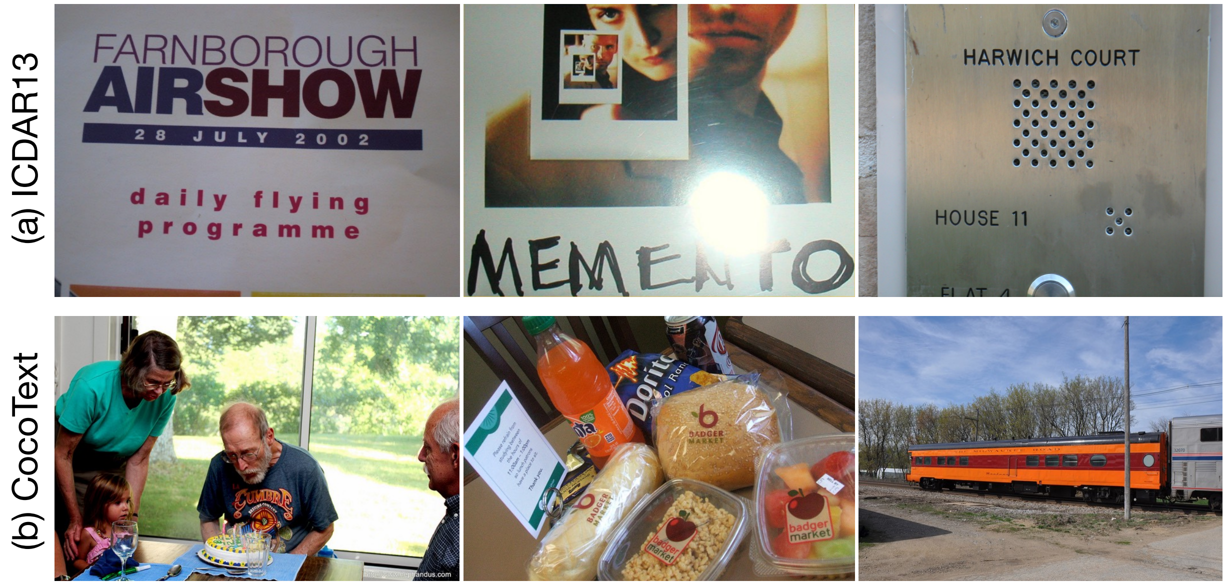


Figure 2.7: Example images of (a) ICDAR13 [82], and (b) Coco-Text [119] datasets that have rectangular bounding box annotations. All the above images are taken from publicly available benchmark datasets.

2.3.4 ICDAR13

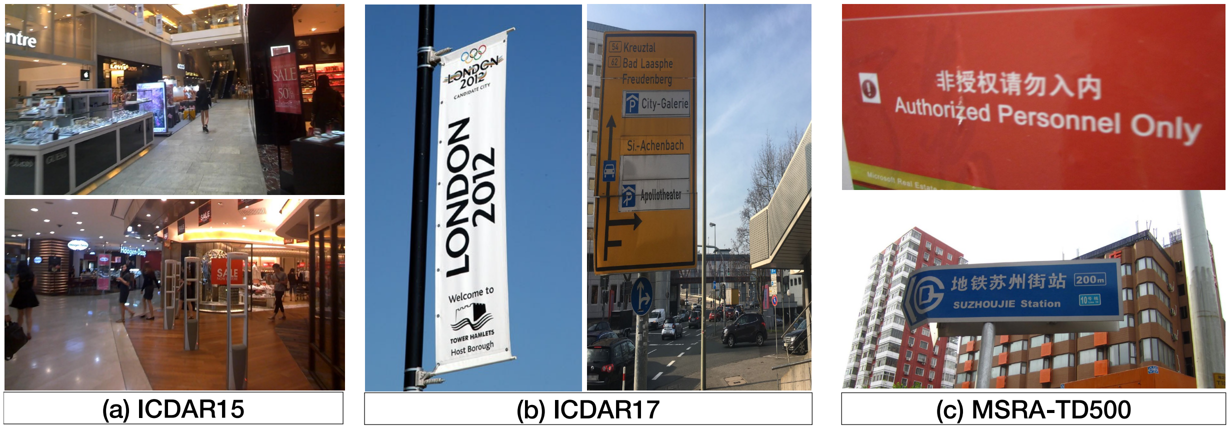
The *ICDAR13* dataset [82] includes images of horizontal text (the ith groundtruth annotation is represented by the indices of the top left corner associated with the width and height of a given bounding box as $G_i = [x_1^i, y_1^i, x_2^i, y_2^i]^\top$ that have been used in ICDAR 2013 competition and it is one of the benchmark datasets that used in many detection and recognition methods [46, 54, 57, 58, 60, 63–65, 67, 93]. The detection part of this dataset consists of 229 images for training and 233 images for testing, recognition part consists of 848 word-image for training and 1095 word-images for testing. All text images of this dataset have good quality and text regions are typically centered in the images.

2.3.5 COCO-Text

This dataset firstly was introduced in [84], and so far, it is the largest and the most challenging text detection and recognition dataset. As shown in Table 2.3, the dataset includes 63,686 annotated images, where the dataset is partitioned into 43,686 training images, and 20,000 images for validation and testing. In this work, we use the second version of this dataset, COCO-Text, as it contains 239,506 annotated text instances instead of 173,589 for the same set of images. As in ICDAR13, text regions in this dataset are annotated in a word-level using rectangle bounding boxes. The text instances of this dataset also are captured from different scenes, such as outdoor scenes, sports fields and grocery stores. Unlike other datasets, COCO-Text dataset also contains images with low resolution, special characters, and partial occlusion.

2.3.6 ICDAR15

The *ICDAR15* dataset [71] can be used for assessment of text detection or recognition schemes. The detection part has 1,500 images in total that consists of 1,000 training and 500 testing images for detection, and the recognition part consists of 4468 images for training and 2077 images for testing. This dataset includes text at the word-level of various orientation, and captured under different illumination and complex backgrounds conditions than that included in ICDAR13 dataset [82]. However, most of the images in this dataset are captured for indoors environment. In scene text detection, rectangular ground-truth used in the ICDAR13 [82] are not adequate for the representation of multi-oriented text because: (1), they cause unnecessary overlap. (2), they can not precisely localize marginal text, and (3) they provide unnecessary noise of background [184]. Therefore to tackle the mentioned issues, the annotations of this dataset are represented using quadrilateral boxes (the ith groundtruth annotation can be expressed as $G_i = [x_1^i, y_1^i, x_2^i, y_2^i, x_3^i, y_3^i, x_4^i, y_4^i]^\top$ for four corner vertices of the text).



(a) ICDAR15 (b) ICDAR17 (c) MSRA-TD500

Figure 2.8: Sample images of (a) ICDAR15 [71], (b) ICDAR17 [85] , and (c) MSRA-TD500 [181] datasets that are used for evaluation of multi-oriented text detection frameworks. All the above images are taken from publicly available benchmark datasets.

2.3.7 ICDAR17

The *ICDAR17* dataset is a large-scale word-level multi-lingual text dataset [85] comprised of 18000 natural scene images, sorted into 7200 for training, 1800 for validation and 9000 for testing. Similar to ICDAR15, This dataset also uses quadrilateral annotations [71], which we convert to our proposed rotated boxes format with the same procedure described in the preceding paragraph. It is noteworthy to mention that ICDAR17 is more challenging than ICDAR15 due to the varying text instances sizes, and the abundance of tiny text instances.

2.3.8 Total-Text

Total-Text [72] is a popular arbitrary-shaped text dataset that contains a large amount of curved text and straight text. Most of the images of the Total-text [72] dataset mainly contain irregular text while guarantee that each image has at least one curved text. The text instance is annotated with polygon based on word-level with only English words.

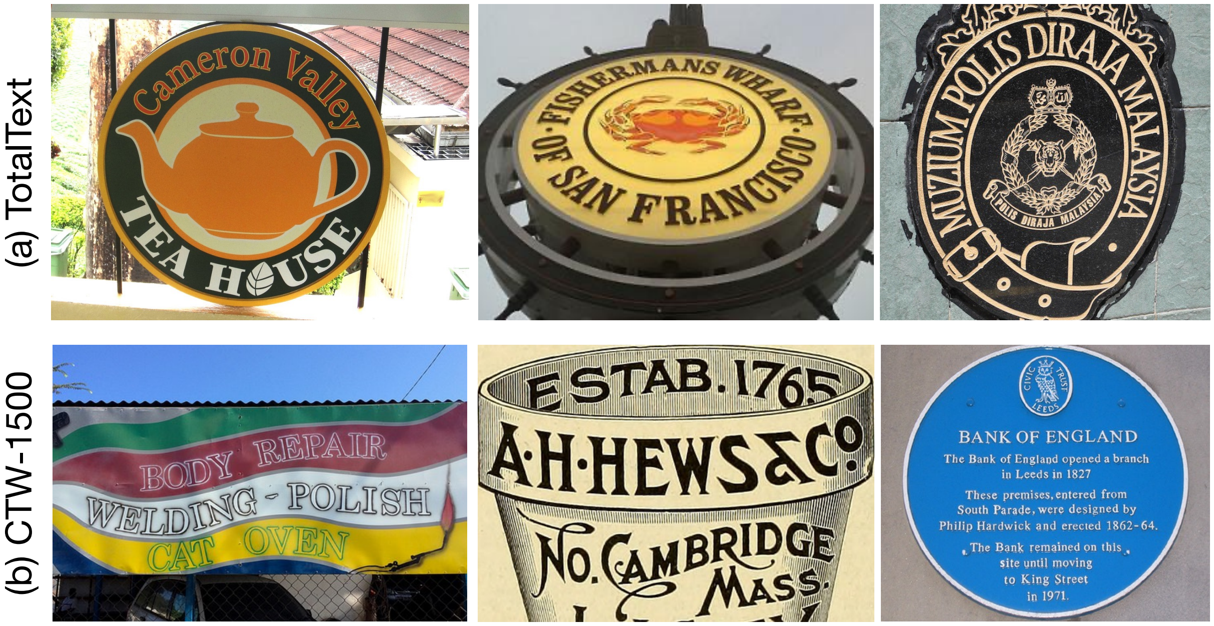


Figure 2.9: Arbitrarily-shaped text sample images of (a) Total-Text [72] and (b) CTW-1500 [86] benchmark datasets used for evaluating the peformance ”scent text detection” and ”end-to-end scene text detection and recognition” models. All the above images are taken from publicly available benchmark datasets.

2.3.9 CTW-1500

The *CTW1500* [86] dataset is also an arbitrary shape text detection dataset contains both English and Chinese text. Annotation in this dataset is based on the text-line level, in which polygons annotate every text instance with 14 vertices. Some text instances in this dataset include document-like text, in which much small text is close and stack together [100].

2.3.10 SVT

The *Street View Text (SVT)* dataset [77] consists of a collection of outdoor images with scene text of high variability of blurriness and/or resolutions, which were harvested using Google Street

View. As shown in Table 2.3, this dataset includes 250 and 647 testing images for evaluation of detection and recognition tasks, respectively. We utilize this dataset for assessing the state of the art recognition schemes.

2.3.11 SVT-P

The *SVT - Perspective (SVT-P)* dataset [81] is specifically designed to evaluate recognition of perspective distorted scene text. It consists of 238 images with 645 cropped text instances collected from non-frontal angle snapshot in Google Street View, which many of the images are perspective distorted.

2.3.12 IIIT 5K-words

The *IIIT 5K-words* dataset contains 5000 word-cropped scene images [79], that is used only for word-recognition tasks, and it is partitioned into 2000 and 3000 word images for training and testing tasks, respectively. In this work, we use only the testing set for assessment.

2.3.13 CUT80

The *Curved Text* (CUT80) dataset is the first dataset that focuses on curved text images [83]. This dataset contains 80 full and 280 cropped word images for evaluation of text detection and text recognition algorithms, respectively. Although CUT80 dataset was originally designed for curved text detection, it has been widely used for scene text recognition [83].

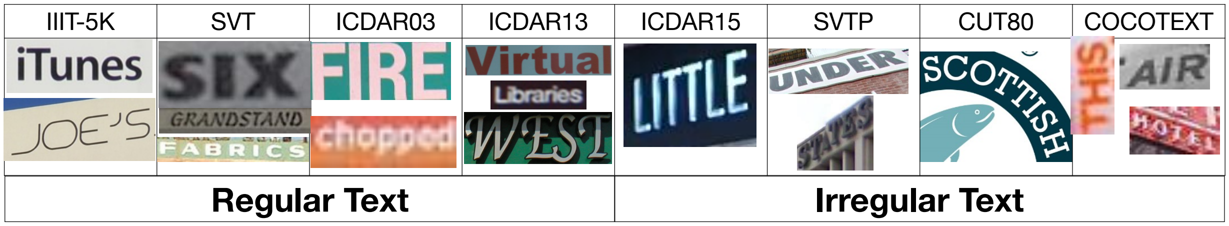


Figure 2.10: Sample cropped word images of benchmark datasets used for scent text recognition. The irregular-text datasets (left) [76, 77, 79, 82] mainly contain horizontal text, and irregular-text datasets (right) [71, 81, 83, 84] consist of primarily oriented, curved, and distorted text instances. All the above images are taken from publicly available benchmark datasets.

2.4 Summary

In the present Chapter, we have presented a detailed review on the recent advancement in scene text detection and recognition fields with focus on deep learning based techniques and architectures. We first have highlighted the related work done for scene text detection, which can be divided into three main categories: (1) bounding-box methods that mostly deployed object detection frameworks. while these methods provide good performance for regular text detection, their performance decline on irregular text. (2) segmentation-based methods, provide more robust in predicting the location of irregular text than the previous approaches. (3) Hybrid bounding box and segmentation based methods that are able to handle better multi-oriented text. However, scene text detection methods require fine-tuning on real-world datasets, and in images with text affected by more than one challenge, all these categories performed weakly.

We then have covered scene text recognition methods into two main categories: attention-based methods and CTC-based methods. In general, attention-based methods, that benefit from a deep backbone for feature extraction and transformation network for rectification have performed better than that of CTC-based methods. Scene text recognition methods often use synthetic scene images for training, and they can recognize text in real-world images without fine-tuning their models. However, there are several unsolved challenges for detecting or recognizing text in the wild images, such as in-plane-rotation, multi-oriented and multi-resolution text, perspective distortion, shadow and illumination reflection, image blurriness, partial occlusion, complex fonts, and special characters. The state-of-the-art detection and recognition methods fail or perform poorly with these challenges.

In the next chapter (Chapter 3), we provide some related background theory to the standard transformer's architecture introduced in [1]. We leverage this framework mainly for scene text recognition in Chapter 6. We also explore the recent Detection Using Transformers (DETR)'s pipeline [96], which is our baseline architecture for text detection in Chapter 4 and Chapter 5.

Chapter 3

Background Theory

This chapter covers the essential fundamental background related to the transformer pipeline. We later (Chapter 4), leverage this network as our main framework for text detection and recognition in wild images.

3.1 Attention

In this section, we define attention, which is the key defining part of a transformer models [1]. attention can be categorized into two types: *self attention* and *cross attention* (§3.2.1.2).

The *self-attention* layer is a normal attention block that allows the model to learn and access information of the past hidden layers. Let $x = [x_1, x_2, ..., x_t]^\top \in \mathbb{R}^{t \times d}$, within t and d denote the length and dimension of the input sequence. Each row of the self-attention function $A_1(x)$ can be demonstrated as a weighted sum of the value matrix $V \in R^{t \times d}$, with the weights determined by similarity scores between the key matrix $K \in \mathbb{R}^{t \times d}$ and query matrix $Q \in \mathbb{R}^{t \times d}$ as follows

[1, 185]:

$$
\begin{aligned}
&A_1(x) = \texttt{Softmax}\Big(\frac{QK^\top}{\sqrt{d}}\Big)V, \\
&Q = [q_1, q_2, ..., q_t]^\top, \qquad q_i = W_q x_i + b_q, \\
&K = [k_1, k_2, ..., k_t]^\top, \qquad k_i = W_k x_i + b_k, \\
&V = [v_1, v_2, ..., v_t]^\top, \qquad v_i = W_v x_i + b_v,
\end{aligned}
\tag{3.1}
$$

where $W_{(q/k/v)}$ and $b_{(q/k/v)}$ are the weight and bias parameters introduced in $A_1(\cdot)$.

As seen in Figure 3.1, rather than only computing the attention once, the *multi-head mechanism* runs through the scaled dot-product attention in equation (3.1) multiple times in parallel (more details in §A.3).

3.2 Transformer

Transformer's architecture has been initially introduced in [98] for machine translation by using a new attention-based mechanism. This architecture introduces self-attention layers, which scan through each element of a sequence and update it by measuring the relationship between this element and the whole sequence [98]. The main advantages of attention-based models in transformer are their parallel computations suitability at lower memory cost, which makes them more suitable than recurrent neural networks (RNNs) [6, 75] on learning from long sequences. This transformer architecture [98] has been later exploited in natural language processing (NLP) [186, 187] and it has been recently integrated in several successful applications in speech recognition [188] and computer vision [189–191].

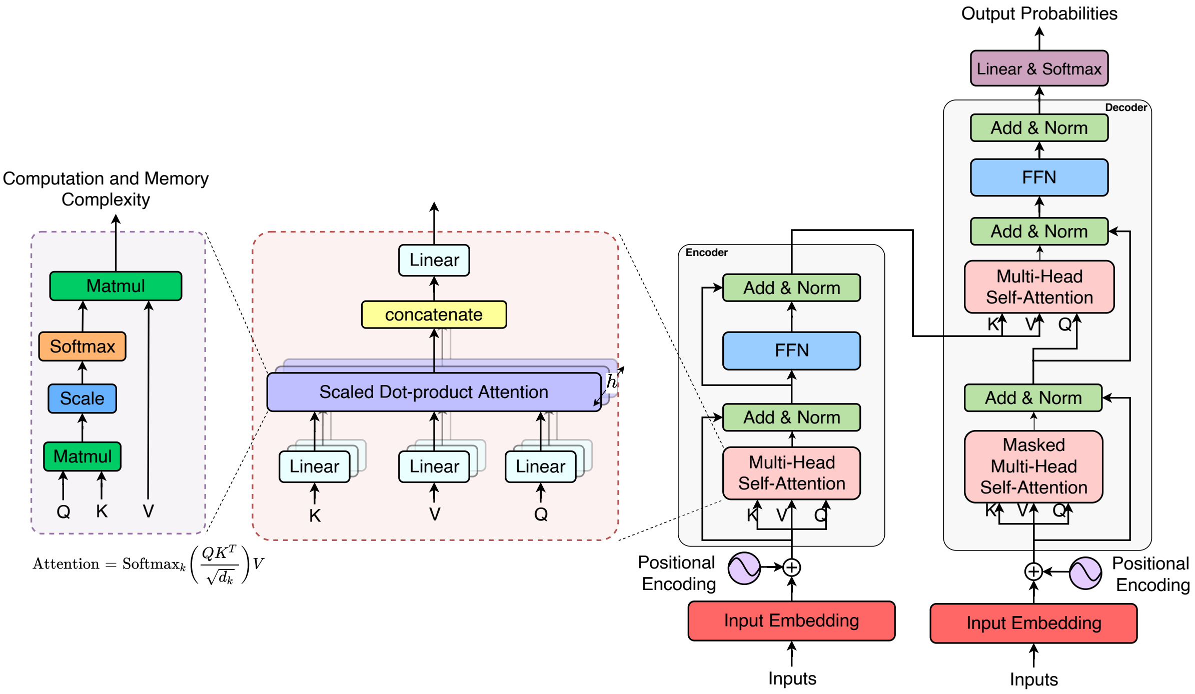


Figure 3.1: The standard Architecture of transformer. The diagram is reproduced from [1].

3.2.1 Transformer's Architecture

Transformer follows a similar architecture to autoencoder [192]; As shown in Figure 3.1, It consists of two major blocks: encoder and decoder, which without residual connections and layer normalization (Add-Norm), the architecture of a simplified transformer encoder/decoder can be seen as a stack of N sub-blocks $B_n : n = 1, \ldots, N$ containing a self-attention $A_n(\cdot)$, a FFN layer $F_n(\cdot)$, and a position encoding PE. Each sub-block B_n for a set of input $x = \{x_i\}_{i=1}^t$ can be expressed as follows [185]:

$$B_n(x) = F_n \circ A_n \circ PE\,(x) \tag{3.2}$$

3.2.1.1 Encoder

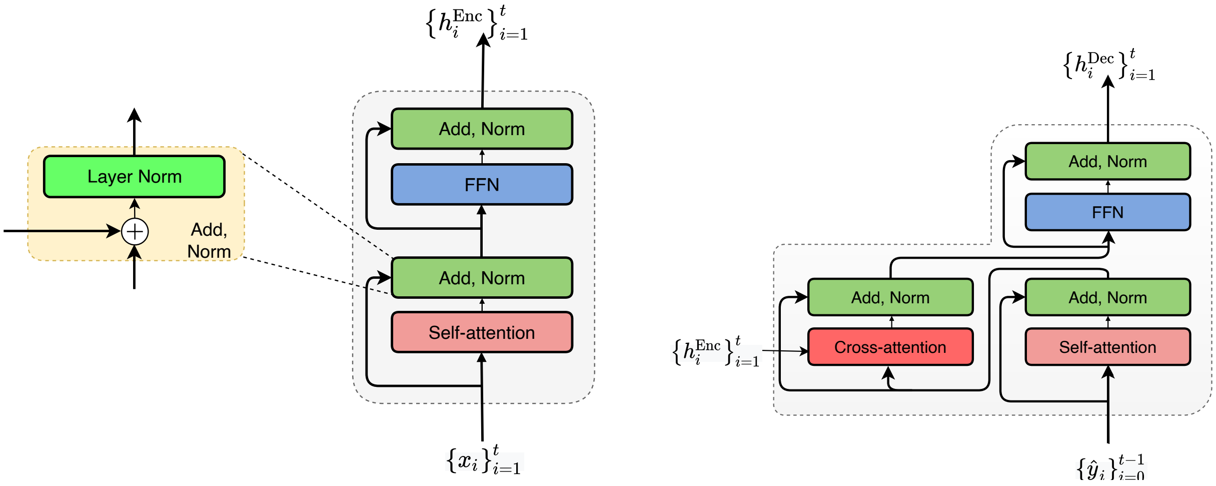
The Encoder module in the transformer, as shown in Figure 3.2a, accepts a set of inputs ($\{x_i\}_{i=1}^t$) at the bottom and then passes them through a Multi-Head Self-Attention (MHSA) followied by a Add & Norm and then a Feed-Forward-Neural network (FFN) layer. The output is then fed to another Add & Norm sub-block, which outputs a set of hidden representation as $\{h_i^{\text{Enc}}\}_{i=1}^t$. After adding 1D positional encodings to x (more details in §A.4), the self-attention sub-block takes the same item of the sequential inputs by generating the query, key, and values and weighs their relevance to each other to generate a set of output encodings, which are later fed through the rest of the encoder (more detail in §3.1). The Add, Norm sub-block in encoder module has two components: addition (residual connection) and layer normalization(LayerNorm), which can be written as follows $x = \text{LayerNorm}\big(x + F_s(x)\big)$, where F_S is the sub-layer module. It can be either the multi-headed self-attention or the feed-forward layer [193]. Following this step, The FFN sub-block is just applying a single multi-layer perceptron to every component in the set, which it later is used to adjust the dimensions of the output h_i^{Enc}. More specifically, the output of the feed-forward $F_1(\cdot)$ is a matrix with t rows, which the i-th row of it can be expressed as:

$$F_1(x) = W_2\sigma(W_1x_i + b_1) + b_2, \tag{3.3}$$

where $W_{1,2}$ and $b_{1,2}$ refer to the weights and biases of linear transforms, and $\sigma(\cdot)$ denotes the activation function.

3.2.1.2 Decoder

The decoder module in a transformer performs similarly to the encoder, which is querying of what is required through the set of representations from the encoder. However, as shown in Figure 3.2b, this module's inputs are different, and one extra sub-block exists in the middle,



(a) Encoder

(b) Decoder

Figure 3.2: Modules of the transformer. The diagram is reproduced from [1].

which is called cross-attention that connected after the self-attention following by an Add, Norm sub-block. The cross-attention sub-block follows the same query, key, and value setup used for the self-attention layer in encoder module, but the only difference is that it accepts the hidden representation (h_i^{Enc}) from the outputs of last layer of the encoder.

3.2.2 Transformer for Object Detection

Recently, in [96], a transformer's architecture [98] is utilized for object-detection and achieved superior results under challenging conditions. The novel design in the proposed architecture in [96] alleviates the need to use post-processing, reduces the need for anchor design to calculate the final object detection and improves the efficiency of target detection. Utilizing the transformer allowed this detector to work well on out-of distribution examples, and has offered a good performance on objects of various resolutions and affected by partial occlusion [96].

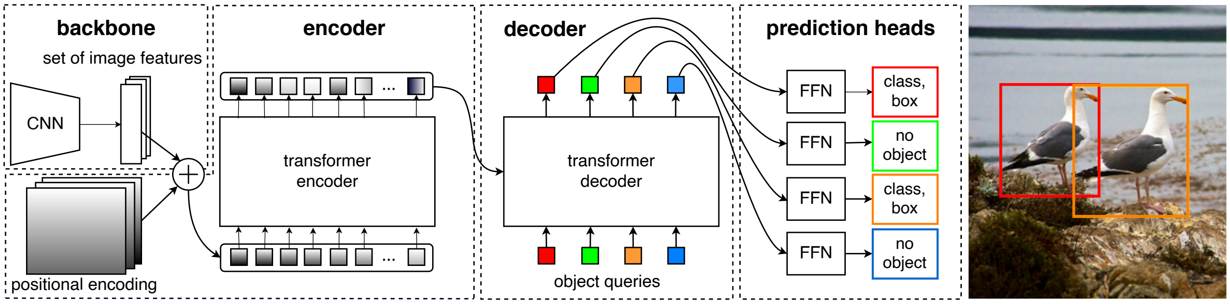


Figure 3.3: Overall Framework of object detection using transformer. See §A.2.1 for more detail. The above framework is reproduced from [96].

3.2.2.1 Architecture

The architecture for object detection using transformer [96] follows the similar standard pipeline of the transformer in [1] as shown in Figure 3.3. In this architecture, First, the input image is fitted to CNN to get image features. Then 2D positional embeddings made of sinusoids at different frequencies are added to the queries and keys of attention modules because the MSHA block in the transformer is permutation equivariant. For decoding, a fixed set of learned embeddings called object queries is passed through a transformer decoder. Finally, the obtained set of feature vectors are fed to shared FFN layers that predict the class and bounding box for each query.

Object Queries: Unlike other object-detection frameworks [9, 10, 73], the architecture object detection using transformer [96], does not manually incorporate any geometric prior for detection. Instead, the model learns it directly from data by using object queries. Object queries are N randomly initialized embedding vectors that they first are refined during training and then fixed for evaluation.

Set Prediction Loss: Set prediction loss introduced in [96] generates an optimal bipartite matching among the predicted and ground-truth objects. DETR [96] uses bipartite matching for one-vs-one matching between predictions and ground truth boxes rather than matching multiple bounding boxes to one ground truth box used in many object detection methods [9, 73]. The mentioned model always outputs N number of predictions, which the value of N is set larger than the number of objects in the image. The number of ground truth objects can be smaller than N, so dummy empty labels are added before performing the matching process.

Let y denotes the ground truth set of objects and $\hat{y}$ the set of N predictions, the loss function finds best possible one-to-one match between the ground truth and prediction pairs that minimize the loss , which it computes the optimal assignment between prediction and ground truth objects as [96]:

$$\hat{\sigma} = \arg\min_{\sigma \in \mathfrak{S}_N} \sum_{i}^{N} \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}), \tag{3.4}$$

where $\mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)})$ denotes pairwise *matching cost* between ground truth set y_i and prediction set $\hat{y}_\sigma(i)$. The model always outputs a set of prediction as c_i, b_i for each element i of N object queries, where c_i is the target class label, which can be $\varnothing$ for no-object class, and b_i is a rectangular bounding box in the format of four vector of $b_i = [x_i, y_i, w_i, h_i]$, where x, y denotes the up-left point and w, h are the width and height of the bounding box ,respectively.

The Bipartite matching loss between the ground truth and predicted is designed based on Hungarian algorithm [194], which determines the optimal assignment between ground truth and prediction. The Hungarian loss of equation (8.1) can be defined as follow:

$$\mathcal{L}_{\text{Hungarian}}(y, \hat{y}) = \sum_{i=1}^{N} \Big[-\log \hat{p}_{\sigma\hat{(i)}}(c_i) + \mathbb{1}_{\{c_i = \varnothing\}} \mathcal{L}_{\text{box}}(b_i, \hat{b}_{\hat{\sigma}}(i)) \Big] \tag{3.5}$$

where $\hat{\sigma}$ denotes the optimal assignment. The loss is defined similarly to the losses of common object detectors [96]. The model also includes bounding box loss, $\mathcal{L}_{\text{box}}$, which is defined as a linear combination of the ℓ_1 loss ($||b_i - \hat{b}_i||_1$) and generalized IOU loss ($\mathcal{L}_{\text{Giou}}$) [96]:

$$\mathcal{L}_{\text{box}}(b_i, \hat{b}_i) = \lambda_{\text{Giou}} \mathcal{L}_{\text{Giou}}(b_i, \hat{b}_i) + \lambda_{\text{L1}} ||b_i - \hat{b}_i||_1 \,, \tag{3.6}$$

and $\lambda_{\text{Giou}}, \lambda_{\text{L1}} \in \mathbb{R}$ are hyperparameters, and the generalized IOU loss [195] can be defined as:

$$\mathcal{L}_{\text{Giou}}(b_i, \hat{b}_i) = 1 - \left(\frac{|b_i \cap \hat{b}_i|}{|b_i \cup \hat{b}_i|} - \frac{|B(b_i, \hat{b}_i) \setminus b_i \cup \hat{b}_i|}{|B(b_i, \hat{b}_i)|} \right). \tag{3.7}$$

and $B(b_i, \hat{b}_i)$ means the largest box containing b_i and $\hat{b}_i$.

3.3 Summary

This chapter has presented the core information needed to understand the fundamentals behind transformer, which they are the main components that we utilize for our proposed scene text detection (Chapter 4 and Chapter 5), scene text recognition (Chapter 6), end-to-end scene text spotting (Chapter 7), and occluded text reading (Chapter 8) frameworks. We first have explored the fundamental theory behind the self-attention, the main block of transformer, and also summarized other sub-block of transformer's architecture. Then, we have explained the background information of a specific transformer architecture used for object detection.

In Chapter 4, we utilize the DETR's framework [96], discussed in this Chapter, for designing a conceptually more straightforward and trainable end-to-end text detection architecture. Since the rotated text is abundant in "in the wild" images, our main focus in the following Chapter is on detecting only multi-oriented text instances. We also discuss why we leverage the transformer's architecture for irregular text detection in wild images.

Chapter 4

Transformer-based Text Detection in the Wild

A major limitation of most state-of-the-art visual localization methods is their incapability to use ubiquitous signs and directions that are typically intuitive to humans. Localization methods can significantly benefit from a system capable of reasoning about various cues beyond low-level features, such as street signs, store names, building directories, room numbers, etc.

In this work, we tackle the problem of text detection in the wild, an essential step towards achieving text-based localization and mapping. While current state-of-the-art text detection methods employ ad-hoc solutions with complex multi-stage components to solve the problem, we propose a transformer-based architecture inherently capable of dealing with multi-oriented texts in images. A central contribution to our work is the introduction of a loss function tailored to the rotated text detection problem that leverages a rotated version of a generalized intersection over union score to properly capture the rotated text regions.

We evaluate our proposed model qualitatively and quantitatively on several challenging datasets namely, ICDAR15, ICDAR17, and MSRA-TD500, and show that it outperforms current state-of-the-art methods in text detection in the wild.

4.1 Introduction

Visual localization has played an essential role in recent advancements of several technologies such as augmented reality, self-driving cars and autonomous robotic navigation. However, most localization methods rely on low-level information (corners, edges, *etc.*) that does not necessarily correlate to topologically meaningful map representations [196]. Therefore, the next generation visual localization methods should be able to understand their surroundings and make use of the ubiquitous navigation labels surrounding them to navigate through previously unexplored environments. This is where text detection in the wild can play an important role as it enables image-based localization methods to reason about the ubiquitous navigation labels surrounding them to navigate through unexplored environments. However, texts can have several fonts, different colors, can appear on various surfaces, in different locations in the image, and with a wide range of orientations and scales [14, 101]. For example, they can appear anywhere from building names, store fronts, street signs, to shopping mall signs, *etc.* Therefore, reliable and consistent text detection is of utmost importance.

At its core, text detection is the process of localizing a word or a sentence in a given image. To that end, several recent scene text detection methods [46, 48, 49, 57, 58, 197, 198] have utilized deep convolutional neural networks (DCNNs) as feature extractors [9, 10, 73, 74], and solved for text detection by casting it as an object detection problem. Despite achieving promising results on various challenging datasets [71, 85, 181], their performance is still lacking in several key challenging scenarios, including and not limited to in-plane-rotations, multi-oriented and multi-resolution text, complex fonts, special characters, perspective distortion, occlusions, shadows, illumination artifacts, and image blurriness [101, 199]. We attribute these shortcomings to the ad-hoc multi-layered approaches most of these methods have deployed in an attempt to model the wide range of variation texts can have in the wild.

As such, we propose to account for these variations within our model, by leveraging the power of the transformers [1], which is a recent deep learning architecture that learns how to encode and decode data by looking not only backward but also forward to extract relevant information from a whole sequence. This new approach allows models to solve for complex tasks, such as machine translation [1], speech recognition [200], and recently, object detection [96, 201] and scene text recognition [202, 203].

In this work, we leverage a transformer framework to detect text instances in wild images. Unlike the baseline transformer-based method in [96, 201] that only generates rectangular bounding boxes for detected objects, and therefore, it is not designed for handling arbitrary shape detection; we propose a new architecture that is able to detect multi-oriented text. Our contributions are as follows:

1. We improve the detection performance by using the transformer [96] architecture, and by leveraging a differentiable loss function that accepts text instances' arbitrary shapes.
2. We propose using a rotated text representation that can better represent multi-oriented text regions.
3. We validate the performance of the proposed method by conducting several quantitative and qualitative experiments on challenging scenarios, and show that the proposed method outperforms the state-of-the-art on three public benchmark datasets, namely, ICDAR15 [71], ICDAR17 [85], and MSRA-TD500 [181].

4.2 Related Work

Text detection methods can be broadly categorized into two main groups:

1. First, *segmentation-based* methods [4, 54, 57, 58, 89] mainly use Mask-RCNN [74] as a backbone to produce a segmentation mask. They also consist of additional segmentation heads alongside the detection bounding box. Although these methods [4, 54, 57, 58, 89] offer high-precision detection when text is horizontal, they usually require multiple post-processing steps to infer the produced segmentation mask and predict precisely oriented bounding boxes [198]. Furthermore, their complicated architectures usually require high inference time due to the refinement of region proposal and label generation for arbitrary oriented text prediction.
2. On the other hand, *region-based* methods [46, 48, 129, 133, 134, 197–199] often predict candidate bounding box directly for the target region of interest. Unlike segmentation-based methods, region-based methods are more straight-forward and efficient for predicting the target region. However, applying the standard object detection frameworks directly for detecting arbitrarily-oriented text may cause redundant background noise, and unnecessary overlap [197]. Thus, for more accurate detection, many methods adopted rotated bounding boxes approach to better represent oriented text as in [46, 48, 49, 197, 198].

Particularly, EAST [46] presented a fast text detector that makes dense predictions which are then processed using locality-aware non-maximum suppression (NMS) to detect multi-oriented texts in an image. Later, TextBoxes++ [48] improved the rectangular detection architecture by using a long convolution kernel, increasing the number of region proposals, and replacing the rectangle bounding boxes of text with rotated boxes in order to detect arbitrarily-oriented text. In [197], Deng *et al.* introduced a mechanism called STELA for learning anchors and making the two-stage framework of Faster-RCNN into a one-stage detector to make the final oriented text detection more efficient. Recently, Wang *et al.* proposed RYOLO [198] that incorporated angle information of rotated boxes and feature maps of different scales to extend the standard YOLO framework for detecting rotated text. Although some of the mentioned methods

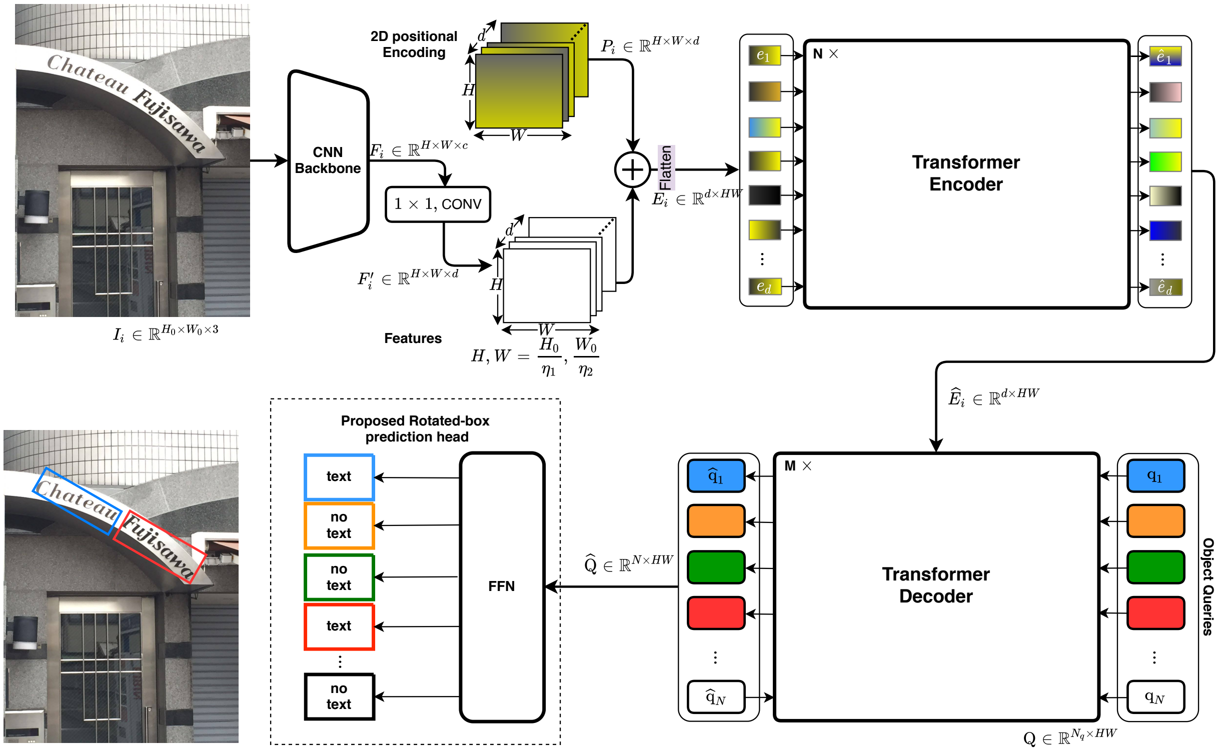

Chateau Fujisawa
$I_i \in \mathbb{R}^{H_0 \times W_0 \times 3}$
CNN Backbone
$F_i \in \mathbb{R}^{H \times W \times c}$
2D positional Encoding
d
H
W
$P_i \in \mathbb{R}^{H \times W \times d}$
1×1, CONV
$F_i' \in \mathbb{R}^{H \times W \times d}$
Features
$H, W = \frac{H_0}{\eta_1}, \frac{W_0}{\eta_2}$
Flatten
$E_i \in \mathbb{R}^{d \times HW}$
e_1
e_d
N ×
Transformer Encoder
$\hat{e}_1$
$\hat{e}_d$
$\widehat{E}_i \in \mathbb{R}^{d \times HW}$
Proposed Rotated-box prediction head
text
no text
no text
text
no text
FFN
$\widehat{\mathrm{Q}} \in \mathbb{R}^{N \times HW}$
$\hat{\mathrm{q}}_1$
$\hat{\mathrm{q}}_N$
M ×
Transformer Decoder
q_1
q_N
Object Queries
$\mathrm{Q} \in \mathbb{R}^{N_q \times HW}$

Figure 4.1: Block diagram of the proposed text-detection scheme using a transformer. Unlike the framework in [96], the proposed framework aims to represent text regions utilizing quadrilateral-based predictions instead of the classical rectangular-based predictions used in [96].

[197, 198] achieved state-of-the-art performance on several benchmark datasets, they require a complicated architecture with multiple stages of post-processing like NMS and rotating anchor design.

4.3 Methodology

Our main goal is to address the challenges of multi-oriented scene text detection by proposing a modified transformer-based architecture [96]. Transformers [1] are attention-based deep-learning architectures that can scan through each element of a sequence using a self-attention module, and provide an update by aggregating information from the whole sequence. When compared to previous deep-learning approaches, transformers can better capture the global dependencies among the input and output sequences with the help of an attention mechanism [204]. During training, the encoder's multi-head self-attention layer learns how to separate individual words in the scene image by performing the global computations, whereas the decoder learns how to attend to different characters in words by using different learnable vectors (also referred to as object queries). This is a very important feature since when properly trained, the last layer of the decoder is capable of directly predicting the targets' location without the need for multiple post-processing steps, as mentioned in Section 5.2, which are typically required by other architectures [46, 48, 57, 58, 90, 197, 198].

4.3.1 Architecture

The overall architecture of the proposed text detection scheme is shown in Figure 4.1. During the *encoding phase*, the i^{th} training color image $\mathcal{I}_i \in \mathbb{R}^{H_0 \times W_0 \times 3}$ is first processed to extract its features. While there are several ways of extracting features from an image such as RCNN [205], YOLO [73] *etc.*, we choose a ResNet [164] as CNN backbone because of its parameter efficiency and its ability in handling the vanishing gradient problem. The CNN produces a corresponding lower resolution feature map $F_i \in \mathbb{R}^{H \times W \times c}$, where c indicates the number of channels, $H = H_0/\eta$, and $W = W_0/\eta$ with η being the downsampling factor. In order to reduce the computational cost of the encoding stage, the number of channels within the feature map F_i, are reduced using a 1×1 convolutional layer, resulting in $F_i' \in \mathbb{R}^{H \times W \times d}$, where $d < c$.

As in [1], we also make use of 2D positional encoding maps $P \in \mathbb{R}^{H \times W \times d}$, which are added to F_i' such that $F_i'' = F_i' + P$. The positional encoded map F_i'', allows the multi-head self-attention layer to better capture the 2D spatial information. Since, the encoder in the transformer only accepts a set of vectors as input, the d channels of F_i'' are vectorized and stacked to form one feature matrix E_i of the form:

$$E_i = \begin{bmatrix} e_{i,1} \\ e_{i,2} \\ \vdots \\ e_{i,d} \end{bmatrix} \in \mathbb{R}^{d \times HW}, \tag{4.1}$$

with the vector $e_{i,j} = \text{Mat2Vec}(F_i^{''}(:,:,j)) \in 1 \times HW$, and Mat2Vec is a matrix to vector converter.

The standard encoder of transformer with $N = 6$ layers [1] is then used to generate the i^{th} encoded feature matrix $\hat{E}_i \in \mathbb{R}^{d \times HW}$. This encoder also includes a multi-head self-attention and FFN layers. The multi-head self-attention mechanism in the transformer's encoder allows the model to handle the scale differences in text instances [201].

In the *decoding phase*, as in [96] the encoded feature matrix $\hat{E}_i$, along with a fixed set of learnable embeddings, called object queries $Q \in \mathbb{R}^{N_q \times HW}$, are passed through a transformer decoder of $M = 6$ layers, where N_q denotes the maximum number of text instance queries that can appear in each input image, and $Q = [q_1^\top, \ldots, q_{N_q}^\top]^\top$ such that the k^{th} vector q_k is of size $1 \times HW$. The decoded set of feature vectors $\hat{Q} \in \mathbb{R}^{N_q \times HW}$ is then fed into the FFN layers, which consists of a three layer perceptron with a ReLU activation function plus a d-dimensional hidden layer, and a linear projection layer to predict the bounding box and class label for each query. Finally, a bipartite matching [194] is used at the end to predict the loss between the predicted and ground-truth text instances.

4.3.2 Rotated Scene Text Representation

Rectangular bounding boxes [47] (shown in Figure 5.1-a), of the form $b' = [x, y, w, h]^\top$, are considered the simplest representation of a localized horizontal text region, where (x, y) are the center point coordinates, and w and h are the box's width and height, respectively. Unfortunately, this representation falls short when dealing with irregular text regions [101] as (a) it limits the ability of a given detector to distinguish between overlapped or nearby text regions, and (b) it includes many irrelevant background areas that can affect the detector's loss function during training, and can generate noisy regions that might hinder subsequent analysis, i.e., text recognition.

To address these limitations, several works [46, 48, 49, 60, 129, 133, 134, 197–199] have used a rotated bounding box representation as shown in Figure 5.1-b. In this work, we also adopt a rotated rectangular-bounding boxes representation that embeds the box orientation angle, θ, within the box description as:

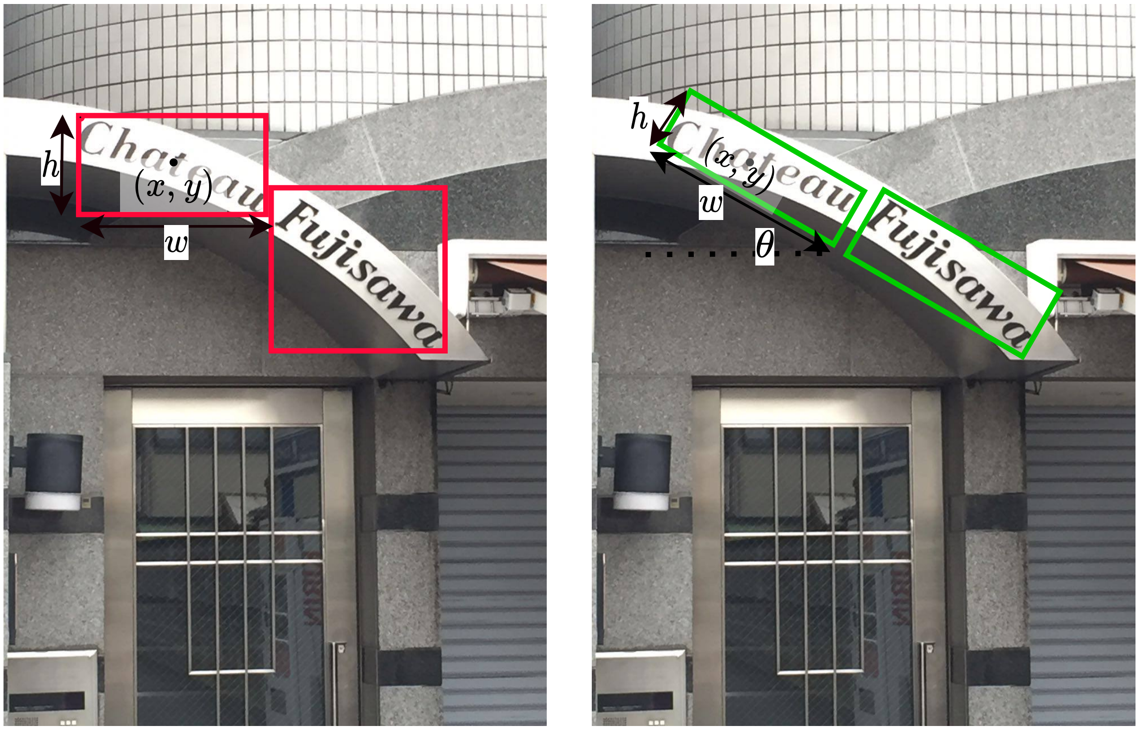
$$b = [x, y, w, h, \cos(\theta), \sin(\theta)]^\top \tag{4.2}$$

where $\theta \in [-90°, 90°)$.

4.3.3 Loss Function

To allow the transformer architecture to predict the orientation of a text region, we propose a loss function tailored to the task at hand.

Unlike [96], which uses a generic Generalized Intersection over Union (GIoU) with ℓ_1-regression [195] (shown in Figure 4.3-a), we propose a rotated-box-based GIoU loss (shown in Figure 4.3-b), along with a Smooth-$\ln$ regression based loss to properly handle rotated texts



(a) Rectangle bounding box (x, y, w, h)

(b) Rotated Rectangle bounding box (x, y, w, h, θ)

Figure 4.2: Illustrations of different techniques for representing bounding boxes for scene text detection. The above images are reproduced from public dataset [72].

as follows.

Let $\hat{b}_i$ and b_j denote the i^{th} predicted and j^{th} ground truth bounding boxes, respectively, then we define our loss function as:

$$\mathcal{L}^r_{\text{box}}(\hat{b}_i, b_j) = \lambda_1 \mathcal{L}^r_{reg}(\hat{b}_i, b_j) + \lambda_2 \mathcal{L}^r_{\text{GIoU}}(\hat{b}_i, b_j) \tag{4.3}$$

where λ_1 and $\lambda_2 \in \mathbb{R}$ are hyper-parameters, and $\mathcal{L}^r_{reg}(\cdot)$ and $\mathcal{L}^r_{\text{GIoU}}(\cdot)$ are the rotated box based loss functions that will be introduced in equation (7.3) and equation (5.7), respectively.

Smooth-$\ln$ **based Regression Loss:** It is used in computing $\mathcal{L}^r_{reg}(.)$ from equation (7.1) as it was found to be more efficient in arbitrary scene text detection than the Smooth-ℓ_1 loss [206], and is also capable of resisting more outliers, and adjusting the regressive steps [184]. As such, our adopted regression loss is defined as:

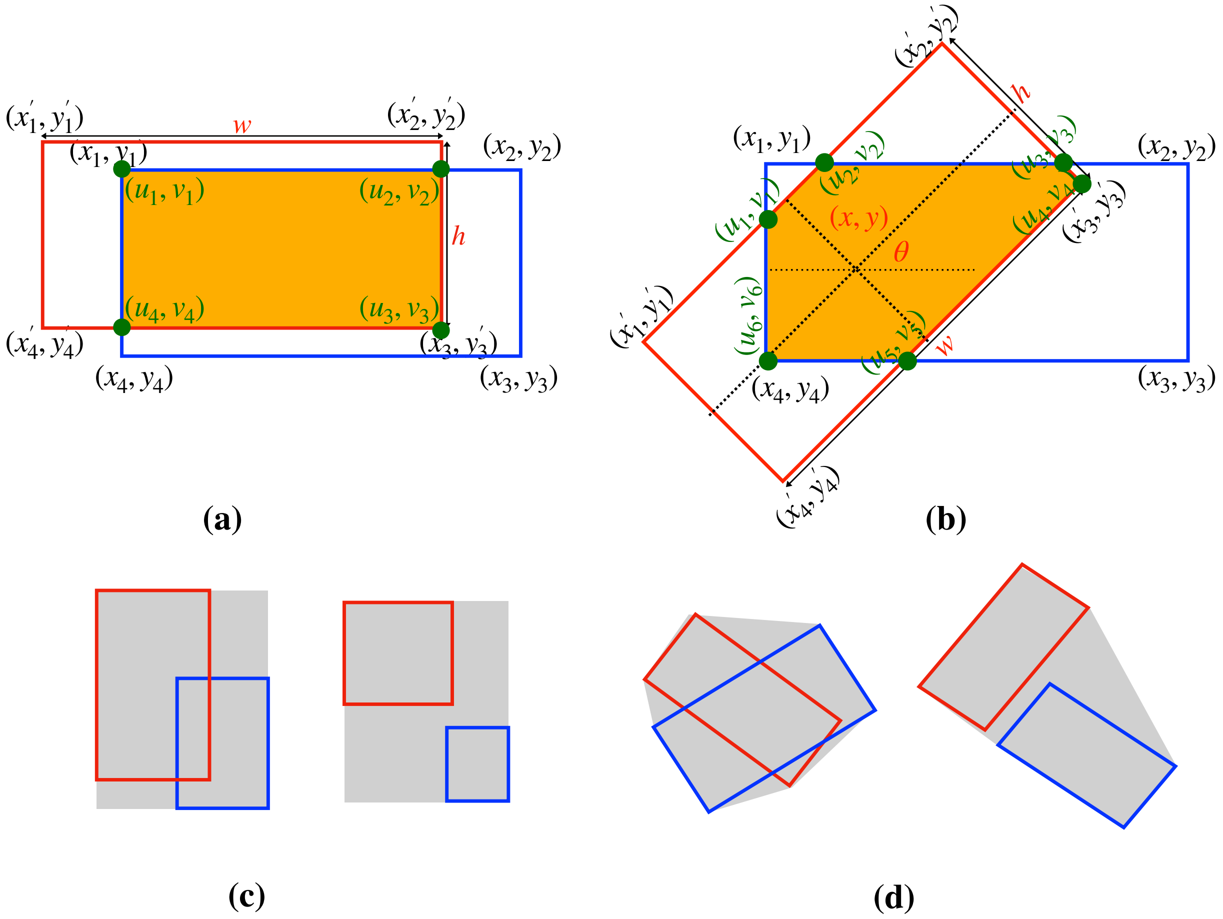
$$\mathcal{L}^r_{reg}(\hat{b}_i, b_j) = (|\Delta b_{ij}| + 1)\ln(|\Delta b_{ij}| + 1) - |\Delta b_{ij}| \tag{4.4}$$

where $\Delta b_{ij} = \hat{b}_i - b_j$ and $|\cdot|$ denotes the absolute operator.

Rotated Box based GIoU Loss: As it was shown in [96], the GIoU loss has a significant impact on the detection performance. In our model, the GIoU loss between the i^{th} predicted and j^{th} ground truth boxes, $\hat{b}_i$ and b_j respectively, is computed as:

$$\mathcal{L}^r_{\text{giou}}(\hat{b}_i, b_j) = 1 - \text{GIoU}(\hat{b}_i, b_j). \tag{4.5}$$

However, unlike [195] that uses a rectangular bounding box representation for the GIoU loss computation, we use a rotated bounding box representation that better fits text regions. The



(a) **(b)**

(c) **(d)**

Figure 4.3: Examples of the intersection (highlighted in *orange*) and convex hull (highlighted in *grey*) computation for horizontal boxes (a) and (c), and for rotated boxes (b) and (d). Note that computing the area of intersection between two rotated bounding boxes can be more complex than the horizontal case.

GIoU for two arbitrarily rotated boxes $\hat{b}_i, b_j \subseteq \mathbb{S} \in \mathbb{R}^n$ is defined as:

$$\text{GIoU}(\hat{b}_i, b_j) = \text{IoU}(\hat{b}_i, b_j) - \frac{\textbf{Area}(C \backslash (\hat{b}_i \cup b_j))}{\textbf{Area}(C)} \tag{4.6}$$

$$\text{with} \quad \text{IoU}(\hat{b}_i, b_j) = \frac{\textbf{Area}(\hat{b}_i \cap b_j)}{\textbf{Area}(\hat{b}_i \cup b_j)}, \tag{4.7}$$

and C denotes the smallest convex hull area that encloses both boxes $\hat{b}_i$ and b_j, and $\textbf{Area}(\cdot)$ is the area of a set. As illustrated in orange in Figure 4.3-b, the overlapping region of two rotated boxes constructs a polygon (p). In the next section, we will describe how we compute the different terms of Equations equation (5.7), equation (5.8) and equation (5.9).

4.3.4 Implementation Details

Computing the term $\textbf{Area}(\hat{b}_i \cup b_j)$ in equation (5.8) and equation (5.9): In order to calculate the area of an arbitrarily rotated box, b, we first obtain the corners of the box using its centered representation, i.e., $b = [x, y, w, h, \cos(\theta), \sin(\theta)]^\top$, as follows [207]:

$$\begin{aligned}
x_1 &= x + \frac{-wc_0 + hs_0}{2\gamma}, & y_1 &= y + \frac{-ws_0 - hc_0}{2\gamma}, \\
x_2 &= x + \frac{wc_0 + hs_0}{2\gamma}, & y_2 &= y + \frac{ws_0 - hc_0}{2\gamma}, \\
x_3 &= x + \frac{wc_0 - hs_0}{2\gamma}, & y_3 &= y + \frac{ws_0 + hc_0}{2\gamma}, \\
x_4 &= x + \frac{-wc_0 - hs_0}{2\gamma}, & y_4 &= y + \frac{-ws_0 + hc_0}{2\gamma}.
\end{aligned} \tag{4.8}$$

where $\{(x_i, y_i), i = 1, ..., 4\}$ are the coordinates of the box corners in counterclockwise direction (Figure 4.3), $\gamma = \sqrt{c_0^2 + s_0^2}$, and c_0 and s_0 are $\cos(\theta)$ and $\sin(\theta)$, respectively. Now, the area of a box can be computed as follows:

$$\mathbf{Area}(b) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \times \\ \sqrt{(x_2 - x_3)^2 + (y_2 - y_3)^2} \tag{4.9}$$

By using equation (4.9) and equation (4.11), the area of the union for two arbitrarily-rotated bounding boxes, i.e., the i^{th} predicted and j^{th} ground truth bounding boxes, can be computed by substituting in the following expression:

$$\mathbf{Area}(\hat{b}_i \cup b_j) = \mathbf{Area}(\hat{b}_i) + \mathbf{Area}(b_j) - \mathbf{Area}(\hat{b}_i \cap b_j) \tag{4.10}$$

Computing the term $\mathbf{Area}(\hat{b}_i \cap b_j)$ in equation (5.9): We first determine the corners of two rotated boxes $(\hat{b}_i, b_j)$ using equation (4.8), and start with one rotated box $(\hat{b}_i)$ as the candidate intersection polygon. Then, we apply the method of sequential cutting [207] for calculating the intersection between an edge in the first candidate box $\hat{b}_i$, i.e., the first line equation $\alpha_i u + \beta_i v + \tau_i = 0$, with any edge in the second box under comparison b_j, i.e., the second line equation $\alpha_j u + \beta_j v + \tau_j = 0$, by solving to obtain the coordinates of the lines intersection (u, v), where $\alpha_i, \beta_i, \tau_i$ and $\alpha_j, \beta_j, \tau_j$ are the coefficients of the lines equations that can be obtained independently using the lines corners in equation (4.8). We repeat the above process until no more edges remain and we come up with the candidate intersection polygon.

Finally, by using the vertices of resulted intersection polygon $p = \hat{b}_i \cap b_j$, its area can be calculated as follow [208]:

$$\mathbf{Area}(p) = \left| \frac{\sum_{k=1}^{n} u_k v_{\mathbf{mod}(k+1,n)} - v_k u_{\mathbf{mod}(k+1,n)}}{2} \right| \tag{4.11}$$

where $|\cdot|$ denotes the absolute operator, and $\mathbf{mod}(a, b)$ represents the modulo operator that obtains the remainder of dividing a by b, and (u_k, v_k) are the coordinates of the k^{th} vertex in the

Table 4.1: Quantitative comparison among some of the recent text detection methods on ICDAR15 [71], ICDAR17 [85] and MSRA-TD500 [181] datasets using precision (P), recall (R) and F-measure, where bold and underline denote best and second best performances respectively, and "–" refer to Non Available data.

Method	ICDAR15			ICDAR17			MSRA-TD500		
	P	R	F-measure	P	R	F-measure	P	R	F-measure
ROTDC [199]	–	–	–	–	–	–	87.00%	63.00%	74.00%
RRPN [49]	84.00%	77.00%	80.00%	–	–	–	82.00%	69.00%	75.00%
D2MO [129]	82.00%	**80.00%**	81.00%	–	–	–	77.00%	70.00%	74.00%
EAST [46]	83.30%	78.30%	80.70%	–	–	–	87.30%	67.40%	76.10%
TextBoxess++ [48]	82.20%	76.40%	79.20%	–	–	–	–	–	–
RRD [133]	85.60%	79.00%	82.20%	–	–	–	87.00%	73.00%	79.00%
FOTS [60]	85.60%	79.80%	82.80%	80.90%	57.50%	67.20%	–	–	–
MOSTD [134]	87.20%	76.70%	81.70%	83.80%	55.60%	66.80%	–	–	–
PSE-Net [89]	81.50%	79.70%	80.60%	73.77%	**68.21%**	70.88%	–	–	–
STELA [197]	88.70%	78.60%	83.33%	78.70%	65.50%	71.50%	–	–	–
R-YOLO [198]	87.00%	78.20%	82.30%	78.00%	66.30%	67.50%	90.20%	81.90%	85.80%
Proposed Method	**89.83%**	78.28%	**83.65%**	**84.75%**	63.23%	**72.42%**	**90.92%**	**83.84%**	**87.23%**

intersection polygon p.

Using equation (4.10) and equation (4.11), the IoU in equation (5.9) for two arbitrarily-rotated bounding boxes can now be obtained.

Computing the variable C in equation (5.8): For computing the convex hull of boxes, the areas highlighted by grey in Figure 4.3-c and Figure 4.3-d, we implemented the Andrew's monotone chain algorithm [209]. In this algorithm, after calculating the corner points of two rotated boxes using equation (4.8), we sort first the 8 points of two rotated boxes. Next, we go through the points and add each point to the hull. Always after adding a point to the hull, we make sure that the last line of two points in the hull does not make a counter-clockwise turn. We then repeatedly remove the second last two point from the hull, and concatenate the lower and upper hulls that gives the convex hull polygon [210]. At the end, we calculate the area of the obtained polygon using equation (4.11).

4.4 Experimental Results

As in [96], we use ResNet-50 as a backbone feature extractor. The whole network with 6 encoders and 6 decoders is trained with a batch size of 2 on four NVIDIA V100 16GB GPUs with AdamW [211] optimizer. Different from [96], we use 300 object queries instead of 100 and replace the original prediction head with our proposed rotated bounding box prediction. We first train the proposed network for ~ 50 epochs on a combination of 10k images of VISD [212] and 10k images of Unreal-Text [213] synthetic datasets and then fine-tune for ~ 200 epochs on each of the real datasets [71, 85, 181]. We apply a standard data augmentation for the training images, which involves randomly resizing between 480 and 1033, horizontal flipping, and normalizing.

4.4.1 Datasets

We evaluate our method on three public benchmark datasets that contain images of different locations like street views, traffic signs, shopping mall billboards, *etc.* These datasets also include multi-oriented text instances, which are described as follows:

ICDAR15: This dataset [71] contains 1000 images for training and 500 images for testing. The annotations of this dataset are at the word-level represented using quadrilateral boxes at the word level. This dataset is more challenging in orientation, illumination variation, and complex background of text instances than ICDAR13 [82]. Most of the images in this dataset are from indoor environments. The annotations of this dataset are represented using quadrilateral boxes, where the ground-truth annotations are in the four corner vertices format that each annotation box can be expressed as $g = [x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4]^\top$. For fine-tuning of our proposed network on this dataset, we convert the bounding boxes of this dataset from quadrilateral boxes into rotated boxes format by using the mapping function Φ as:

$$g \xmapsto{\Phi} b \tag{4.12}$$

where b is the annotation in rotated bounding box format equation (4.2), and Φ represents the `cv2.minAreaRect`[1] function in OpenCV [214], followed by a conversion that maps the rotation angle $\theta \in [-90°, 90°)$ to match the box representation definition in equation (4.2).

ICDAR17: It is a large-scale word-level multi-lingual text dataset [85] comprised of 18000 natural scene images, sorted into 7200 for training, 1800 for validation and 9000 for testing. Similar to ICDAR15, This dataset also uses quadrilateral annotations [71], which we convert to our proposed rotated boxes format with the same procedure described in the preceeding paragraph. It is noteworthy to mention that ICDAR17 is more challenging than ICDAR15 due to the varying text instances sizes, and the abundance of tiny text instances.

MSRA-TD500: This dataset [181] has been explicitly designed for arbitrarily oriented text detection, which has rotated bounding box representation in the text line level. This dataset contains 200 test and 300 training images of Chinese and English languages. This dataset's images vary from indoor (office and mall) and outdoor (street) scenes. The bounding boxes in this dataset are annotated in $('x, 'y, w, h, \theta)$ format, where $('x, 'y)$ are the coordinates of the top left corner, w and h are the width and height of the box, and θ represents the rotation angle. This format is mapped to the standard rotated box format in equation (4.2) by obtaining the center of the box, and the terms $\cos(\theta)$ and $\sin(\theta)$.

SVT: The images of the street view text (SVT) dataset [31] are collected using Google Street View camera. The images are mainly taken from outdoor locations, and it has a large number

[1] https://bit.ly/3dCauBm

of text instances with low resolution, and some images are blurry. We only use this dataset for *qualitative results* (Section 4.4.4) due to its annotations are in rectangular bounding boxes format that does not offer a fair, objective measure when used to assess rotated bounding boxes representation based methods.

4.4.2 Evaluation Metrics

For quantitative evaluation, we use the ICDAR15 IoU Metric [71], which is obtained for the i^{th} ground-truth and j^{th} detection bounding box as shown in equation (5.9), where a threshold of IoU ≥ 0.5 is used for counting a correct detection and therefore calculating the precision (P) and recall (R). As in [46, 54, 57, 58], we also use the F-measure that is a function in the precision and recall, and it is defined as follow:

$$\text{F-measure} = 2\frac{P \times R}{P + R} \tag{4.13}$$

4.4.3 Quantitative Results

In this section, a quantitative comparison for the proposed and existing state-of-the-arts methods in [46, 48, 49, 60, 89, 129, 133, 134, 197–199] on three challenging datasets, namely, ICDAR15 [71], ICDAR17 [85] and MSRA-TD500 [181] datasets, is presented.

Detection Accuracy: As depicted in Table 4.1, the proposed method offers an F-measure of 83.65% on the ICDAR15 dataset, which outperforms all the methods in comparison, including one-stage [46, 48, 89, 197, 198] and two-stage [49, 134] text detectors. By considering ICDAR17, which is a larger and more challenging dataset than ICDAR15, our proposed method also offers the highest performance in terms of precision and F-measure, than that offered by

FOTS [60], MOSTD [134], STELA [197] and R-YOLO [198]. This higher accuracy confirms the advantage of using a transformer in focusing on the regions of interest.

For the MSRA-TD500 dataset, which requires predicting line-level instead of word-level text detection, as can be seen from Table 4.1 also the proposed method provides the best detection performance compared to the considered state-of-the-art methods [46, 49, 129, 133, 198, 199].

We argue that this high performance of the proposed method is mainly attributed to the attention mechanism that allows the transformer architecture to relate among different parts of characters of a word or text-line in a given text image to make the final prediction. In addition, utilizing the GIoU loss with the rotated box representation offers the entire architecture a precise detection capability.

Loss Function Ablation: We validate the performance gain caused by our proposed loss function by comparing its detection accuracy on ICDAR15 and ICDAR17 datasets against a baseline model. The baseline model [96] uses a rectangular box based prediction head which consists of an ℓ_1 bounding box regression loss [206], and a rectangular GIoU based loss [195]. On the other hand, the proposed method uses an rotated box based prediction head, consisting of a Smooth-$\ln$ loss equation (7.3) and a rotated GIoU based loss equation (5.7) as presented in Section 4.3.3. The results in Table 4.2 show that the proposed rotated box based method outperforms the baseline by a large margin; not to mention that using non-rotated rectangular boxes for text detection exhibit poor results on the multi-oriented datasets.

Computational Speed: Using a single NVIDIA RTX 2070 (8GB GPU), our proposed model clocks at an average of 10 FPS inference speed. This speed is higher than some of the segmentation-based [54, 57, 89] and two-stage detectors [49], that require multiple stages of post-processing and regional proposal [101]. Nevertheless, some one-stage detectors *e.g.* STELA [197] and R-YOLO [198] are capable of performing inference at higher speeds when compared to transformer-based architectures [101, 204] at the cost of a slightly reduced accuracy.

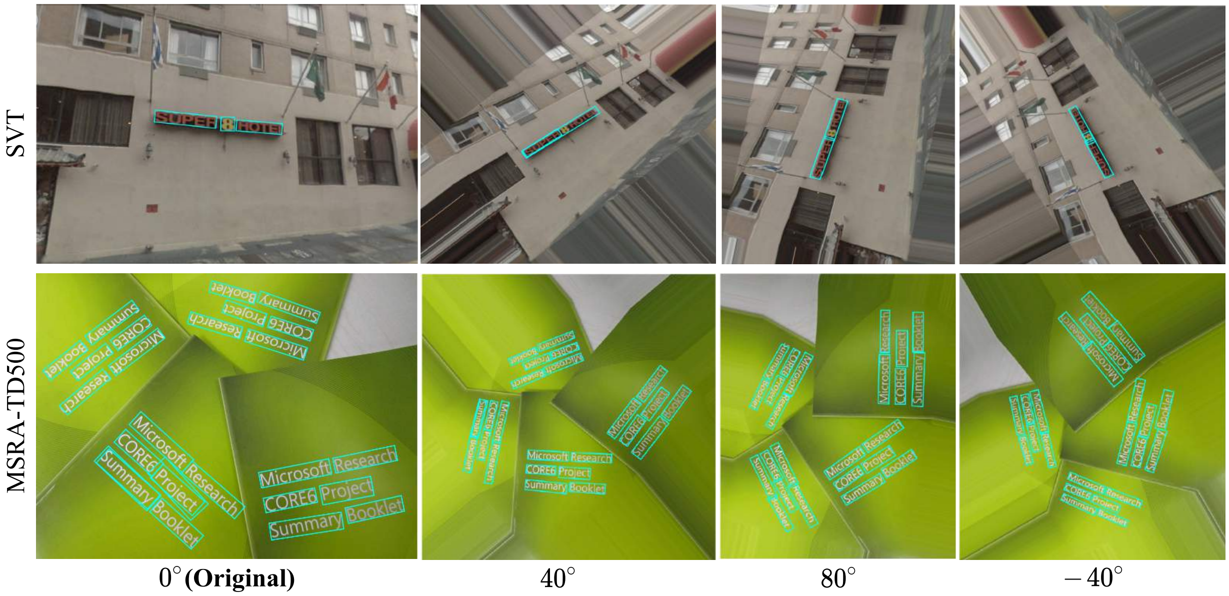


Figure 4.4: Sample qualitative results showing text detections when rotating the original image from 0° with different orientation angles. The bounding boxes of detected regions are shown with a *cyan* color. These images are taken from public datasets [81, 181].

4.4.4 Qualitative Results

Robustness to Rotated Text: We also experimented with rotated images at four different angles of $(-40^\circ, 0^\circ, 40^\circ, 80^\circ)$ and evaluated the proposed method's robustness to different text orientations. Figure 4.4 illustrates some qualitative samples from this experiment. As it can be seen, the proposed method can detect text instances of various orientations accurately.

Challenging Conditions: Figure 4.5 illustrates the proposed method's detection results for several challenging cases from ICDAR15, ICDAR17, MSRA-TD500 and SVT datasets. As it can be seen, the proposed method performs well on the first three datasets that include challenging fonts, illumination variation, in-plane rotation, and low contrast text instances. To show the generalization capability of our proposed method, we also experimented with using our ICDAR17 fine-tuned model on a different dataset, namely, the SVT dataset. It can be seen from Figure

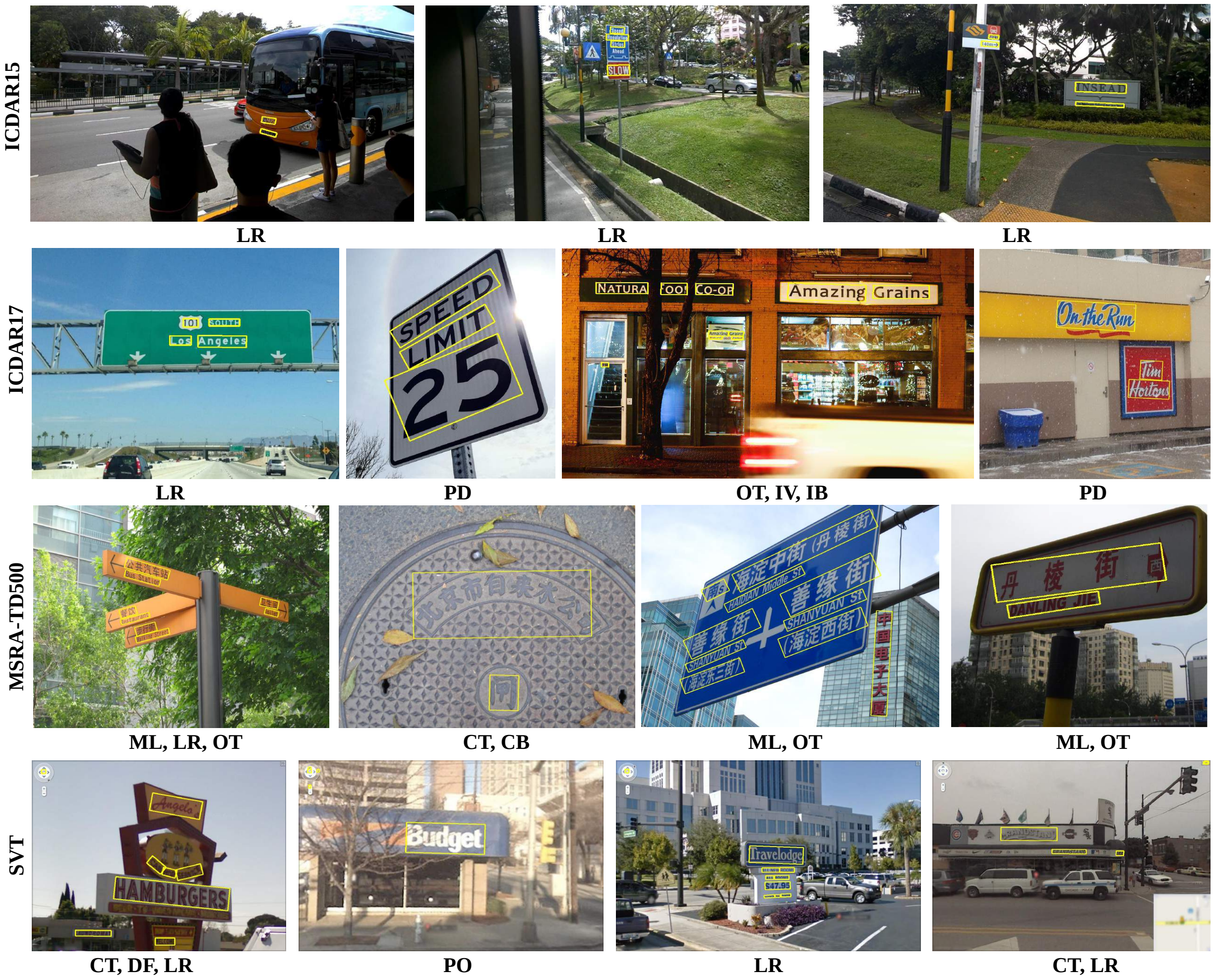
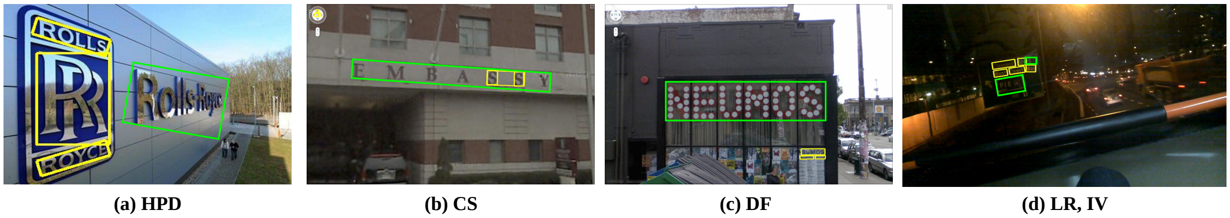


Figure 4.5: Sample qualitative results of the proposed method on some challenging examples from ICDAR15, ICDAR17, MSRA-TD500 and SVT datasets. PO: Partial Occlusion, DF: Difficult Fonts, IV: Illumination Variation, IB: Image Blurriness, LR: Low Resolution, PD: Perspective Distortion, OT: Oriented Text, and CT: Curved Text. All of the above images are reproduced from the publicly available benchmark datasets [71, 81, 85, 181].

Table 4.2: Effect of using prediction head with a loss function that is based on a rectangular (baseline method [96]) or rotated (proposed method) box representation, where the ICDAR15 [71] and ICDAR17 [85] datasets are used, and P, R and F denote precision, recall and F-measure.

Method	ICDAR15			ICDAR17		
	P	R	F	P	R	F
Baseline	69.77%	69.23%	69.50%	67.46%	**66.00%**	66.72%
Proposed	**89.83%**	**78.28%**	**83.65%**	**84.75%**	63.23%	**72.42%**



(a) HPD **(b) CS** **(c) DF** **(d) LR, IV**

Figure 4.6: Qualitative results of failed cases. The first left image is from [85], the two middle images are from [81], and the last right image is from [71]. Yellow and green bounding boxes show the correct detection and missed ground truths, respectively. HPD: High Perspective distortion, CS: Character space, DF: Difficult font, LR: Low Resolution, IV: Illumination Variation. All of the above images are reproduced from the publicly available benchmark datasets [71, 81, 85].

4.5 that the proposed model is able to handle low resolution and rotated texts without requiring any extra fine-tuning, thereby confirming the transformer's attention modules capability to reason about feature maps in different scales. While our proposed method is designed for detecting multi-orient text, it can be seen from Figure 4.5 that it is also capable of detecting curved text instances. For example, from this figure, the proposed method detected the curved line-text in the second image of MSRA-TD500 with one bounding box, and it also detected the three curved words in the first image of SVT with three separate boxes.

Figure 4.6 shows some failure cases of the proposed method. For instance, Figure 4.6-a characterizes failure cases caused by large perspective distortions, and similar text font color to the background, leading to some missed detections. Also Figure 4.6-b shows the effect of large separation between the word's characters on the detection; causing our model to only detect a subset of the whole word.

For complex fonts as shown in Figure 4.6-c, the proposed method also fails to detect the text. We attribute this missing detection to the scarcity of such fonts in the training data. Despite the severe illumination changes and small text instances shown in Figure 4.6-d, our proposed model was able to detect most instances and only missed a few. The missed detections are mainly caused by the transformer's reduced performance when detecting text of low-resolution [96, 201].

These challenging examples indicate that there is still a room to improve the proposed scheme's performance by tackling the challenges of complex fonts, illumination variations, low-resolution text and geometric distortions.

4.5 Conclusion

We have presented a transformer-based architecture for multi-oriented text detection in the wild. Extensive experiments on three challenging datasets have solidified the viability of our approach as it outperforms state-of-the-art methods, including recent rotated-bounding-box-based text detectors, in terms of precision and F-measure, while maintaining a favorable recall. Achieving these results would not have been possible without the proposed rotated bounding box representation and its associated loss function, tailored to the multi-oriented text detection problem.

In this Chapter, our proposed model only detects multi-oriented text instances. However, there are many text instances in the wild images that appear in the forms of curved or arbitrarily shapes that require polygon representation. The rotated-rectangular bounding boxes do not entirely fit these types of text boundaries. Therefore, in Chapter 5, we leverage a polygon or Bezier curve representation by extending the current transformer's architecture to detect curved or arbitrarily shaped text instances.

Chapter 5

Arbitrary-shape Text Detection using Transformers

Recent text detection frameworks require several handcrafted components such as anchor generation, non-maximum suppression (NMS), or multiple processing stages (*e.g.* label generation) to detect arbitrarily shaped text images. In contrast, we propose an end-to-end trainable architecture based on Detection using transformers (DETR), that outperforms previous state-of-the-art methods in arbitrary-shaped text detection.

At its core, our proposed method leverages a bounding box loss function that accurately measures the arbitrary detected text regions' changes in scale and aspect ratio. This is possible due to a hybrid shape representation made from Bezier curves, that are further split into piece-wise polygons. The proposed loss function is then a combination of a generalized-split-intersection-over-union loss defined over the piece-wise polygons, and regularized by a Smooth-$\ln$ regression over the Bezier curve's control points.

We evaluate our proposed model using Total-Text and CTW-1500 datasets for curved text, and MSRA-TD500 and ICDAR15 datasets for multi-oriented text, and show that the proposed method outperforms the previous state-of-the-art methods in arbitrary-shape text detection tasks.

5.1 Introduction

Scene text detection is the process of accurately localizing text instances in wild images; it is an essential component that enables various practical applications such as text recognition, blind navigation, and topological mapping to name a few [101, 215]. While recent text detection methods [46–48, 54, 197, 198, 216] have shown reliable performance on horizontal and multi-oriented text, accurate detection of texts in an arbitrary geometric layout is still an open-ended problem.

The majority of State-Of-The-Art (SOTA) arbitrary shape text detectors are built on object detection or segmentation frameworks, and can be categorically divided into two classes: segmentation-based [4, 53, 54, 57, 89, 217, 218] and regression-based [46, 48, 100, 217, 219–221]. The segmentation-based methods [4, 53, 54, 57, 89, 216, 218, 222] encode text instances at a pixel level, and aggregate the resulting pixels to generate a segmentation mask per text instance. While they are flexible in detecting arbitrarily shaped texts, they require complex architectures and computationally expensive post-processing steps to be able to detect quadrilateral and curved text instances. This results in a high inference time, and increased difficulty to train them, which in turn requires extensive amounts of training data.

On the other hand, regression-based methods [46, 48, 100, 217, 219–221] are inspired from generic object detection frameworks [5, 9, 10, 74, 223], and model text instances as objects. Unlike segmentation-based methods, they output bounding boxes around the text regions using relatively simple architectures; as such, they are fast and easy to train. While some of these methods can achieve good performance on irregular texts, appropriately formulating anchors to fit arbitrarily-shaped text instances is not a solved problem, and requires post-processing steps (*e.g.*, NMS) to achieve a reliable final detection.

Recent advancements in object detection enabled transformer frameworks [224–226] like DETR (Detection Transformer) [96] to eliminate the need for many of the existing handcrafted post-processing steps such as anchor generation, and non-maximum suppression (NMS) from the object detection pipeline [9, 10, 73, 74], all while achieving superior performance. For example, Raisi *et al.* [2], leveraged the DETR [96] architecture for multi-oriented scene text detection and achieved SOTA performance in some benchmark datasets. Nevertheless, DETR has difficulties detecting small objects and suffers from a slow convergence rate. To address these issues, [201] introduced a deformable attention module to focus on a sparse small set of prominent key elements, thereby performing better in terms of average precision, and obtaining faster convergence during training. However, [96, 201] frameworks can only generate rectangular bounding boxes around the detected objects, and cannot handle arbitrarily shaped texts.

In contrast to [96, 201], we propose an end-to-end transformer-based object detection architecture that can directly localize multi-oriented or curved text instances in the given image. Our proposed text representation is tailored to the scene text detection task as it predicts 8 or 16 control points of a quadrangle box or Bezier curve respectively, for each text region; this allows our method to overcome the drawbacks of directly deploying a generic object detector as in [96] that predicts only 4 points of every rectangular box.

Our main contributions can be summarized as follows: (1) We propose an end-to-end trainable transformer-based framework for arbitrary shaped text detection; the proposed architecture can directly output fixed vertices for the Bezier curves that bound multi-oriented and curved text shapes. This is achieved by modifying the prediction head of the baseline pipeline via designing a new text detection technique that aims to infer n-vertices of a polygon or the degree of a Bezier curve that is better suited for irregular-text regions; and (2) We propose a loss function that is accurate in measuring the changes in scales and aspect ratios of the detected text regions, and accepts arbitrary shapes of text instances using both Bezier curves and polygon bounding boxes. (3) We study the effect of different vertices of polygon representation with the transformer's architecture on arbitrary shape text instances.

5.2 Related Work

5.2.1 Segmentation-based Methods

Segmentation-based methods typically decompose text instances in a given image into pixels/segments that are then aggregated into an output mask. Segmentation methods cover a large body of research including [4, 53, 54, 57, 89, 216, 218, 222] to name a few. For example, PixelLink [54], adopted a segmentation framework of SSD [10] with a FCN [3] to predict the relationship links between pixels of text and non-text instances, to localize similar adjacent pixels, and to group them. TextSnake [53] proposed to detect the arbitrary shape of text instances with ordered disks and text centre lines. To efficiently separate close text instances, PAN [4] made use of an efficient instance semantic segmentation framework that selectively aggregates text pixels according to their embedding distances, resulting in a model that can handle arbitrary shape text regions. PSENet [89] expanded the final local segmented areas from small kernels to predefined scales, allowing close text instances to be separated using a progressive scale algorithm. TextField [218] deployed a deep direction field approach to generate candidate text parts, and to link neighboring pixels. Different from mentioned word-level detectors, CRAFT [57] proposed to detect and connect character regions to generate polygons of arbitrary-shape text instances; this was achieved by training a U-Net [223] type framework in a weekly semi-supervised learning process.

5.2.2 Regression-based Methods

Regression based methods such as [46, 48, 100, 217, 219–221, 227, 228] are mostly inspired by general object detectors (*e.g.*, Faster R-CNN [9] and SSD [10]); they directly regress the entire word or text-line with arbitrary shape in an image at object level.

Early regression-based methods such as TextBoxes++ [48] and EAST [46] used SSD's [10] architecture to detect text regions with rotated rectangles or quadrilateral descriptions. More recently, [2] extended DTER's [96] architecture to output rotated rectangular boxes directly and achieved SOTA performance in multi-oriented benchmark datasets. However, these representations ignore the geometric traits of the arbitrary shape of curved texts and end up producing considerable background noise.

To better fit arbitrary shaped text, more advanced methods proposed the use of polygons; For example, LOMO [217] took advantage of both segmentation and regression-based architectures by utilizing Mask-RCNN [74] as their base framework, and introducing iterative refinement and shape expression modules to refine bounding box proposals of irregular text regions. TextRay [219], leveraged the SSD framework by eliminating the anchor design, and detecting polygons in the polar coordinate system to better represent arbitrary shape text instances. ABC-Net [100, 221] build on a ResNet-50 [164] feature extractor with a Feature Pyramid Network (FPN) [5] as their backbone, and introduce a Bezier curve representation in order to detect multi-oriented and curved scene text instances. FCENet [220] extends the base network of [100] by performing some post-processing steps like Inverse Fourier Transforms (IFT) and NMS to reconstruct text contours of arbitrary-shape text instances.

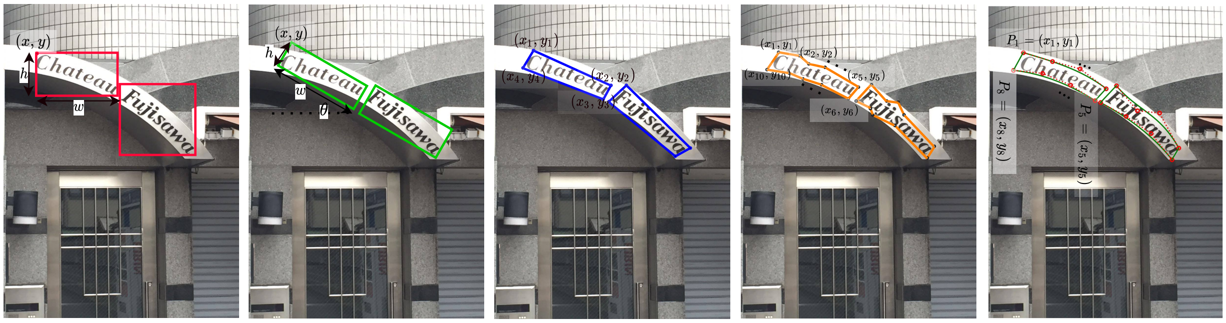
5.3 Methodology

Our proposed framework leverages an efficient and fast-converging encoder-decoder detector, namely Deformable-DETR [201], as the main detection architecture. A CNN backbone extracts first multi-scale feature maps from the input. After attaching positional encodings to the resulted features, they fed into the transformer encoder, which outputs refined multi-scale features. Then A fixed small set of learnable embedding called object queries is passed through the transformer decoder parallelly. The decoder generates instance-aware query embeddings, which are then fed into a prediction head that directly converts the decoders' outputs into each query's class and bounding box set. The proposed network is trained by a Bipartite matching loss that utilizes the Hungarian matching algorithm [194] to compare a one-to-one mapping between N queries and N ground-truths [96].

In this work, instead of computing 4 scalars that correspond to the (x, y, w, h) coordinates of the centers (x, y) and the height (h) and width (w) of the box, we extend the number of predicted variables to $2 \times n$ scalars that correspond to the coordinates of the n control points of a Bezier curve in equation (5.2) and the k polygon points in equation (5.10). To train the network, we modify the regression head, along with the loss and matching functions as described in Section 5.3.2.

5.3.1 Text Regions Representations

Rectangular Bounding Boxes: Rectangular bounding boxes are one of the most intuitive representations of horizontal text regions; as shown in Figure 5.1(a), a bounding box $b = [x, y, w, h]^\top$ can encase the text region by simplify defining (x, y) as the bounding box's center point coordinates, and w, h representing the box's width and height respectively. However, rectangular bounding boxes suffer from several limitations that render them inadequate for irregular text representations; some of these limitations include: (a) limited ability to distinguish among overlapped or nearby text regions, (b) they can not precisely bound marginal-text, and (c) they include large irrelevant background areas that can affect the detector's loss function during trainin and



(a) Rectangle bounding box (x, y, w, h)

(b) Rotated Rectangle bounding box (x, y, w, h, θ)

(c) Quadrilateral bounding box $(x_i, y_i) \mid i = 1, 2, 3, 4$

(d) Polygon bounding box $(x_i, y_i) \mid i = 1, 2, ..., n$

(e) Bezier curve bounding box $P_i = (x_i, y_i) \mid i = 1, 2, ..., 8$

Figure 5.1: Illustrations of different techniques for representing bounding boxes for scene text detection. The Bezier curves in (e) better draw smooth lines between arbitrary shaped text instances with fixed 8 control points that are more suitable for training our proposed framework. Furthermore, we can better rectify the detected regions in (e), which later lead to a more accurate word recognition performance [100]. The above images are from public dataset [72].

can generate noisy regions for subsequent analysis, i.e., text recognition. To address these limitations, arbitrary shaped text regions are typically represented using other categories of bounding boxes as shown in Figure 5.1(b)-(d) which all aim to bound text of arbitrary orientations and shapes.

Quadrilateral Representation: A Quadrilateral bounding box can be described as follows:

$$b = [x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4]^\top, \tag{5.1}$$

where (x_i, y_i) are the four vertices of the quadrilateral arranged in a clockwise order. The added dimensions allow the quadrilateral to precisely represent various types of text regions including horizontal, multi-oriented, and slight-round texts. It is then no surprise that they have been used in many text detection benchmark datasets [71, 85, 229],

Polygon Representation: Polygons are a natural extension of quadrilaterals, where the number of points is increased from 4 to $n - point$ vertices; the bounding box defined by the polygon vertices can then be defined as $b = [(x_i, y_i)|i = 1, 2, ..., n]^\top$, which can essentially better follow the boundary of a text region, and accordingly represent any arbitrarily-shaped text.

Bezier Curves: Similar to [100], we adopt Bezier curve to represent the boundaries of the text regions in equation (7.1), where an example of this representation is shown in Figure 5.1(e). Unlike polygons, a Bezier curve is a parametric curve of degree n, $Y_n(t)$, which is used to draw smooth lines between text bounds. The general form of an n-degree Bezier curve can be expressed in terms of a set of $n + 1$ control points $\{P_i\}_{i=0}^{n}$ as:

$$Y_n(t) = \sum_{i=0}^{n} B_{i,n}(t) P_i, \qquad 0 \leq t \leq 1 \tag{5.2}$$

where $P_i = (x_i, y_i | i = 0, 1, \ldots, n)$, t is a normalized independent variable that is used to move along the Bezier curve with a step that determines the smoothness of the curve, and $B_{i,n}(t)$ denotes the ith version of the $n-$degree Bernstein Polynomials [230] that are defined using:

$$B_{i,n}(t) = \binom{n}{i} t^i (1 - t)^{n-i}, \qquad i = 0, 1, \ldots, n \tag{5.3}$$

and $\binom{n}{i}$ is the Binomial coefficient.

While a 3rd-degree Bezier curve, defined by 4 control points, is effective in representing one side of an arbitrary shape text, another 3rd-degree Bezier curve is needed the represent the opposite side (as shown in Figure 5.1(e)), bringing the total number of control points needed to fully represent text boundaries to 8. The 8 control points are then computed during regression and prediction as: Therefore, as shown in Figure 5.1(e), to cover the main two sides of arbitrary shape text, we adopt a pair of 3rd-degree Bezier curves, with 8 control points in total to represent text boundaries during regression and prediction phases as follow:

$$(P_{ij} = x_{ij}, y_{ij} | i = 0, 1, ..., 3, j = 0, 1) \tag{5.4}$$

where b_i in equation (7.1) are the vertices of the Bezier curve obtained using equation (5.2).

5.3.2 Proposed System

Similar to [100], we adopt Bezier curves to represent the boundaries of arbitrary shape text instances. To achieve this, we modify the prediction head of deformable DETR's architecture [201] to output 16 parameters that represent the Bezier control points. However, unlike [96] and [201] that use a generic Generalized Intersection over Union (GIoU) with ℓ_1-regression [195] (shown in Figure 5.1(a)), we propose a split GIoU loss for Bezier control points of equation (5.4) (shown in Figure 5.2), along with a Smooth-$\ln$ regression based loss [2].

The intuition behind the split GIoU is to better compute the difference (loss) between the ground truth and estimated text boundaries. While GIoU can be computed over the Bezier curves, it is computationally inefficient and more complex to calculate the area of intersection between two Bezier curves. To mitigate this, we split the Bezier curve computed from the regressed control points into several rectangles. The piece-wise GIoU over the rectangles can then be computed efficiently, and the overall set of rectangles defining one text instance are smoothed with the regression loss function over the Bezier curve control points.

The bounding box loss function of [96] uses a linear combination of ℓ_1 and GIoU loss. Let $\hat{b}_i$ and b_j denote the i^{th} predicted and j^{th} ground truth bounding boxes, respectively, then we define our loss function as:

$$\mathcal{L}^B_{\text{box}}(\hat{b}_i, b_j) = \lambda_1 \mathcal{L}^B_{reg}(\hat{b}_i, b_j) + \lambda_2 \mathcal{L}^B_{\text{GIoU}}(\hat{b}_i, b_j) \tag{5.5}$$

where λ_1 and $\lambda_2 \in \mathbb{R}$ are hyper-parameters, and $\mathcal{L}^B_{reg}(\cdot)$ and $\mathcal{L}^B_{\text{GIoU}}(\cdot)$ are the Bezier-curved loss functions based on regression and GIoU. For regression, we use the Smooth-$\ln$ based Regression Loss as in [2]. The regression loss is then defined as:

$$\mathcal{L}^B_{reg}(\hat{b}_i, b_j) = (|\Delta b_{ij}| + 1)\ln(|\Delta b_{ij}| + 1) - |\Delta b_{ij}| \tag{5.6}$$

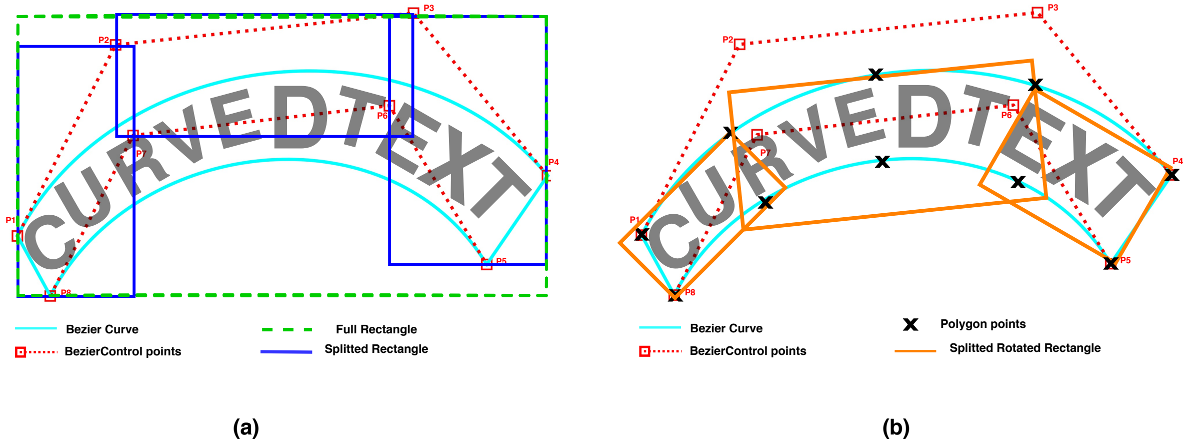


Figure 5.2: Illustration of the proposed methods. The control points (dotted lines) in (a) and polygon vertices ('x' points) in (b) are predicted directly by the network. The entire rectangle (green dash lines) in (a) is used for Full GIoU calculation. The three split rectangles (blue lines in (a)) and rotated rectangles (orange lines in (b)) make the GIoU and then the Bezier curves (cyan line) and polygon vertices to better bound to high curved text instances.

where $\Delta b_{ij} = \hat{b}_i - b_j$ and $|\cdot|$ demonstrates the absolute operator. The second part of equation (7.1) consists of GIoU loss, which plays an important role in the framework of detection using transformers [96]. The GIoU loss is computed as:

$$\mathcal{L}^{B}_{\text{giou}}(\hat{b}_i, b_j) = 1 - \text{GIoU}(\hat{b}_i, b_j), \tag{5.7}$$

The GIoU for two arbitrarily bounding boxes $\hat{b}_i, b_j \subseteq \mathbb{S} \in \mathbb{R}^n$ can be defined as follows:

$$\text{GIoU}(\hat{b}_i, b_j) = \text{IoU}(\hat{b}_i, b_j) - \frac{\mathbf{Area}(C \backslash (\hat{b}_i \cup b_j))}{\mathbf{Area}(C)} \tag{5.8}$$

$$\text{with} \quad \text{IoU}(\hat{b}_i, b_j) = \frac{\mathbf{Area}(\hat{b}_i \cap b_j)}{\mathbf{Area}(\hat{b}_i \cup b_j)}, \tag{5.9}$$

where C shows the smallest area that encloses both prediction and ground-truth boxes $\hat{b}_i$ and b_j, and $\mathbf{Area}(\cdot)$ denotes the area of a set. To compute the GIoU loss for 16 Bezier points of the architecture, we start by calculating the rectangular bounding box that bound all control points of equation (5.4) in the ground truth and prediction outputs of the network. To better fit to high curved text instances in arbitrary shape benchmarks [72, 86], we then split the Bezier control points into several axis-aligned rectangular bounding boxes, where the first rectangular box is computed from P_1, P_2, P_7, P_8 Bezier control points, the second and third boxes are also obtained from P_2, P_3, P_6, P_7 and P_3, P_4, P_5, P_6, respectively. This process is summarized in Figure 5.2(a).

5.3.3 $n-$point Polygon Ground Truth Generation

The Bezier control points move outside of the image when the text appears near the margin of an image, requiring negative values of (x, y). Since the final prediction head of [96, 201] only outputs positive values, it fails to precisely detect the mentioned text instances. To address this issue, instead of using the Bezier control points directly as shown in Figure 5.2(a), we first calculate the 3rd-degree Bezier curve for each side of the text, defined by 4 control points. We then recalculate the $n-$polygon vertices (as illustrated in Figure 5.2(b)) by uniformly sample n_v points as follows:

$$p_k = \sum_{i=0}^{n=4} P_i B_{i,n} k/n_v, \tag{5.10}$$

where p_k demonstrates the new k-th sampled polygon points, P_i indicates the i-th Bezier control points and n_v shows the polygon points used for sampling. $B_{i,n}$ represents the $n-$degree Bernstein Polynomials [230] as described in equation (5.3).

5.4 Experimental Evaluation

We evaluate the performance of our proposed system, on public scene text detection datasets [71, 72, 85, 86] that cover a wide range of challenging scenarios. We also perform a set of quantitative and qualitative experiments to benchmark the SOTA text detection [4, 46, 53, 54, 57, 89, 100, 118, 216–222, 227, 228, 231, 232] techniques against our proposed model. Following the criteria used in [100] to evaluate performance on arbitrary shaped text, and the evaluation metrics [71, 85] used to evaluate ICDAR's multi-oriented text, we report on the Recall, Precision, and F-measure of the various methods.

5.4.1 Implementation Details

We adopt the recent Deformable DETR [201] model with a ResNet-50 [164] backbone as our base object detector architecture. The number of object queries are set to 300 and an AdamW [211] optimizer is used to optimize the parameters of the model. We use a horizontal flip and and resize the images similar to [201] for augmentation. All our proposed models are pre-trained on synthetic datasets as in [100] for 20 epochs with a batch size of 2 per GPU using 4 Tesla V100 GPUs with a learning rate (LR) of 1×10^{-4}. We follow [201] for other hyper-parameters during pre-training. During fine-tuning, we adopt a different LR schedule and train for about 200 epochs for both the Total-text and CTW-1500 datasets, and drop the LR by a factor of 10 after 70 epochs. As for ICDAR15, we further pre-train the models using about $10,000$ images of ICDAR17 [85] dataset for 50 epochs and then fine-tune for about 300 epochs to ensure the training converges. For calculating the rotated version of bounding box loss function, we used the method described in [2].

5.4.2 Datasets

We make use of several recently published and challenging datasets, that can be categorized into multi-oriented text datasets, ICDAR15 [71] and MSRA-TD500 [181] with quadrilateral representation (Figure 5.1(c), and arbitrary-shaped text datasets, Total-Text [72] and CTW-1500 [86] with n-vertices polygon representation as shown in Figure 5.1(d).

5.4.3 Comparisons with SOTA Methods

In this section, we first compare the proposed model with the SOTA methods [4, 46, 54, 57, 89] on two popular datasets containing curved text: Total-Text [72] and CTW-1500 [86]. We evaluate the datasets on two models: (1) that uses 16 control points of the Bezier curve with three splits rectangularly (Figure 5.2(a)) and (2) that uses 20-points polygon with three splits rotated rectangularly (Figure 5.2(b)).

Arbitrary-Shape Text Datasets: We first compare our baseline and proposed models on two popular benchmarks, Total-Text and CTW-1500, containing curved text and have n-vertices polygon annotations.

Results of Total-Text: As seen in Table 5.1, both proposed models achieved the best performance in terms of Recall and Precision compared to other segmentation-based and regression-based methods. The second model outperformed the first model, overall by ~ 0.6. The effectiveness of our contributions are evident in the qualitative results of Figure 5.3 as it demonstrates how the Bezier curve and 20-point polygons estimated by our proposed methods can better fit more challenging arbitrary-shaped text instances.

Results of CTW-1500: Despite the highly curved text instances in this dataset, our first method surpassed other SOTA systems, achieving the best precision of 88.3% and a F-measure of 86.1%. The second method also performed better than the first on this dataset, which shows how effectively using $20-$points polygon can bound high curved text-line instances. The qualitative results using the proposed methods for some challenging samples of the CTW-1500 [86] dataset are shown in Figure 5.4, where the proposed methods perform better than ABC-Net [100] and TextRay [219] and exhibit competitive results in some cases against FCENet [220] that uses a smoother curve. The second model that uses 20-points of a polygon with split rotated rectangular outperformed the first model, by overall ~ 0.6. It is worth mentioning that the Bezier curve

Table 5.1: Comparison of the detection results on Total-Text, CTW-1500, ICDAR15, and MSRA-TD500 datasets with recent regression and segmentation based methods. The best performance is highlighted in **bold**. ”R”, ”P”, and ”F” denote Recall, Precision, and F-measure respectively.

Methods	**Total-Text**			**CTW-1500**			**MSRA-TD500**			**ICDAR15**		
	R	**P**	**F**	**R**	**P**	**F**	**R**	**P**	**F**	**R**	**P**	**F**
SegLink [118]	-	-	-	-	-	-	70.0	86.0	77.0	76.8	73.1	75.0
Textboxes++ [48]	-	-	-	-	-	-	-	-	-	78.5	87.8	82.9
EAST [46]	50.0	36.2	42.0	49.7	78.7	60.4	67.4	87.3	76.1	78.3	83.3	80.7
TextSnake [53]	74.5	82.7	78.4	77.8	82.7	80.1	73.9	83.2	78.3	84.9	80.4	82.6
TextDragon [231]	75.7	85.6	80.3	82.8	84.5	83.6	-	-	-	83.7	**92.4**	87.8
TextField [218]	79.9	81.2	80.6	79.8	83.0	81.4	75.9	87.4	81.3	80.0	84.3	82.4
PSENet-1s [89]	77.9	84.0	80.9	79.7	84.8	82.2	-	-	-	84.5	86.9	85.7
Seglink++ [227]	80.9	82.1	81.5	79.8	82.8	81.3	-	-	-	80.3	83.7	82.0
LOMO [217]	79.3	87.6	83.3	76.5	85.7	80.8	-	-	83.5	**91.3**	87.2	
CRAFT [57]	79.9	87.6	83.6	81.1	86.0	83.5	78.2	88.2	82.9	84.3	89.8	86.9
PAN [4]	81.0	89.3	85.0	81.2	86.4	83.7	83.8	84.4	84.1	81.9	84.0	82.9
DDRG [228]	84.9	86.5	85.7	83.0	85.9	84.5	82.3	88.0	85.1	84.7	88.5	86.5
TextRay [219]	77.9	83.5	80.6	80.4	82.8	81.6	-	-	-	-	-	-
ABC-Net-v1 [100]	81.3	87.9	84.5	78.5	84.4	81.4	-	-	-	-	-	-
FCENet [220]	82.5	89.3	85.8	83.4	87.6	85.5	-	-	-	82.6	90.1	86.2
CounterNet [232]	83.9	86.9	85.4	84.1	83.7	83.9	-	-	-	86.1	87.6	86.9
DB [222]	82.5	87.1	84.7	80.2	86.9	83.4	79.2	**91.5**	84.9	82.7	88.2	85.4
ABC-Net-v2 [221]	84.1	**90.2**	87.0	83.8	85.6	84.7	81.3	89.4	85.2	86.0	90.4	**88.1**
Our model-1	85.7	89.4	87.5	84.0	88.3	86.1	84.5	87.4	85.9	81.5	89.3	85.2
Our model-2	**86.4**	89.1	**87.8**	**85.3**	**89.2**	**87.2**	**85.0**	88.1	**86.5**	83.1	90.2	86.5

model showed poor performance in detecting text instances near the margin of the images. The second proposed model performed better in these types of text instances.

Multi-oriented Text Datasets: We also compare the detection performance of the transformer's architecture using the Bezier curve for multi-oriented datasets of MSRA-TD500 and ICDAR15. For this purpose, we use the baseline-4 with Smooth-$\ln$ regression and rectangular GIoU loss for training of Bezier curve and 20-points polygon models because of the quadrilateral annotation in these datasets. It is worth mentioning that splitting the GIoU in these datasets does not affect to the final performance.

Results of MSRA-TD500: As shown in Table 5.1 our proposed methods achieves SOTA results in terms of Recall of 85.0% and F-measure of 86.5%. Our model-2 that uses a 20-point polygon representation outperformed the Bezier curve representation and it surpasses the previous best method by a relatively significant margin of $\sim 4\%$ and $\sim 1.5\%$ on the Recall and F-measure performances, respectively.

Results of ICDAR15: As shown in Table 5.1, our both models achieve competitive results with SOTA detection models in ICDAR-15 datasets. When using a $20-$points polygon our models outperform the Bezier curve representation with 16 control points. Nevertheless, both models' F-measure and recall performances were lower than some of the best-performing approaches on this dataset. We believe this reduced performance is related to the transformer architecture's limited capabilities in detecting low-resolution and small text instances.

5.4.4 Ablation Study

To assess the added value of the various components in our model, we performed an extensive ablation study on Total-Text and CTW-1500 as demonstrated in Table 7.2.

Table 5.2: Ablation study on the effects of the various proposed components on the F-measure metric for Total-Text [72] and CTW-1500 [86] datasets. R and RR denote the rectangle and rotated-rectangle, respectively.

Method	**Reg**	**GIoU**	**# split**	**Total-Text**	**CTW-1500**
Baseline-1	✓	-	-	79.01	78.25
Baseline-2	✓	-	-	79.52	78.63
Baseline-3	-	✓	R(1)	82.46	80.83
Baseline-4	✓	✓	R(1)	83.41	83.70
Our model-1	✓	✓	R(3)	**87.50**	**86.10**
Our model-2	✓	✓	RR(3)	87.80	87.20

We started the experiments by eliminating the GIoU loss and training the model with ℓ_1 loss only; the model achieved a F-measure performance of 79.01% and 78.25% for Total-Text and CTW-1500 datasets, respectively. We then replaced the ℓ_1 with the Smooth-$\ln$ loss, yielding a slightly improved F-measure.

We found that only using the GIoU loss defined over the entire rectangle led to further performance boosts, which in turn was further improved when we combined both GIoU and Smooth-$\ln$ losses. Then, we evaluated the split version of GIoU loss with 3 rectangles achieved the best performance by improving $\sim 4\%$ and $\sim 2.5\%$ for Total-Text and CTW-1500 datasets in the ablation study.

Finally, we conducted another experiment by using a $20-$points polygon representation with 3 split rotated rectangles and rotated loss functions as shown in Figure 5.2(b). Applying this system on the network's head outperformed the first model, especially on the CTW-1500 dataset by a margin of $\sim 1\%$. It is worth mentioning that using a split version of the rotated rectangle does not affect the Bezier curves' F-measure performance on the mentioned datasets. The qualitative results on some challenging cases of Total-Text (shown in Figure 5.3) confirm the effectiveness of the proposed methods with split GIoU when compared to only using a single rectangular GIoU.

Table 5.3: Ablation study of our model using different points of Polygon vs. Bezier (16 points) representation for Totat-Text.

Method	**# points**	**Recall**	**Precision**	**F-measure**
Bezier curve	**16**	64.5	71.3	67.7
Polygon	8	51.7	59.6	55.4
Polygon	16	62.0	68.6	65.1
Polygon	**20**	64.2	**73.5**	**68.5**
Polygon	24	63.6	67.6	65.5
Polygon	40	**64.8**	59.7	62.1
Polygon	80	20.4	58.7	30.3
Our model-1	16	**66.2**	74.3	70.0
Our model-2	20	66.1	**76.6**	**70.9**

We also trained the Total-Text [72] dataset with different fixed $8, 16, 20, 24, 40, 80$-*points* of polygon representation and compared it with Bezier curve representation in Table 5.3. The reason for using the Total-text dataset in this experiment is that it contains challenging curved and oriented text instances at the word level. For a fair comparison, we used a model with similar loss function and split rectangle in Table 7.2 and the whole training set of Total-text. We trained both models for 300 epochs. As seen, the Bezier curve with 16 control points and $20-$points polygon representation are more suitable for detection than using other vertices of a polygon. In addition, we continue experimenting by training the first and second models that use three split GIoU with 16 Bezier control points, and three splits rotated GIoU with 20-point polygon representations, respectively, which the second model performed better in terms of precision and F-measure.

5.5 Conclusion

We have presented an arbitrary-shape text detector that directly outputs the bounding boxes of arbitrary shape text instances in natural images. The proposed framework builds on DETR's

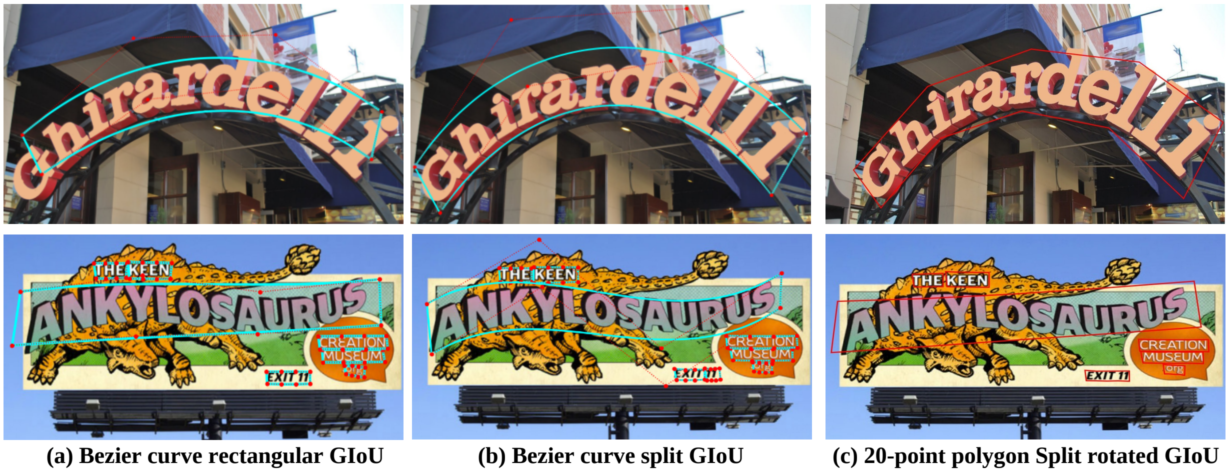

(a) Bezier curve rectangular GIoU (b) Bezier curve split GIoU (c) 20-point polygon Split rotated GIoU

Figure 5.3: Compare the effect of using split GIoU and baseline GIoU. As seen, the proposed methods with split GIoU in Table 7.2 better fits the highly curved text instances. The above images are reproduced from public dataset [72].

architecture to output a fixed set of Bezier curve's control vertices and $n-$points of polygon, which in turn can be used to represent arbitrary polygons of curved and multi-oriented texts. For accurate detection, especially on different challenging arbitrary shape text instances in irregular-text datasets such as Total-Text and CTW-1500, we have also proposed a split version of the Bezier curve and $n-$points of polygon computed from the regressed control points into several rectangles to better fit to the highly curved texts.

We have validated our proposed system using several quantitative and qualitative experiments on challenging benchmark datasets, including multi-oriented quadrilateral annotated text and curved text with n-vertex polygons representations. We have also compared the performance of our proposed method with SOTA scene text detection methods, and demonstrated the superior performance of our models on arbitrary shape text and multi-oriented text benchmarks. Our best proposed model that uses a 3 splits rotated rectangular loss function achieves the best F-measure performance of 87.8% and 87.2% for Total-Text and CTW-1500 datasets, respectively. Our system also exhibits SOTA performance in Recall (85.0%) and F-measure (88.1%) on the

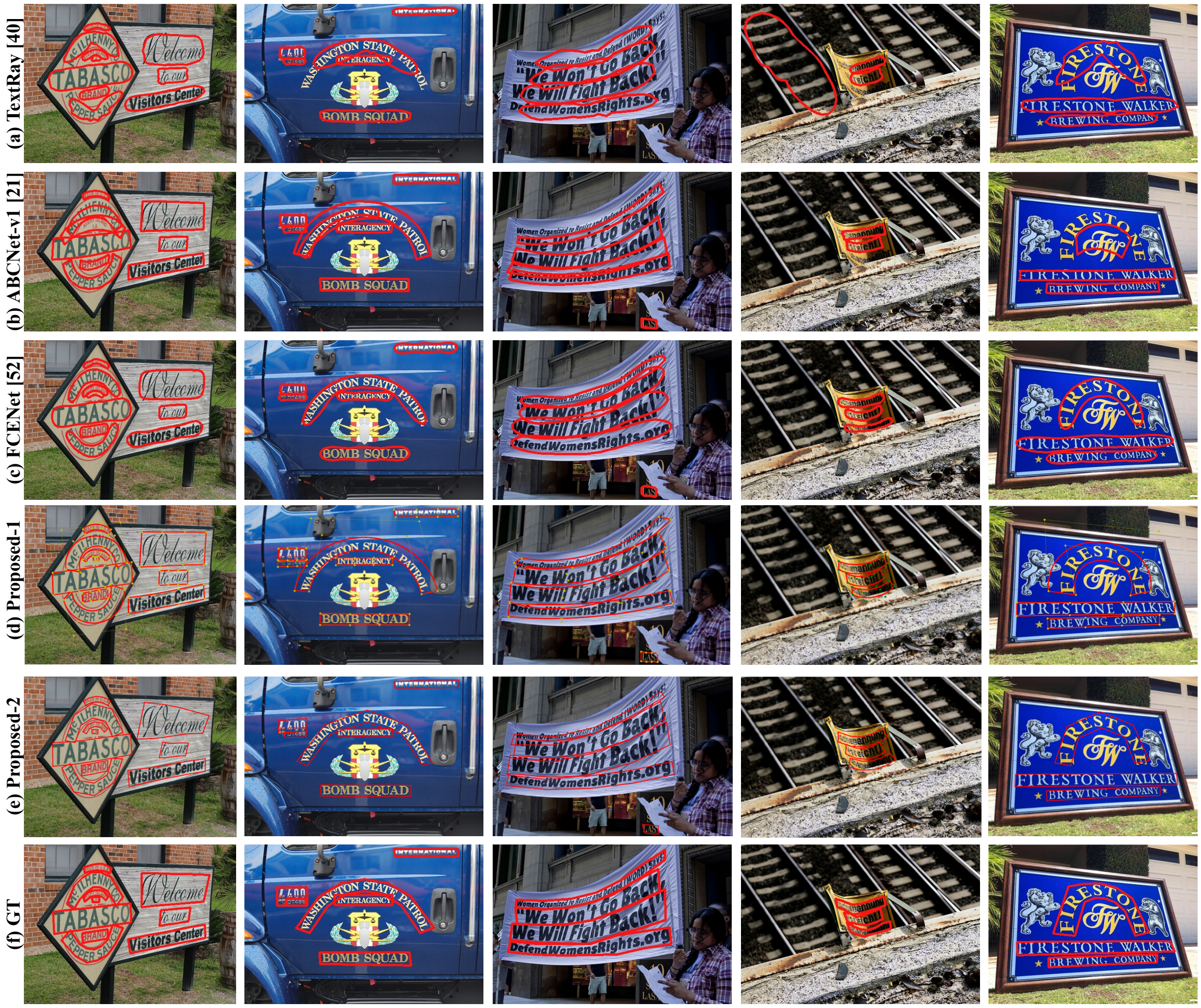


Figure 5.4: Qualitative comparison of our proposed models among SOTA methods on the CTW-1500 dataset [86]. The sample results of other methods are taken from [220].

MSRA-TD500 dataset and yield competitive results for ICDAR15 benchmarks.

In the next Chapter (Chapter 6), we focus on the recognition task, which aims to output a string/word instance from the cropped word images. The recognition task requires a different pipeline than the detection pipeline used in the previous two chapters. We use an encoder-decoder transformer architecture for addressing the irregular text instances. Our main contribution in the next Chapter is leveraging a 2D Learnable Sinusoidal frequencies Positional Encoding (2LSPE) with a modified feed-forward neural network to better encode the 2D spatial dependencies of characters in the irregular text instances.

Chapter 6

2LSPE: 2D Learnable Sinusoidal Positional Encoding using Transformer for Scene Text Recognition

Positional Encoding (PE) plays a vital role in a transformer's ability to capture the order of sequential information, allowing it to overcome the permutation equivarience property. Recent state-of-the-art transformer-based scene text recognition methods have leveraged the advantages of the 2D form of PE with fixed sinusoidal frequencies, also known as 2SPE, to better encode the 2D spatial dependencies of characters in a scene text image. These 2SPE-based transformer frameworks have outperformed Recurrent Neural Networks (RNNs) based methods, mostly on recognizing text of arbitrary shapes; However, they are not tailored to the type of data and classification task at hand. In this work, we extend a recent Learnable Sinusoidal frequencies PE (LSPE) from 1D to 2D, which we hereafter refer to as 2LSPE, and study how to adaptively choose the sinusoidal frequencies from the input training data. Moreover, we show how to apply the proposed transformer architecture for scene text recognition. We compare our method against 11 state-of-the-art methods and show that it outperforms them in over 50% of the standard tests and are no worse than the second best performer, whereas we outperform all other methods on irregular text datasets (i.e., non horizontal or vertical layouts). Experimental results demonstrate that the proposed method offers higher word recognition accuracy (WRA) than two recent transformer-based methods, and eleven state-of-the-art RNN-based techniques on four challenging irregular-text recognition datasets, all while maintaining the highest WRA values on the regular-text datasets.

6.1 Introduction

Recent state-of-the-art scene text recognition methods [27, 63–67, 93–95] are mainly based on the combination of a Convolutional Neural Network (CNN) as a feature extractor, and Recurrent Neural Networks (RNNs) for capturing sequential dependencies and producing sequences of characters. Although these RNN-based methods [27, 63–67, 93–95] perform well when the text in an image is horizontal or nearly horizontal, they often fail to correctly recognize irregular text[1] [101]. The main reason for these failures is that RNN-based methods require converted 1D features and are not designed for recognizing irregular-text instances, thereby cannot localize spatial information within 2D images. Several RNN-based methods have tried to mitigate the high curvature recognition problem using a spatial rectification module [11] that first rectifies the input image into a normalized image, and then treat the recognition problem as a sequence prediction task. However, rectification may cause errors in character recognition when the text exhibits severe curvature or orientation [101] values.

Transformer [1, 96] and its variations, such as Performer [233], are fairly recent deep learning architectures that mitigate the aforementioned issues for CNNs. Different from RNN based

[1] Irregular-text refers to text with arbitrary shapes that usually have severe orientation and/or curvature.

sequence-to-sequence models, a transformer adopts a global attention mechanism to encode and decode characters inside the text image using a look ahead strategy that is agnostic to the order of pixels. These capabilities enabled the application of transformers to a wide variety of problems with sequential data, such as machine translation [1], speech recognition [200], and computer vision [96, 204].

Central to the success of transformers, a Positional Encoding (PE) is an essential mechanism that enables the self-attention block to overcome its own permutation equivariance; that is without PE, the transformer's representation behaves similar to that of a bag-of-words model [185, 234]. There are several types of PE that have been used for the transformers (summarized in Table 6.1), and were mostly introduced for Natural Language Processing (NLP) applications: For example, the Sinusoidal PE (SPE) of fixed frequencies was introduced in [1] for language modeling; it is inductive and can handle input sequences of variable sizes. Relative PE (RPE) [235] and fully Learnable PE (LPE) [236] have also been used for machine translation, where the positional encoding values are learned from the data; However, these models are data-driven and therefore can not generalize to out of distribution samples.

In this work, we first extend the Learnable Sinusoidal Positional Encoding (LSPE) [234] from 1D to 2D, and apply the introduced 2LSPE version for scene text recognition. The proposed model has a learnable frequency capability, allowing it to adjust itself during training according to the input text instances of different lengths. As in [1], the development of the proposed scene text recognition model is based on the idea that a scene text image can be treated as a sequence of characters, which allows for the auto-regressive recognition of characters in an image.

Our contributions can be summarized as follows:

1. We are the first to apply the 2D Learnable Sinusoidal Positional Encoding (2LSPE), in which the frequencies are learned, to scene text recognition.

2. We show that the proposed model offers better recognition accuracy when compared to other state-of-the-art techniques, namely, [27, 63–67, 93–95] on five out of eight scene text recognition datasets [71, 79, 81, 83, 119], and not worse than second best results. Moreover, our method outperforms all other methods on irregular text datasets.

6.2 Related Works

Inspired by speech recognition solutions, previous state-of-the-art scene text recognition methods [27, 63–67, 93, 94, 145, 147, 149, 150] were mainly based on RNN frameworks [75] that leverage Long-Short-Term-Memory (LSTM) [6] for encoding and decoding a given image. In these methods, a CNN backbone is first used for feature extraction; Next a RNN encoder is leveraged to capture more contextual information and convert the extracted features into a sequence of features. Finally, a prediction module is used to predict the sequence of characters in the given input image.

In this regard, several prediction heads were suggested, for example some methods [63, 65, 66] use *Connectionist Temporal Classification* (CTC) to predict the output characters. Methods such as [63], employ a VGG model [171] as a backbone to extract features from the input images, followed by a Bidirectional Long-Short-Term-Memory (BLSTM) [6] for contextual information, and finally a CTC loss is applied to identify character sequences. Different solutions, such as those proposed in [27, 64, 145, 147, 149, 150] adopt an *attention mechanism* [172], where implicit attention is automatically learned and subsequently enhances the deep features in the decoding process. Although these methods [63, 66] perform well when the text instances are horizontally aligned, they fail to recognize curved or rotated text.

To improve the recognition accuracy on irregular input text images (*e.g.*, curved texts), some methods [27, 64, 65, 67, 94] proposed an extra rectification module *e.g.*, Spatial Transformer

Network (STN) [11], to handle geometrically distorted text instances. For example, Shi *et al.* [64] introduced a spatial attention mechanism to transform a distorted text region into a nearly horizontal text that is suitable for recognition. In later work, they [27] used a series of control points within a Thin-Plate Spline transformation to better rectify curved text, and to improve the recognition results on irregular text datasets. However, most of these methods (including [27, 64, 67, 94]) require one-dimensional (1D) features; they were not designed for recognizing irregular-text instances as they cannot keep track of the spatial information within two-dimensional (2D) images.

On another note, and similar to language modeling, the order of words in a sentence and the order of characters in a word are essential in scene text recognition. To that end, many recent transformer-based scene text recognition techniques [202, 203, 237] have used different types of PE and have outperformed the previous RNN-based state-of-the-art methods [27, 63–67, 93–95] on many benchmark datasets [71, 77, 79, 81–83, 119, 182]. Example of such methods include [237] that used a 1D fixed Sinusoidal PE for horizontal handwritten text recognition. Raisi *et al.* [203] proposed a 2D SPE (2SPE) to better capture the 2D spatial dependencies among characters in irregular text. On the other hand, to make the transformer's encoder more suitable for 2D inputs, Lee *et al.* [202] proposed a 2D SPE with adaptive amplitude, which achieved the state-of-the-art recognition accuracy in the majority of popular scene text recognition datasets; This is due to its learning capability in adjacent height and width directions. However, these methods [202, 203] made use of manually selected frequencies, as such they cannot handle variability in the text data [185].

Table 6.1: Summary of related abbreviations.

Abbreviation	Description
PE	Positional Encoding
SPE [1]	1D PE with fixed Sinusoidal frequencies
LPE [236]	1D PE with Learnable weights
LSPE [234]	1D Learnable frequencies SPE
2SPE [203]	2D Sinusoidal PE
2LSPE (*Proposed*)	2D Learnable frequencies SPE

6.3 Background

6.3.1 Multi-Head Self-Attention

The transformer architecture was initially introduced in [1] for machine translation of natural language using an attention-based mechanism. This architecture leverages self-attention layers, which scan through each element of a sequence, and accordingly compute an update by measuring the relationship between this element and the whole sequence [1]. The main advantages of attention-based models in transformers are their parallel computations suitability at a lower memory cost, making them more suitable than RNNs [6, 75] for learning from long sequences.

Each *self-attention* layer (the main defining part of a transformer), is made of an attention block that allows the model to learn and access information from the past hidden layers. Let $x_i \in \mathbb{R}^d$ denote the i-th input vector of size $d \times 1$; then a set of t input vectors can be represented in a matrix form as:

$$X = [x_1, x_2, ..., x_t]^\top \in \mathbb{R}^{t \times d}. \tag{6.1}$$

The functionality of the self-attention layer in a transformer can be described as a mapping

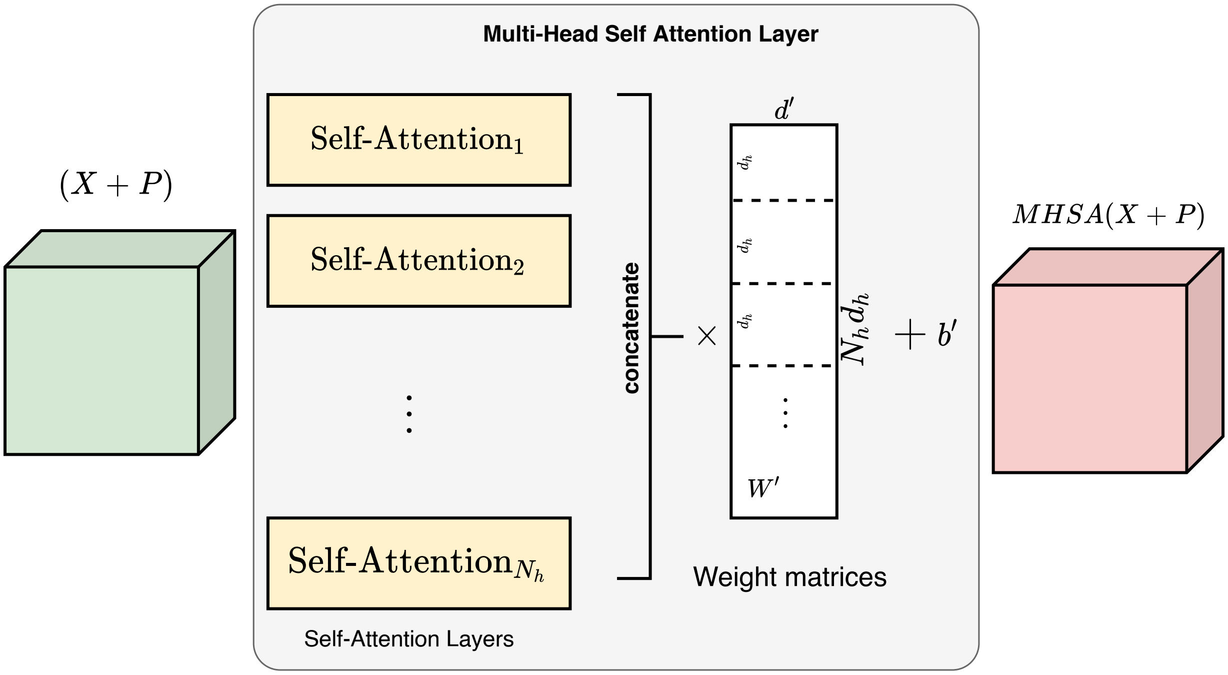


Figure 6.1: Multi-Head Self-Attention layer in a transformer. The above diagram is reproduced from [234].

between the matrix of input vectors (X) from d (input dimension) to d' (output dimension) using:

$$\text{Self-Attention}(X) = \texttt{Softmax}\,(A)\,XW^V, \tag{6.2}$$

where A is a $t \times t$ *attention scores* matrix that can be calculated as:

$$A = XW^Q W^{K^\top} X^\top, \tag{6.3}$$

and $W^Q \in \mathbb{R}^{d \times d''}$, $W^K \in \mathbb{R}^{d \times d''}$ and $W^V \in \mathbb{R}^{d \times d'}$ represent query, key and value matrices respectively.

As it can be seen in Figure 6.1, rather than only computing the attention once, the *multi-head mechanism* invokes the self-attention mechanism in equation (6.2) multiple times and in parallel, which enables the transformer to focus on different parts of the input In multi-head self-attention, the output of N_h heads (each of dimension d_h), are concatenated and projected to a dimension d' using:

$$\mathrm{MHSA}(X) = \underset{h \in 1,2,\ldots,N_h e}{\mathrm{concat}} \left[\text{Self-Attention}_h(X)\right] W' + b', \tag{6.4}$$

where $W' \in \mathbb{R}^{N_h d_h \times d'}$ and $b' \in \mathbb{R}^{d'}$ are the projection matrix and bias terms respectively.

6.3.2 Positional Encoding

The aforementioned self-attention layer in the transformer is permutation equivarient [96], that is the order of the input set is irrelevant, and the output result is the same irrespective of the input order. To tackle this limitation [1], a PE is introduced to modify the input set before feeding its output to the self-attention module equation (6.2) using:

$$A = (X + P)W^Q W^{K^\top}(X + P)^\top, \tag{6.5}$$

where $P \in \mathbb{R}^{t \times d}$ denotes the embedding matrix for each position. Several attempts to define P within the self-attention components were proposed, and can be categorized as *relative* PE [235] and *absolute* PE [1, 185]. In this section, we will focus on the absolute PE category that includes *Sinusoidal PE (SPE)* with fixed frequencies and the *learned version of PE*. *Sinusoidal PE (SPE)* with fixed frequencies was first introduced in [1], where the position information $P \in \mathbb{R}^{t \times d}$ is defined as follows:

$$\begin{aligned} P(x, 2i) &= \sin\left(x \cdot c^{2i/d}\right), \\ P(x, 2i+1) &= \cos\left(x \cdot c^{2i/d}\right), \end{aligned} \tag{6.6}$$

where $c^{2i/d}$ is the i-th fixed frequency, $c = 10^{-4}$, and x denotes the order in the input set ($x = 0, 1, \dots, t - 1$), and i refers to the position along the encoding vector dimension ($i = 0, 1, \dots, \lfloor (d-1)/2 \rfloor$) and $\lfloor \cdot \rfloor$ is the floor operator.

It has been shown in [1, 185] that injecting the position information using equation (6.5) in each block of the transformer yields better performance. However, for fairness of comparison against existing methods ([202, 203]), we only consider adding the PE to the first block of self-attention in the transformer.

The *learned version of PE* was introduced in [236], where the entire positional information is learnable ($P \in \mathbb{R}^{t \times d}$). However, the fully learnable PE was missing the inductive property, as it required a fixed maximum length of its input set before training [185]; this can unfortunately cause generalization issues, especially for a variable length test set.

6.4 Overview on The Proposed Method

Figure 6.2 illustrates the proposed architecture for scene text recognition which is built upon the standard transformer's architecture [1]. We can categorize it into two main modules: encoder and decoder. The encoder's main role is to extract high-level 2D feature representations of an input image, whereas the decoder is used to convert these feature maps to a sequence of characters.

6.4.1 Encoder

The proposed encoder module makes use of the Multi-Head Self-Attention (MHSA) mechanism presented in Section 6.3, along with three main sub-blocks:

(1) *CNN Feature Extraction*: A CNN first processes the input image to extract a compact feature representation and to learn a 2D representation of an input image. To that end, we adopt a

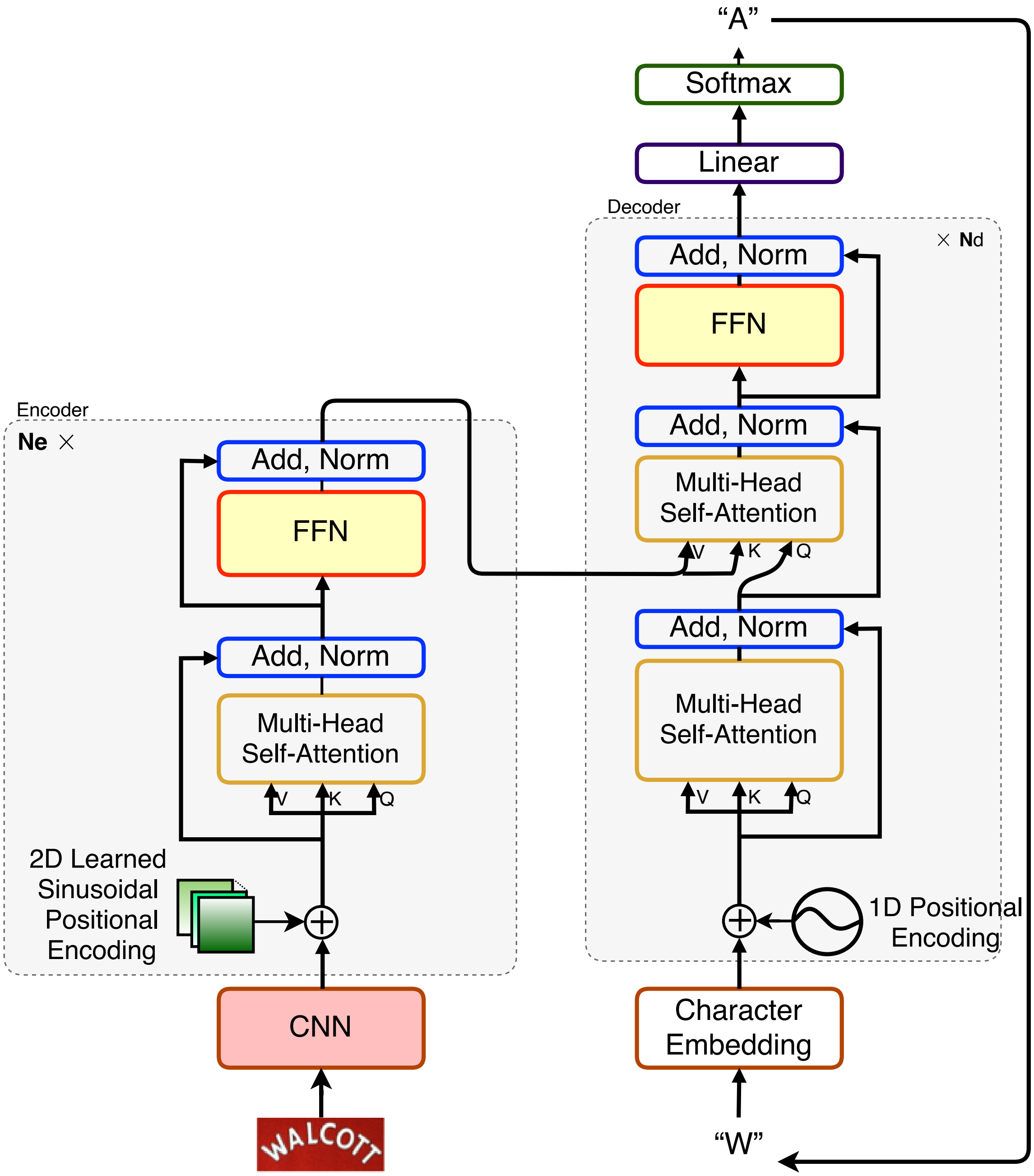


Figure 6.2: The proposed text recognition using transformer architecture, where N_e and N_d denote the number of layers in the encoder and decoder. Unlike [1], our proposed architecture utilizes 2D Learnable Sinusoidal Positional Encoding, a ResNet-31 backbone and two layers of FFN.

modified ResNet-31 architecture [164] as the CNN backbone. At runtime, all the input images are converted to grayscale and resized to 32×100 pixels.

(2) *2D Learnable Sinusoidal Positional Encoding (2LSPE)*: In this module, the proposed 2D PE signals are learned, generated and then incorporated within the MHSA process. Further information about the proposed 2LSPE scheme are detailed in Section (6.5).

(3) *Feed-Forward Network (FFN)*: Here, we use a modified version of the FFN layer in the original transformer [1] so that it becomes capable of capturing the features generated by the encoder's MHSA mechanism. The modified FFN consists of 2 layers of 1×1 convolutions with ReLU activations followed by a residual connection.

6.4.2 Decoder

Similar to [1], the decoder module includes 1D SPE, MHSA and FFN layers as shown in Figure 6.2. The decoder's main role is to use an autoregressive model by attending to the visual features generated by the encoder to predict the next sequence of characters.

6.5 2D Learnable Sinusoidal Positional Encoding

Let $I_i \in \mathbb{R}^{H_0 \times W_0}$ denote the i-th training image, where H_0 and W_0 are the height and width of each training image, $i = 1, 2, \ldots, N_T$, and N_T is the number of training images. Each training image I_i is first processed through a CNN to produce image features of lower resolution, let it be denoted as $\mathcal{X}_i \in \mathbb{R}^{H \times W \times d}$, where d indicates the number of channels, H and W are the height and width of $\mathcal{X}_i$ such that $H = H_0/\eta$, $W = W_0/\eta$ and η is the downsampling factor. In this case, instead of the 1D input set as described in Section 6.3.1, we have a 2D feature map that comes from the CNN. Accordingly, the input is a tensor $\mathcal{X}$ of dimension $H \times W \times d$, and for every position in the input $\mathcal{X}$, an attention score is used to associate a query and a key at that given position, thereby constructing the attention score tensor $\mathcal{A}$ of dimension $H \times W \times d'$. To keep the formulas consistent with the 1D case, we slice the tensors as follows: if $k \in [1, \ldots, d]$, we write $\mathcal{X}_{:,:,k}$ and $\mathcal{A}_{:,:,k}$ to refer to the k-th 2D slice of the tensors $\mathcal{X}$ and $\mathcal{A}$, respectively. With this notation in place, each self-attention layer output at slice k can be expressed as:

$$\text{Self-Attention}(\mathcal{X})_{:,:,k} = \texttt{Softmax}\left(\mathcal{A}_{:,:,k}\right) \mathcal{X}_{:,:,k}\, \mathcal{W}^{V}_{:,:,k}. \tag{6.7}$$

The above formula also can be extended to the multi-head self-attention mechanism in equation (6.4). For each position $(h, w) \in [1, \ldots, H] \times [1, \ldots, W]$, we obtain the proposed 2D Learnable Sinusoidal PE, $\mathcal{P} \in \mathbb{R}^{H \times W \times d}$, as:

$$\begin{aligned} \mathcal{P}(h, w, 2i) &= \sin\left(h \cdot f_i\right), \\ \mathcal{P}(h, w, 2i+1) &= \cos\left(h \cdot f_i\right), \\ \mathcal{P}(h, w, 2j + d/2) &= \sin\left(w \cdot f_j\right), \\ \mathcal{P}(h, w, 2j + 1 + d/2) &= \cos\left(w \cdot f_j\right). \end{aligned} \tag{6.8}$$

where $f_i, f_j \in \mathbb{R}$ are the learnable frequencies for the 2D PE signal, h and w specify the horizontal and vertical positions, and $i, j \in [0, d/4]$ and d denote the number of channels in the input $\mathcal{X}$. Similar to equation (6.5), the attention scores can be obtained as follows:

$$\mathcal{A}_{:,:,k} = (\mathcal{X}_{:,:,k} + \mathcal{P}_{:,:,k})\mathcal{W}^{Q}_{:,:,k}\mathcal{W}^{K^{\top}}_{:,:,k}(\mathcal{X}_{:,:,k} + \mathcal{P}_{:,:,k})^{\top} \tag{6.9}$$

where $\mathcal{W}^{Q}$ and $\mathcal{W}^{K}$ correspond to the weights of query and key tensors, respectively. While the SPE is commonly used in different transformer architectures for scene text recognition [202, 203, 237], our model foregoes the fixed frequencies and instead can adaptively learn those frequencies ($f_i, f_j \in \mathbb{R}$) using in a data-driven approach. We show in the next section that learning those

Table 6.2: Comparing the WRA of some of the recent text recognition techniques using IIIT5k [79], SVT [77], ICDAR03 [182], ICDAR13 [82], ICDAR15 [71], SVT-P [81], CUT80 [83] and COCO-Text [119] datasets. The models provided in [67] are utilized for evaluating the methods in [27, 63–67] on all the datasets, while the model in [202] is tested on COCO-Text dataset. The rest of the results are as reported by the authors [93–95, 202]. Best and second best methods are highlighted in bold and underline, respectively, and "–" refers to unavailable results. Our method outperforms all the other methods on irregular-text.

	Regular-Text Datasets				Irregular-Text Datasets			
Method	IIIT5k	SVT	ICDAR03	ICDAR13	ICDAR15	SVT-P	CUT80	COCO-Text
CRNN [63]	82.73%	82.38%	93.08%	89.26%	65.87%	70.85%	62.72%	48.92%
RARE [64]	83.83%	82.84%	92.38%	88.28%	68.63%	71.16%	66.89%	54.01%
ROSETTA [66]	83.96%	83.62%	92.04%	89.16%	67.64%	74.26%	67.25%	49.61%
STAR-Net [65]	86.20%	86.09%	94.35%	90.64%	72.48%	76.59%	71.78%	55.39%
CLOVA [67]	87.40%	87.01%	94.69%	92.02%	75.23%	80.00%	74.21%	57.32%
ASTER [27]	93.20%	89.20%	92.20%	90.90%	74.40%	80.90%	81.90%	60.70%
MORAN [93]	91.20%	88.30%	95.00%	92.40%	68.80%	76.10%	77.40%	–
ESIR [94]	93.30%	90.20%	–	91.30%	76.90%	79.60%	83.30%	–
SCRN [95]	94.40%	88.90%	95.00%	93.90%	78.70%	80.80%	87.50%	–
SATRN [202]	92.80%	**91.30%**	**96.70%**	**94.10%**	79.00%	86.50%	87.80%	65.11%
2SPE [203]	89.23%	89.34%	95.85%	93.89%	75.78%	84.34%	84.03%	65.80%
2LSPE (Proposed)	**94.75%**	90.44%	96.42%	94.09%	**80.49%**	**86.76%**	**88.19%**	**73.38%**

frequencies help the self-attention module to focus more on spatial dependencies of irregular text.

6.6 Experimental Results

In this section, we present an experimental evaluation for the proposed method against a select state-of-the-art scene text recognition techniques [27, 63–67], using recent public datasets [71, 79, 81–83, 119, 182] that cover a wide variety of challenges. All the methods are evaluated using the Word Recognition Accuracy (WRA) metric [67, 203] that is computed as:

$$\text{WRA (\%)} = \frac{\text{No. of Correctly Recognized Words}}{\text{Total Number of Words}} \times 100 \tag{6.10}$$

6.6.1 Implementation Details

As in [67], we train our model with a combination of images from *SynthText (ST)* [87] and *Mjsynth (MJ)* [88] datasets, where all the input images are resized to 32×100 pixels. We train our proposed method with $N_e = 9$ and $N_d = 6$ by a batch size of 192 on four NVIDIA V100 16GB GPUs. We use Adam as an optimizer [238] with the initial learning rate of 3×10^{-4} at 3×10^5 iterations. A ResNet-31 [164] is used as a backbone feature extractor and a union of the training sets ICDAR13 [82], ICDAR15 [71], IIIT5k [79], SVT [77], and CUT80 [83] are used for validation purposes. The final model is chosen based on the best recognition accuracy from the mentioned datasets. For training and validating our models, 36 classes of alphanumeric characters, 10 digits $(0-9)$ + 26 capital English characters (A-Z) = 36 are used.

6.6.2 Datasets

Two types of datasets are used for evaluating the recognition results, (1) *regular-text* recognition datasets: *IIIT5k* [79], *SVT* [77], *ICDAR03* [182] and *ICDAR13* [82] that mainly contain horizontal text, and (2) *irregular-text* recognition datasets: *ICDAR15* [71], *SVT-P* [81], *CUT80* [83] and *COCO-Text* [119], which contain multi-oriented and curved text. These datasets are more challenging than their regular-text counterparts.

6.6.3 Quantitative Results

Table 6.2 shows a comparison of the WRA for the proposed method versus other methods in the literature such as [27, 63–67, 202]. We first compare our proposed model with RNN-based methods [27, 63–67, 202] and show that the proposed 2LSPE outperforms them with a large margin, it even provides higher WRA values compared to the methods in [93–95] that deploy powerful spatial rectification modules in their architectures to better recognize irregular text. We also compare our method with the recent transformer based methods, namely SATRN [202] and 2SPE [203] that use fixed frequencies of Sinusoidal PE. Our 2LSPE outperforms STARN and 2SPE on irregular-text datasets offering high improvement on the challenging COCO-Text dataset, while maintaining high WRA values on the regular-text ones. We believe that the performance gap on regular text datasets with [202] can be attributed to the use of a deeper CNN backbone (ResNet-101) in SATRN, while our proposed model uses ResNet-31 to evaluate the effect of the learnable frequencies based PE. This opens an interesting future work to study the effect of a deeper network, as used in [202], on the recognition accuracy of the proposed 2LSPE.

6.6.4 Qualitative Results

In Figure 6.3, we provide sample qualitative results for the proposed method when tested on challenging cases from various datasets (Section 8.4.2). It can be seen that the proposed method with 2LSPE is able to recognize regular or horizontal text, and irregular text (curved and vertical text). As it can also be noticed from this Figure, the proposed technique is robust to different challenging conditions, such as different colors, fonts, orientations, blurriness and backgrounds. We also show some failure cases of our proposed model in Figure 6.4. These challenging examples indicate that their is still a room to improve the performance of the proposed scheme by tackling the challenges of partial-occlusions, illumination variations and geometric distortions.

6.6.5 Effect of Positional Encoding

We have also conducted experiments using the transformer architecture *without* PE, as well as *with* 2D fixed or 2D learnable SPE on the same eight benchmark datasets presented in Section 8.4.2, where Table 6.3 shows the average WRA of these results. First for the transformer *without*

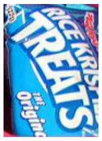
Method						
CRNN [1]	flickrl	cafl	os	b	t	o
CLOVA [6]	flickr	cafc	survo	cle	a	con
ASTER [5]	flickr	cafc	1993	i	the	xref
SATRN [34]	flickri	cafc	the	barting	realization	single
Proposed	**flickr**	**cafe**	**10000**	**builder**	**railways**	**treats**

Figure 6.3: Qualitative comparison results among CRNN [63], ASTER [27], CLOVA [67], and SATRN [202] methods, and our proposed method for some challenging examples. These images are from [71, 79, 81, 83, 119] datasets. Green and red denote characters that are correctly and incorrectly recognized, respectively.

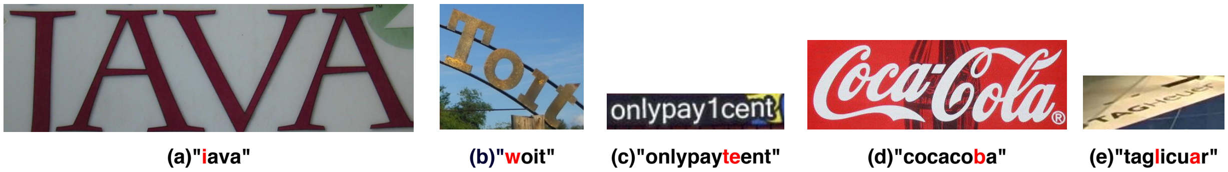

(a)"iava" (b)"woit" (c)"onlypayteent" (d)"cocacoba" (e)"taglicuar"

Figure 6.4: Qualitative results on some sample images that the proposed method failed to recognize all the characters correctly, which contain the following challenges (a) partial-occlusion, (b) illumination variation and complex background, (c) alphanumeric, (d) complex font, and (e) geometric distortion and low resolution. The above images are from the publicly available datasets [81–83, 239].

Table 6.3: Comparing the effect of different PE schemes on the proposed transformer's architecture. Note: transformer's architecture without using PE causes a significant drop in the average WRA.

Method	Encoder	Decoder	Average WRA
Without PE	MHSA	MHSA	80.19%
2SPE [203]	2SPE + MHSA	SPE + MHSA	84.78%
Proposed	2LSPE + MHSA	SPE + MHSA	**88.06%**

PE case, we completely remove the positional encoding in the encoder and decoder, without changing the MHSE and FFN modules; the model achieves an average WRA of 80.19% which is the lowest performance compared to the other PE based techniques. This low performance is more observable on irregular-text datasets; Therefore, we conclude that PE is a necessary part to use in a transformer architecture for scene text recognition.

Next, we apply a 2D positional encoding with fixed sinusoidal frequencies (2SPE), as in [203], and compare it with our proposed 2LSPE with the frequencies that are learned during training; As seen in Table 6.3, the proposed transformer with 2LSPE achieves the highest average WRA compared to the transformer with 2SPE [203] on benchmark datasets; This improvement in recognition performance for 2LSPE over 2SPE can be attributed to the flexible capability of the learnable frequencies in complementing the MHSA in a transformer's architecture through a data-driven way.

It is worth mentioning that the average recognition inference time in milliseconds per word for 2LSPE and 2SPE models are 87.7% and 88.8%, respectively, which shows that using 2LSPE provides slightly better inference time. However, RNN-based methods [27, 63–67] are able to provide significantly lower inference time compared to the transformer-based architectures [101, 204].

6.7 Conclusion

Throughout this work, we have extended the capabilities of a recently proposed positional encoder with learnable sinusoidal frequencies from one-dimensional to a two-dimensional format. Moreover, we show how to incorporate the learned positions within the multi-head self-attention (MHSA) of the transformer's architecture for scene text recognition.

To evaluate the proposed system, we first report on its WRA results and compare them with two recent transformer-based methods and eleven state-of-the-art RNN-based techniques. Experimental results further show that the proposed model has achieved the state-of-the-art WRA performance on five out of eight benchmark datasets. Furthermore, the effect of different PE schemes on the transformer's architecture has been studied. The proposed 2D sinusoidal PE technique with learnable frequencies has outperformed the baseline method that uses fixed PE frequencies in terms of recognition accuracy in all cases.

In the previous Chapters, we separately addressed text detection (Chapter 4 & Chapter 5) and recognition (Chapter 6) for irregular text instances. However, reading text in the wild require both detection and recognition modules. In the following Chapter (Chapter 7), unlike SOTA methods that combine two different pipelines of detection and recognition modules for a complete text reading, we propose a different model that reads characters from the wild images. We utilize a similar detector as in Chapter 4 with a multi-scale feature extraction backbone to capture any character shape in the given input image. We also discuss our interest in character detection and recognition rather than word spotting in the wild images.

Chapter 7

End-to-End Text Detection and Recognition Using Transformers

This chapter utilizes a transformer-based object detection framework, namely Detection using Transformers (DETR), to detect and recognize at the same time the characters in unconstrained environments (i.e., in the wild), which offers simpler and robust end-to-end architecture than the previous methods. The proposed framework leverages an adaptive feature extraction to better focus on the position of character regions and a bounding box loss function that is more precise in detection and recognition of characters with different scales and aspect ratios. We conduct experiments on the ICDAR13 benchmark dataset designed explicitly for character-level text detection to evaluate our proposed architecture's effect. Experimental results show that the proposed method outperforms the state-of-the-art detectors.

7.1 Introduction

Reading of text in a scene requires two stages: locate the text and then recognize the character in the detected regions, which are called scene text detection and scene text recognition. Some methods combine these two stages, which leads to an end-to-end detection and recognition (scene text spotting) [59, 60, 100, 101, 240]. Inspired by deep-learning frameworks like Convolutional Neural Network (CNN) [164, 171] and Recurrent Neural Network (RNN) [6], many end-to-end scene text detection and recognition methods [59, 62, 100, 149, 241] proposed. Some of these methods achieved superior performance end-to-end text detection and recognition at word-level in different benchmark datasets [71, 72, 86]. However, these CNN and RNN based methods require several handcrafted components such as anchor generation, non-maximum suppression (NMS) in *regression-base* methods, or multiple processing stages (*e.g.* label generation) in *segmentation-based* method to detect following by a rectification module before to output the sequences of characters using RNN. Furthermore, Some of these models, as described in [101, 102] show poor performance when characters in the text are vertical or partially occluded.

As mentioned above, previous end-to-end scene text detection and recognition approaches aim to output word instances whose primary components are characters. Therefore, we aim to design a simple and end-to-end framework that directly and precisely extracts the characters from the given image and then combines the extracted characters to form the final word. To achieve this goal, we utilize state-of-the-art transformer-based techniques that alleviate the issues of previous CNN-based methods. detection and recognition the text at character level by using an end-to-end transformer architecture eliminates the complexity of detection and recognition on different architectures. It also removes the need for rectification module [11, 27, 101] for detection and recognition arbitrary-shape text instances as used in many end-to-end words detection and recognition methods [59, 60, 100, 240].

Transformer [1] is an attention-based pipeline that, after achieving superior performance in sequence modelling and machine translation tasks [242], recently emerged in many computer vision fields and achieved state-of-the-art results in many benchmarks [204, 243]. Current state-of-the-art object detectors [96, 201, 244–246] mainly inspired by self-attention mechanism in transformers outperformed prior Convolution Neural Networks (CNN) models [247]. For example, Detection using Transformer (DETR) [96], was the first transformer-based detector that introduced a new concept for object detection framework. DETR uses a new technique called object queries and task object detection as a set prediction problem [247]. In contrast to other detectors, it removed the need to design hand-designed components like anchor design and non-maximum suppression (NMS) post-processing and directly detects objects in the given image using so-called object queries. However, DETR has low accuracy on small objects and slow convergence during training [201, 247].

Many recent works proposed efforts to alleviate the issues mentioned for [96], for example, Deformable-DETR [201] aims to design data-dependent sparse attention to address the small object detection problem of [96] and achieved higher precision performance and fewer training epochs. Pyramid Vision Transformer (PVT) [244] is a hierarchical pure transformer backbone that achieved superior performance in classification, object detection, and segmentation tasks. PVT utilizes a non-overlapping patch partition followed by a linear patch embedding to reduce the sequence length in the given input and preserve the fixed channel dimensions. This backbone can be accompanied by a transformer framework like [96] to predict dense objects efficiently.

Sparse R-CNN [245] proposed a sparse algorithm for object detection without relying upon dense candidate regions. In order to detect objects, it first generates a random sparse set of boxes and then iteratively performs classifications and detection of the candidate boxes. In a recent work, Deformable Patch-based Transformer (DPT) [246] presented DePatch that adaptively split images in a data-driven way which address the problem of PVT [244] that uses the predefined fixed-patched. DePatch forces the network to concentrate on desired object regions and extract

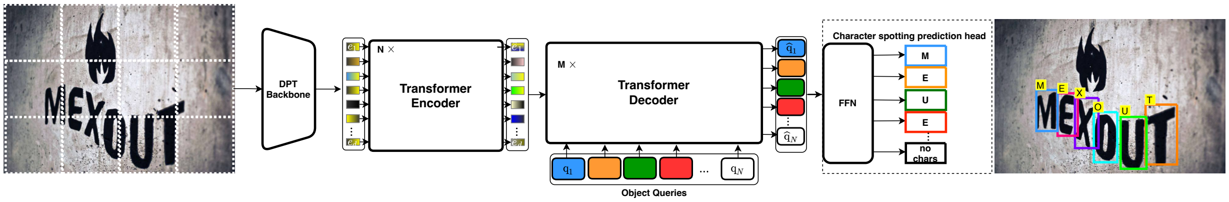


Figure 7.1: The proposed end-to-end character-level text detection and recognition framework [2].

more semantic formations in patches with different positions and scales. DPT achieved state-of-the-art performance on image classification and object detection.

In this chapter, we only focus on character detection and recognition by leveraging the DETR [96] as our baseline detector. The contribution of these works are: (1) We propose a new transformer based model based on [96] by modifying its feature extraction backbone and prediction head by leveraging a robust bounding box loss function. (2) We compare state-of-the-art transformer-based methods on detection and recognition the characters of the wild images with our proposed architecture. (3) We provide quantitative and qualitative results to show the performance of our proposed model.

7.2 Methodology

Figure 7.1 shows the proposed architecture. The framework of our proposed method mostly follows the encoder-decoder detector form [96]. The network first adaptively extracts image features using a DPT-Small [246] backbone from different small patches; The resulting feature set is passed to a transformer encoder. For decoding, a fixed set of learned embeddings called object queries are passed through a transformer decoder. The feature vectors tests obtained are fed to shared fully connected layers that directly predict each query's class and bounding box set. The Bipartite matching loss is used for training the network, which leverages the Hungarian matching algorithm [194] for comparing and establishing a one-to-one mapping between N queries and N ground-truths [96]. The prediction head outputs rectangular bounding boxes $b = [x, y, w, h]^\top$ can encase the character region by simplifying defining (x, y) as the bounding box's center point coordinates, and w, h representing the box's width and height respectively. To train the network, we also modify the prediction head, along with the loss and matching functions as described in below.

Loss Function: The bounding box loss function of [96] uses a linear combination of ℓ_1 and GIoU loss. Let $\hat{b}_i$ and b_j denote the i^{th} predicted and j^{th} ground truth bounding boxes, respectively, then we define our loss function as:

$$\mathcal{L}_{\text{box}}(\hat{b}_i, b_j) = \lambda_1 \mathcal{L}_{reg}(\hat{b}_i, b_j) + \lambda_2 \mathcal{L}_{\alpha-\text{GIoU}}(\hat{b}_i, b_j) \tag{7.1}$$

where λ_1 and $\lambda_2 \in \mathbb{R}$ are hyper-parameters, and $\mathcal{L}_{reg}(\cdot)$ and $\mathcal{L}_{\alpha-\text{GIoU}}(\cdot)$ are the rectangular bounding box loss functions based on regression and $\alpha-$GIoU. The intuition behind using $\alpha-$IoU loss is that it improves the average precision performance of small and large characters, converging faster on small datasets during training. The $\alpha-$GIoU is defined as [248]:

$$\mathcal{L}_{\alpha\text{-GIoU}} = 1 - IoU^{\alpha} + (\frac{|C \setminus (\hat{b}_i \cup b_j)|}{|C|})^{\alpha}, \tag{7.2}$$

where $\mathcal{L}_{\alpha\text{-IoU}} = 1 - IoU^{\alpha}$, C denotes the smallest convex shape enclosing b_i and $\hat{b_j}$. In our experiments, the $\alpha = 3$ showed better performance (See §7.3 for more detail).

For regression, we use the Smooth-$\ln$ based Regression Loss as in [2], which is more robust to the variation of scales and ratios in different character instances than the ℓ_1 used in [96]. The regression loss is then defined as:

$$\mathcal{L}_{reg}(\hat{b}_i, b_j) = (|\hat{b}_i - b_j| + 1)\ln(|\hat{b}_i - b_j| + 1) - |\hat{b}_i - b_j| \tag{7.3}$$

where $|\cdot|$ demonstrates the absolute operator

7.3 Experimental Results

In this section, we first compare our proposed method with state-of-the-art transformer-based detectors and then present some qualitative results to show the model's performance. Finally, we provide an ablation study to investigate the effect of the different components in the proposed pipeline.

Implementation Details: We adopt the DETR's [201] architecture as our main framework with a DPT-small [164] backbone for feature extraction. The number of object queries are set to 300 and AdamW [211] optimizer is used to optimize the parameters of the model. We use horizontal flip and resize the images similar to [96] for augmentation. We first pre-train our proposed model and methods in comparison on $100k$ images of Synth-text [87] with character level annotations for 8 epochs and then fine-tuned on the ICDAR13 dataset to ensure the training converges. We train our model with a batch size of 2 per GPU using 4 Tesla V100 GPUs and a learning rate (LR) of 1×10^{-4}. The pre-training of our proposed model on Synth-text datasets takes ~ 20 hours, and fine-tuning takes ~ 3 hours.

Datasets: The ICDAR13 dataset [82] is a benchmark dataset that includes both word-level and character-level annotations using rectangular boxes containing 229 and 233 images for training and testing. Most of the text instances of this dataset are horizontal and high-resolution. Since ICDAR13 is a well-known benchmark dataset that contains character-level annotations, we conduct our experiment on this dataset. However, we provide some qualitative sample results on other arbitrary-shape text datasets including Total-Text [72] and CTW-1500 [86] to better show the performance of our proposed model.

Table 7.1: Comparing the character detection and recognition performance of our proposed methods with state-of-the-art detectors [96, 244–246] on ICDAR13 [82] dataset.The best performance is highlighted in **bold**.

Model-Name	AP	AP_{50}	AP_{75}	AP_s	AP_m	AP_l	epochs
DETR [96]	0.49	0.78	0.57	0.48	0.58	0.41	700
PVT [244]	0.57	0.83	0.68	0.55	0.67	0.53	200
Sparse R-CNN [245]	0.59	0.80	0.70	0.49	0.69	0.65	200
DPT [246]	0.62	0.86	0.76	0.61	0.68	0.58	200
Proposed	**0.66**	**0.89**	**0.78**	**0.64**	**0.72**	**0.63**	**200**

Evaluation Metric: To our best knowledge, this is the first work that focuses on character detection and recognition; there is no evaluation metric to measure the performances of the predicted characters in the scene text detection community. Nevertheless, we can task characters as different classes of objects; Thus, we can use mean average precision (AP) as our evaluation metric adopted as a standard in many recent object detection algorithms to detect and recognize 36 alphanumerical (10 digits + 26 capital) characters directly in the images.

Quantitative Results: To evaluate the performance of the proposed method, We compare it with DETR [96], PVT [244], Sparce R-CNN [245], and DPT [246]. The quantitative comparison is shown in Table 7.1. Our proposed method outperformed the state-of-the-art detectors by a large margin,$\sim$ 4% compare to the best detector in AP performance. It also performed better in the detection and recognition of small, medium, and large characters. The baseline DETR [96] not only performed poorly on small and large characters, but it also required more training epochs to converge on the ICDAR13 dataset. On the other hand, with a lower number of training iterations, PVT significantly outperforms DETR by $\sim$ 8%. While Sparce-RCNN outperformed the PVT in overall AP by $\sim$ 2% in reading better of medium and large characters, it showed poor performance in detection and recognition of small characters. In contrast, DPT performed better in small character detection and recognition and achieved the second-best performance in terms of AP.

Table 7.2: Ablation study of our model using different components. The models trained only on the train set of ICDAR13 and no synthetic images used for pre-training. The best performance of our model is shown in **bold** . We also show the best performances in the ablation studies of Backbone and Bounding-box-loss (different αs) experiments with red and blue, respectively.

Model	Backbone	Bounding-box loss	AP
Baseline	ResNet50	GIoU+ℓ_1	0.410
Baseline-2	PVT-Small	GIoU+ℓ_1	0.460
Baseline-3	DPT-Small	GIoU+ℓ_1	0.480
Baseline-3 $(\alpha = 0.5)$	DPT-Small	$\alpha-$GIoU+ℓ_1	0.474
Baseline-3 $(\alpha = 2)$	DPT-Small	$\alpha-$GIoU+ℓ_1	0.486
Baseline-3 $(\alpha = 3)$	DPT-Small	$\alpha-$GIoU+ℓ_1	0.511
Baseline-3 $(\alpha = 4)$	DPT-Small	$\alpha-$GIoU+ℓ_1	0.488
Baseline-3 $(\alpha = 5)$	DPT-Small	$\alpha-$GIoU+ℓ_1	0.462
Proposed $(\alpha = 3)$	DPT-Small	$\alpha-$GIoU+Smooth-ln	**0.520**

Qualitative Results: Figure 7.3 shows the qualitative results on some challenging sample images. As seen, the proposed model is robust in detection and recognition small, medium, large and even complex fonts characters compared to the baseline model. It also performed well on detection and recognition of partially occluded and oriented characters as shown in Figure 7.3(b) and Figure 7.3(c), respectively.

To see the generalization ability of the proposed method, we also provided some qualitative result of arbitrary-shape text of Total-text [72] and CTW-1500 [86] datasets, where model was agnostic to the text instance of them. As shown in Figure 7.2 the model was able to detect and recognize precisely the characters in various text instances of the given images.

Ablation Study: To assess the added value of the various components in our model, we performed an extensive ablation study on ICDAR13 datasets. Table 7.2 summarizes the obtained results.

We started the experiments by baseline model that uses a ResNet-50 backbone for feature extraction, GIoU+ℓ_1 loss for bounding box regression; the model achieved an AP performance

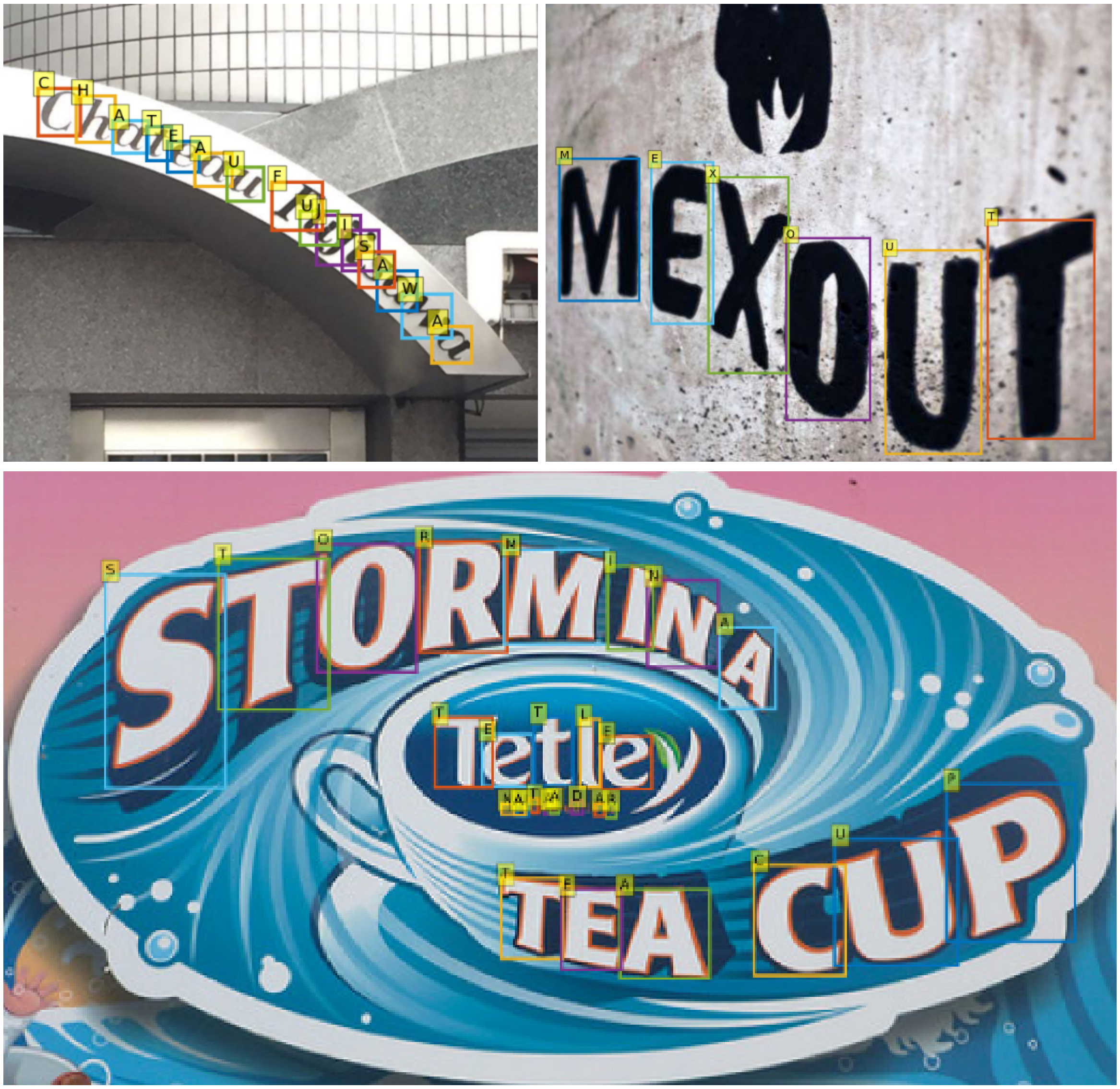
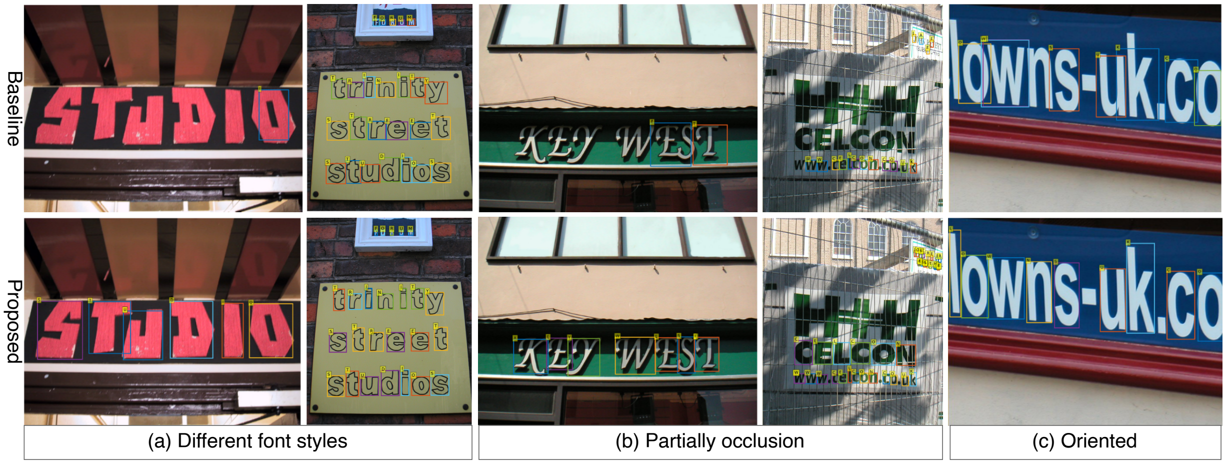


Figure 7.2: Qualitative results of the proposed method in out of distribution samples from Total-text [72] and CTW-1500 [86] datasets. The proposed method detects characters in arbitrary-shape text instances. The above images are from the mentioned public benchmark datasets [72, 86].



(a) Different font styles

(b) Partially occlusion

(c) Oriented

Figure 7.3: Qualitative comparison of the baseline [96] and proposed methods on some of the challenging images of the ICDAR13 public benchmark dataset [82]. Best viewed when zoomed.

of 0.41. We then replaced the backbone with PVT-small yielding an AP=0.46, which outperformed the baseline; We found that using DPT-small as backbone led to further performance boost compared to PVT-small backbone.

To evaluate the sensitivity of our proposed loss function in the detector's performance, we then continued our experiments with different values of α in equation (7.2) in combination of the baseline ℓ_1 loss on the ICDAR13 dataset. As shown in Table 7.2, the baseline-3 model with $(\alpha = 3)$ achieved the best performance (AP=0.511). It is worth mentioning that, when $\alpha < 2$ (e.g., $\alpha = 0.5$) or $\alpha > 4$ the AP performance decreased compared to the baseline-3 model that used $(\alpha = 1)-$GIoU. The AP improvement in Table 7.1 (see AP_s and AP_l columns respectively) confirm the the effectiveness of using $\alpha-$GIoU in detecting of small and large characters.

We finally replaced the baseline regression bounding-box loss (ℓ_1) with Smooth-$\ln$ as shown in equation (7.3). The Smooth-$\ln$ in combination with $(\alpha = 3-)$GIoU loss achieved the best performance on the ICDAR dataset by outperforming the baseline DETR model [96] by $\sim 11\%$, which uses GIoU and ℓ_1 losses and fewer iteration during training; the proposed model also improved the SOTA detection model (Baseline-3) by a large margin of $\sim 4\%$ (See Table 7.1 and

Table 7.2 for more details).

7.4 Conclusion

This chapter has leveraged a new end-to-end transformer-based architecture for character detection and recognition in the wild images. The proposed method has leveraged Deformable-Patch (DPT) as a feature extraction backbone and a bounding box loss function for reading characters with different sizes, scales, and aspect ratios in the wild images. To evaluate our proposed method's performance with that of the state-of-the-art object detection approaches, we used the ICDAR13 benchmark dataset. Experimental results have shown that the proposed method outperforms the state-of-the-art methods, including the recent transformer based detectors, in terms of mean average precision. Our end-to-end robust character level detector is an essential step towards the word or text-line detection, which remains part of our future work.

Since occlusion happens at the character level, characters are the main components to be predicted from the text instances in the wild. In the next Chapter (Chapter 8), we extend the proposed framework in this Chapter for addressing the occluded-text challenge. We leverage a recent transformer-based framework in deep learning, Masked Auto Encoder (MAE), as a backbone for scene text recognition and *end-to-end scene text detection and recognition* pipelines to overcome the partial occlusion limitation.

Chapter 8

Occluded Text detection and Recognition in the Wild

The performance of existing deep-learning scene text recognition-based methods fails significantly on occluded text instances or even partially occluded characters in a text due to their reliance on the visibility of the target characters in images. This failure is often due to features generated by the current architectures with limited robustness to occlusion, which opens the possibility of improving the feature extractors and/or the learning models to better handle these severe occlusions. In this work, we first evaluate the performance of the current scene text detection, scene text recognition, and scene text spotting models using two publicly-available occlusion datasets: Occlusion Scene Text (OST) that is designed explicitly for scene text recognition, and we also prepare an Occluded Character-level using the Total-Text (OCTT) dataset for evaluating the scene text spotting and detection models. Then we utilize a very recent transformer-based framework in deep learning, namely Masked Auto Encoder (MAE), as a backbone for scene text detection and recognition pipelines to mitigate the occlusion problem. The performance of our scene text recognition and end-to-end scene text spotting models improves by transfer learning on the pre-trained MAE backbone. For example, our recognition model witnessed an average improvement of 4% word recognition accuracy on the OST dataset compared to the baseline model. Our end-to-end text spotting model achieved 68.5% F-measure performance outperforming the stat-of-the-art methods when equipped with an MAE backbone compared to a convolutional neural network (CNN) backbone on the OCTT dataset.

8.1 Introduction

Text can be occluded by itself or by an external object (Figure 8.1). Occlusion in each level can affect the performance of both detection and recognition algorithms, and due to the lack of specific real-world occluded text datasets this problem remained one of the open issues in the field of scene text detection, and recognition [67, 101, 249]. Although several remarkable breakthroughs have been made in the recent deep-learning pipeline, the accuracies of existing scene text detection and recognition methods [27, 57, 58, 67, 101, 203] suffer from the amount of occlusion that can occur to the target text in the wild images, where text can be occluded by itself or other objects. This is due to the current deep learning methods assuming training and testing data are sampled from the same distribution, and the large variability of occluders introduces a distribution gap that can lead to false detection or recognition. Figure 8.1 illustrates several examples of partially occluded text instances that can appear in the wild images.

There are some classical algorithms [251, 252] that attempt to address the partial occlusion in Optical Character Recognition (OCR). For example, Chang *et al.* [251] proposed a patch-based restoration algorithm to fill the occluded part of watermark characters before recognition. Pham *et al.* [252] proposed an algorithm for occluded number recognition. They first used SURF [253] to extract futures of the input number image and then compare the extracted future coordinates with the interest points of a database cluster. However, these algorithms are only applicable to OCR documents with clean backgrounds, and they mainly target addressing the watermark and

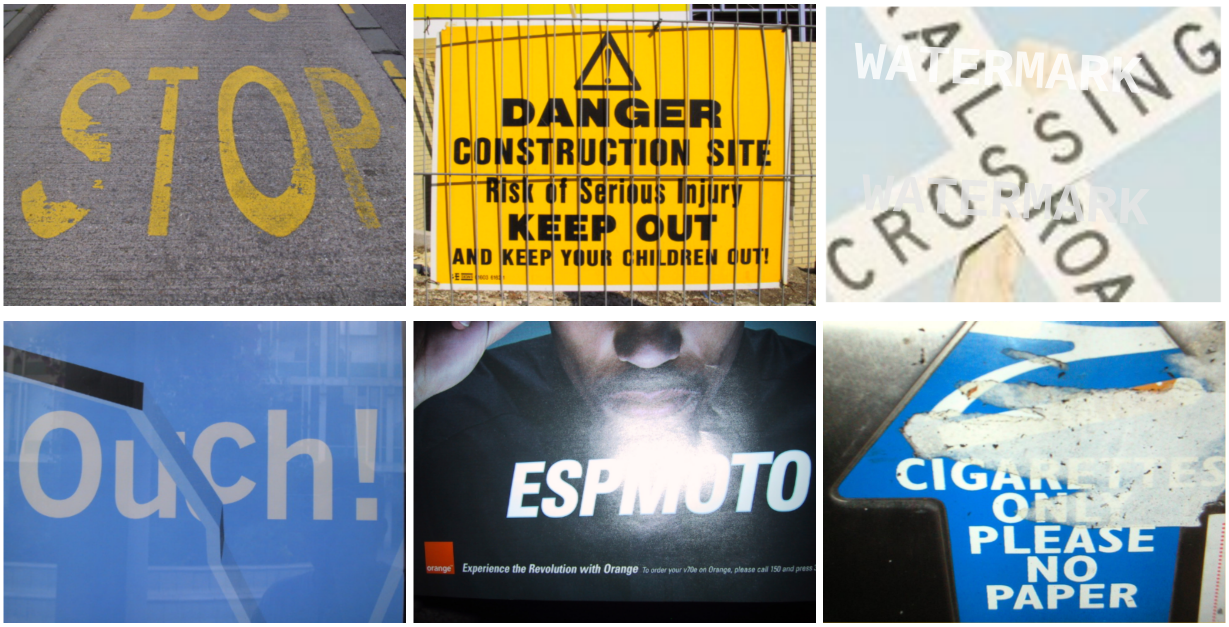


Figure 8.1: Examples of partially occluded text in the wild images. In most wild images, occlusion happens when an external object or illumination blocks a portion of some characters and when a part of a character is missing. Images are taken from the public benchmark datasets in [82, 119].

removing or restoring the missed parts of characters. Furthermore, they need prior knowledge like character stroke width makes them insufficient for occluded text spotting in the wild.

One way to address the partial occlusion problem is to use a compositionality approach, which is defined as understanding complex phenomena by breaking them into simpler parts. Using this approach, we can increase the generalization capability of a given classifier by tackling unseen scenarios. In other words, new representations can be constructed through the combination of primitive elements [254]. For example, in [255] they proposed a compositional framework that recognizes characters from their small parts of lines in handwritten characters, which achieved promising performance on the datasets that they used. However, this framework requires human interaction for creating strokes of characters. Rather than part-level semantic representation, the approach proposed in [256] learns how to represent CNN features better using

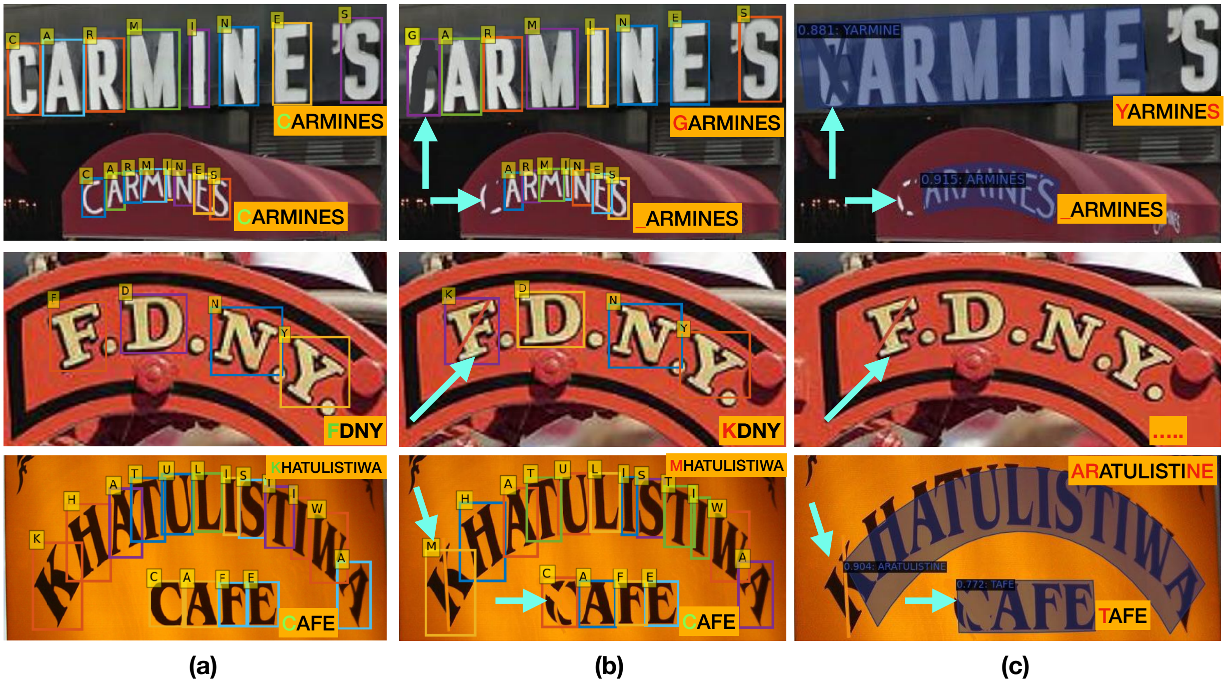


Figure 8.2: Comparison of the effect of partially occluded characters on the state of the art end-to-end scene text spotting models on some sample of Total-Text dataset [72]; results of character-based text spotting model [250] in (a) without occlusion and (b) with occlusion, and (c) shows results of [100] on occluded text instances. The "cyan" arrow, "green", and "red" characters in yellow text denote the occluded characters, correctly recognized and missed characters by the models, respectively [We replicated these results using the pre-trained models in [100, 250]]. Best viewed in color when zoomed. The above images are taken and reproduced form the public benchmark dataset [72].

object compositionality to provide a more generalizable model. Some recent methods [257–259] applied compositionality combined with CNN to make its model more robust for the detection of occluded objects. However, applying the compositionality idea to the current deep-learning architectures is more complicated. In addition, compositionality requires part-level annotations with human collaboration, which is expensive.

Masking was already introduced in the Natural Language Processing (NLP) community for concealing the input tokens to learn a strong bi-directional representation as in [260] and handle the visual and language task as in [261]. Recently, a transformer-based network, namely Masked Auto-encoders (MAEs), was proposed in [7] for masking a significant portion of the input image ($\sim 75\%$) and reconstructing the missing pixels. MAE achieved state-of-the-art (SOTA) performance in unsupervised and supervised computer vision tasks, like classification, detection, and segmentation [7, 262]. Recently several text recognition algorithms [263, 264] utilized the masking idea in combination with language models in their recognizer to to fully exploit the external language priors [263–265]. For example, Wang *et al.* [264] proposed a language-aware visual mask that achieved good performance in recognizing partially masked characters. However, these models are not explicitly designed for occlusion problems and require pre-trained language models during training and testing.

Since we can view the occluded text as a problem whose elements are masked, we apply the above MAE as a backbone for extracting more semantic features to address the challenging occluded text problem. In this work, we first leverage a pre-trained MAE backbone as the feature extractor in our transformer-based recognition architecture to predict the sequence of characters without using any language models. Then, unlike the current scene text spotting methods [59, 100, 231, 266, 267] that combine detection and recognition frameworks for final word outputting, we propose a new end-to-end scene text spotting pipeline by leveraging the MAE and Detection using Transformers (DETR) frameworks [7, 96, 201, 262] to directly predict character and word instances from a given image. Since occlusion usually happens to characters of the text instances in the wild images, we argue that this framework can better solve the occlusion challenge compared to previous end-to-end scene text spotting methods [100, 266, 267] that have a distinct separation between the detection and recognition branches (See Figure 8.2). Our contributions are as follows:

1. We evaluate and compare the performance of several SOTA scene text detection and recognition methods on partially occluded text instances.
2. We prepare an occluded dataset using Total-Text [72] dataset with annotation at the character level (See §8.4.2).
3. We propose an end-to-end scene text spotting architecture of a given image without using word-cropped image instances. This model achieves SOTA performance in occluded text instances by leveraging a pre-trained masked backbone [7].
4. We design a multitask prediction head for the end-to-end method that outputs character classes, bounding boxes, and bounding boxes for the word instances.
5. Our proposed end-to-end framework also outputs polygon representations for word instances, which alleviates the need for accurate annotations such as polygons annotations.

8.2 Related Work

8.2.1 Scene Text Recognition

In *scene text recognition (STR)*, the goal is to convert the patch of cropped word images into a sequence of characters. By using deep learning frameworks, researchers proposed many STR techniques that achieved significant performance in benchmark datasets [71, 77, 79, 81–83].

Previous deep-learning recognition methods [63, 66] have utilized convolutional neural network (CNN) as a feature extractor [164] and Recurrent Neural Networks (RNN) [6, 67, 75] for capturing sequential dependencies. Early methods [63, 66] first extract the sequential visual features using CNN and RNN blocks. Then a Connectionist Temporal Classification (CTC) [169] decoder maximizes the probability of all paths of extracted features for final prediction. These methods performed well on the horizontal and near-horizontal images of benchmark datasets [79, 82, 182]. Later, by utilizing the attention mechanism in the RNN framework and using rectification modules [11, 94], many approaches [27, 64, 67] improved the accuracy of recognizing multi-oriented and curved text instances in several benchmarks. For example, in ASTER [27], a rectification module first makes the highly curved text instance into regular and near-horizontal text, helping the RNN-based recognition pipeline to capture the linguistic information of arbitrarily shaped text instance better.

In the past few years, with the success of the transformers [1] in natural language processing and computer vision fields [193, 204, 224], several transformer-based pipelines proposed in STR that achieved superior performance in benchmarks [202, 203, 243, 263, 268]. For example, methods in [202, 203, 243, 268] proposed a 2D positional encoding with via improvement in the transformer's [1] architecture to make it suitable for the arbitrary shape of text recognition. Recently, some other transformer-based algorithms incorporated semantic knowledge into a text recognizer to fully exploit the external language priors [263–265]. For example, a new language model proposed by Fang *et al.* [263] that utilizes semantic information in their architecture in order to guide the recognition network and improve the final accuracy.

Existing methods in scene text recognition rely on the visibility of all the target characters in the given input image to form an accurate word instance as output. Nevertheless, text affected by heavy occlusion may significantly undermine the performance of these methods [27, 63–67]. This failure is often due to features generated by the current CNNs architectures that have limited robustness to occlusion, which opens the possibilities to either improve the feature extractors and/or the learning models to handle better these severe occlusions (more details in §8.4.4). Nevertheless, some methods have recently tried to take advantage of the transformers' capability and language models to address the partially occluded problem. For example, Wang *et al.* [264] proposed a language-aware visual mask that achieved superior performance. The proposed method occludes selected character regions during the training phase to enhance the text instances' visual clues in the given word image. They proved that combining visual clues and semantic knowledge improves the STR performances.

8.2.2 Scene Text Spotting

Scene text spotting, also called *end-to-end scene text detection and recognition* is one of the challenging problems in computer vision filed [9, 59, 60, 100, 231, 266, 267, 269]; In this task, the goal is to simultaneously localize and read the sequence of characters from a given image. Existing methods [9, 59, 60, 100, 231, 266, 267, 269] usually use two separate modules between the detection and recognition branches, requiring accurate annotations for the two tasks.

Early methods [269] aimed to read regular text from the wild images. For example, Li *et al.* [269] proposed the first deep-learning-based end-to-end scene text spotting technique by integrating the detection and recognition modules into a unified end-to-end framework. This method uses a shared backbone encoder, and RoIPooling [9] to feed the detection features into the recognition head via a two-stage framework. Liu *et al.* [60] proposed an efficient training framework by leveraging RoIRotate and adopted an anchor-free mechanism to extract more robust features and improve the inference speed.

Recently, several methods [59, 100, 231, 266, 267] proposed to read irregular text instances. For example, to focus more on arbitrary-shaped text regions, RoiMask proposed in [266]. Liu [100] proposed fitting a Bezier curve for arbitrary-shaped text detection joint with a BezierAlign module for rectification of curved text instances before the recognition framework. There are also efforts in [149, 270, 271] to detect and recognize characters and group them as word instances in the scene images. For example, CharNet [149] designed a weakly-supervised technique to predict individual characters and text bounding boxes using text instance detection results and grouping the predicted character boxes to form the final word results. Baek *et al.* [271] used the special character region features from the detector module as input to the attention-based recognizer module to spot arbitrary-shaped text instances accurately. In a different work, Raisi *et al.* [250] leveraged Detection using a Transformer's (DETR) architecture to directly detect and recognize individual character classes from scene images. More recently, Kittenplon *et al.* [272] introduced TextTranSpotter (TTS), by utilizing Deformable-DETR's [201] architecture for scene text spotting by proposing a multi-task prediction head that predicts both detection and recognition outputs.

Although scene text spotting approaches benefit the two detection and recognition modules respectively, they are not exceptional dealing with occluded text reading as text recognition models. As shown in Figure 8.2(c), SOTA scene text spotting methods [100, 250] fail to read the occluded characters. This failure also is more evident in methods that use attention-based module as recognizers (Figure 8.2(c)). In §8.4.4, we show how a small portion of occlusion in one character can affect the final scene text spotting performance.

8.3 Methodology

8.3.1 Occluded Scene Text Recognition

We utilize MAE [7, 262] as our feature extraction backbone, which is based on a standard the transformers [1] architecture, namely ViT [273]. MAE has an asymmetric encoder-decoder pipeline where the encoder takes the randomly unmasked (visible) patches of the given input image. The decoder reconstructs the missing pixels of the target image. Masking a large portion ($\sim 75\%$) of the input image make it robust in many supervised and unsupervised tasks due to its powerful capability of hidden representation [274].

We adopt an upgraded framework in [268] as our central scene text recognition architecture. The intuition behind using this architecture is that it is simple, and we can easily fit the pre-trained MAE in its pipeline. In our proposed scene recognition pipeline, we first fine-tune the pre-trained version of MAE (ViT-Base) [7] on 36 classes of cropped characters of Synth-Text [87], We then remove the decoder, use the MAE encoder as our feature extraction backbone, and add the rest recognition modules to recognize the sequence of characters.

8.3.2 Occluded End-to-End Scene Text Spotting

Architecture. The overall pipeline of our proposed method is shown in Figure 8.3, which uses a pre-trained backbone of MAE and an upgraded Deformable-DETR [201] detector on top of that. We select the Deformable-DETR framework [201] as our baseline detector due to its simple and powerful capability of detection as well as its good performance in many recent scene text detection [275, 276] and spotting [272] methods. As shown in Figure 8.3, the network split the input image $I = H \times W \times C$, with height H, width W, and C channels into a sequence of 2D fixed-size patches with shape of $N \times (P^2 \times C)$, where $N = HW/P^2$ is the number of patches with resultion of (P, P). Next, we use a pre-trained encoder model of MAE [7] that use a ViT Transfromer (ViT B/16) as our Network's backbone to extracts the 2D features from these input patches. However, unlike a conventional CNNs like ResNet [164] with multi-scale feature maps used in [201], the ViT encoder in MAE [7] has a ”columnar” structure [244] and generates a single-scale features f specifically designed for classification tasks, make it unsuitable for our character detection that require multi-scale features. To address this problem, we follow the [262, 279] to leverage the idea of upsampling or downsampling into the intermediate ViT's feature map with d block by using four modules that produce multi-scale features for the given resolutions input.

As shown in Figure 8.3, the first block's output feature map is upsampled by a factor of

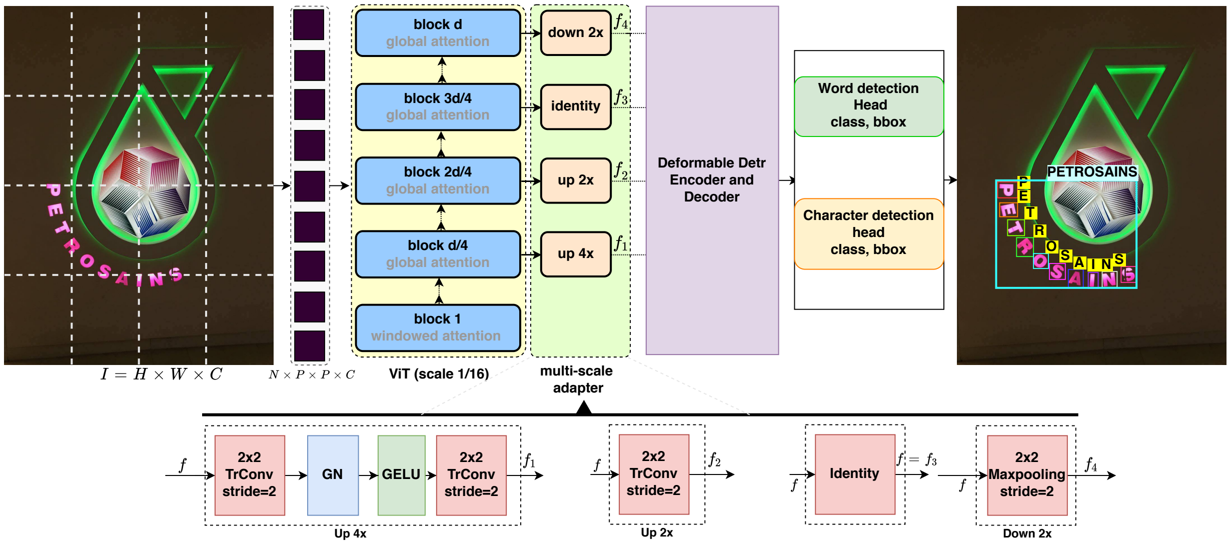


Figure 8.3: Our proposed architecture for end-to-end scene text spotting. "TrConv" means transposed convolution. The ViT backbone is modified from [262]. "GN" and "GELU" denote Group Normalization [277] and Gaussian Error Linear Units [278], respectively. Our main contributions to this architecture are proposing a multi-task prediction head and leveraging a multi-scale pretrained MAE as the backbone (See §8.3.2 for details). The right and left images are reproduced from the publicly available benchmark dataset [72].

4, resulting in f_1. The next $d/4$ block's output is upsampled by 2, producing f_2. The next $d/4$ block's output is remained unchanged, generating $f_3 = f$, and the final block's output is downsampled by a factor of 2 producing f_4. The resulted multi-scale features $\{f_1, f_2, f_3, f_4\}$ produce multi-scale feature maps contain the strides of 4, 8, 16, and 32 pixels concerning the input image, which are then fed into the upgraded [201] detector.

Different from [262] that uses Feature Pyramid Network (FPN) [5], we eliminate it because the [201] detector can be naturally extended to aggregate multi-scale features without the utilizing of FPN. We upgrade the detection head of Deformable-Detr, making it suitable for end-to-end scene text spotting. Our model's prediction head can generate the class and bounding boxes of characters and the rectangular bounding boxes of the word instances by using a loss function described as follows.

Multi-task Prediction Head. Inspired from [96, 201, 245, 272, 276] to allow the transformers' architecture to predict the characters and words of a text regions, we propose a loss function based on the optimal bipartite matching between N predicted $\hat{y}$ and ground-truth y capable it for finding the one-vs-one matching $\hat{\sigma}$ using the Hungarian algorithm [194] tailored to the task at hand as follows:

$$\hat{\sigma} = \text{arg} \min_{\sigma \in \mathfrak{S}_N} \sum_{i}^{N} \mathcal{L}_{\text{match}}(y_i, \hat{y}_{\sigma(i)}), \tag{8.1}$$

where $\mathfrak{S}_N$ denotes the set of possible matches.

$$\begin{aligned} \mathcal{L}_{\text{match}}(y, \hat{y}_{\sigma(i)}) = -\alpha_c \hat{p}_{\sigma(i)}(c_i) - \alpha_w \hat{p}_{\sigma(i)}(w_i) + \\ \mathbb{1}_{\{c_i \neq \varnothing\}} \alpha^c_{\text{box}} \mathcal{L}^c_{\text{box}}(b_i, \hat{b}_{\sigma(i)}) + \mathbb{1}_{\{w_i \neq \varnothing\}} \alpha^w_{\text{box}} \mathcal{L}^w_{\text{box}}(t_i, \hat{t}_{\sigma(i)}), \end{aligned} \tag{8.2}$$

where c_i and w_i denote ground truth class for the characters and word instances, $\hat{p}_{\sigma(i)}(c_i)$ and $\hat{p}_{\sigma(i)}(w_i)$ are the predicted probability for class c_i and class w_i. b_i and t_i denote the bounding box of character and word instances, and α_c, α_w, α^c_{box} and α^w_{box} are the weights for the character classification, word classification, character bounding box, word bounding box criteria, respectively. Then we define the Hungarian loss function $\mathcal{L}_{\text{Hung}}$ as follows:

$$\begin{aligned}\mathcal{L}_{\text{Hung}}(y, \hat{y}) = \sum_{i=1}^{N} -\beta_c \log \hat{p}_{\sigma(\hat{i})}(c_i) - \beta_w \log \hat{p}_{\sigma(\hat{i})}(w_i) + \\ \mathbb{1}_{\{c_i \neq \varnothing\}} \beta^c_{\text{box}} \mathcal{L}^c_{\text{box}}(b_i, \hat{b}_{\hat{\sigma}}(i)) + \mathbb{1}_{\{w_i \neq \varnothing\}} \beta^w_{\text{box}} \mathcal{L}^w_{\text{box}}(t_i, \hat{t}_{\hat{\sigma}}(i)) \quad (8.3)\end{aligned}$$

Where β_c, β_w are classification weights of characters and words. β^c_{box} and β^w_{box} denote the weights of character and word bounding box. $\mathcal{L}^{(.)}_{\text{box}}$ which is defined as a linear combination of a Smooth-$\ln$ based regression loss [2] and Generalized Intersection over Union (GIoU) loss [195] as proposed in [96].

Arbitrary-shaped Text Representation. As shown in Figure 8.4, during inference time, using the coordinates of character bounding boxes and the predicted word boxes, we can also output a polygon representation for each word instance in a given image without using any polygon annotation during training. We start from the top-left points of the first character, use the top-middle points of the central characters, and end with the top-right points of the last character. We also repeat this process by using the bottom-left, bottom-middle, and bottom-right points to generate the bottom polygon using the first, middle, and last characters, respectively.

8.4 Experimental Results

8.4.1 Implementation Details

We train all our final STR and scene text spotting models on 4 GPUs of NVidia A100, and we use a pre-trained encoder backbone of MAE (ViT-Base/16) [7] and fine-tune it more on 1M cropped alphanumeric characters of [87]. We follow the same setting of [268, 280] to train our

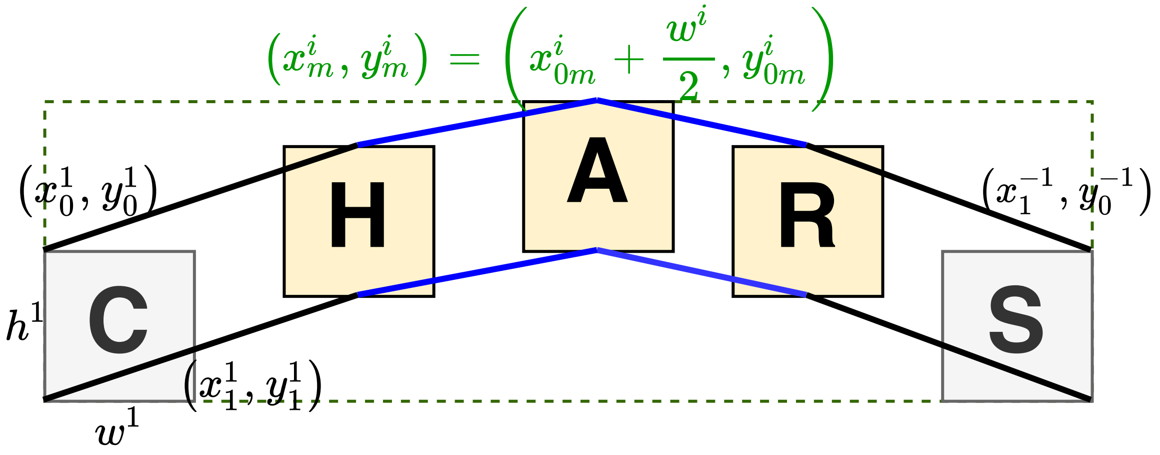


Figure 8.4: Creating a polygon representation for the arbitrary shape of word instances (black and blue lines) using the bounding box coordinates of the detected characters.

STR model. For the scene text spotting model, we set the number of object queries to 300 and use an AdamW [211] optimizer to optimize the parameters of the model. For augmentation, we use different rotation angles $[90°, 180°, 270°]$, and resize the input image as in [201]. We train this model using a batch size of 2 for 24 epochs on the synthetic character and bounding box annotations of [87] and then fine-tune on the real-world dataset for 300 epochs with a learning rate of 1×10^{-4}.

8.4.2 Datasets

Occlusion Scene Text (OST) dataset is a new benchmark dataset that is prepared to evaluate the performance of STR methods for occluded text recognition [264]. OST consists of manually occluded images of six well-known public benchmark datasets: IC13 [82], IC15 [71], IIIT5K [79], SVT [77], SVTP [81] and CUTE80 [83] datasets. This dataset has 4832-word images of a weak and heavy level degree of occlusion, which in each image at least one character randomly occluded by using one or two lines; these lines have a similar background color to the characters

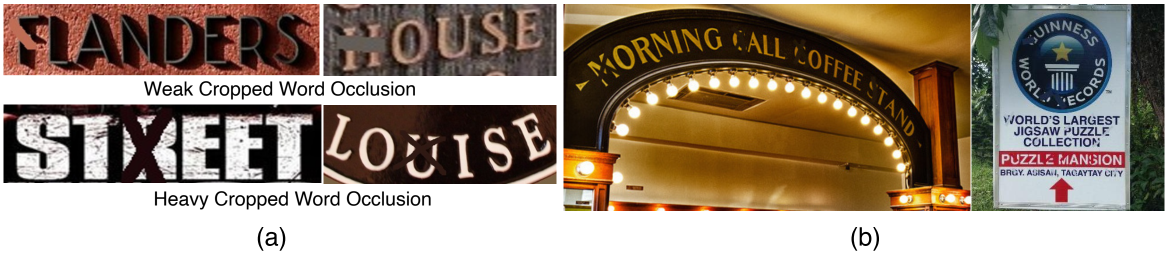


Figure 8.5: Examples of manually occluded text instances; Images in (a) are cropped words in the publicly available OST [264] dataset with heavy and weak occlusion, and images in (b) are occluded text images of the OCTT dataset. "Weak" and "heavy" denote the degree of random occlusion using one and two lines to cover the characters, respectively. The images in (b) are reproduced from the benchmark dataset [72]. All the images in above are from public benchmark dataset available in [264].

to make the occlusion more realistic. Figure 8.5(a) illustrates some sample images of the OST dataset.

Occluded Character-level Total-Text (OCTT) dataset is a new dataset proposed in this work for evaluating occluded text in scene text spotting from the raw images of the test set of the Total-Text [72] dataset. We selected Total-Text because it has various arbitrary shapes of text instances, including multi-oriented and curved text. OCTT contains 300 images that at least one character is weakly occluded following the same procedure described in [264]. Some examples of this dataset are shown in Figure 8.5(b).

8.4.3 Evaluation Metrics

For evaluating our recognition task, we use the well-known Word Recognition Accuracy (WRA) metric, that commonly used in measuring the accuracy of STR methods [27, 67, 101]. For evaluating of detection and end-to-end spotting models we use the same metric in [100, 272]. Similar to [250] we also use mean average precision (AP) as our evaluation metric adopted as a standard in many recent object detection algorithms to evaluate the 36 alphanumerical (10 digits + 26 capital letters) characters directly spotted in the OCTT dataset.

8.4.4 Quantitative Results

Comparison SOTA Recognition Methods on OST Dataset. We evaluate the performance of some SOTA methods on OST [264] datasets in Table 8.1. As shown, all methods' recognition accuracy declines by a large margin when applied to occluded text. This decline is more evident in RNN-based methods [27, 63–67]. For example, ASTER [27] that has the best average WRA on all sets of word instances without occlusion, $\sim 86\%$, witnessed $\sim 26\%$ and $\sim 46\%$ WRA decrease on OST dataset that is the same images with weak and heavy occlusion on only one character, respectively. However, the WRA performance was better for the transformer-based methods like 2DSPE [203] and 2LSPE [243] and recent method, VisionLAN [264], that uses of pre-trained language model in its framework.

On the other hand, our proposed method that takes advantage of MAE in its backbone outperforms the SOTA methods in OST by $\sim 5\%$ on average on the OST dataset. This performance confirms that a pre-trained masked backbone during training can improve the WRA on occluded text.

Comparison SOTA Text Spotting Methods on OCTT Dataset. We also evaluate the performance of some SOTA scene text detection and scene text spotting methods in Table 8.2. As seen, occlusion does not affect detection performance but leads to a big difference in text spotting methods. For example, the H-mean performance of scene text detection of ABC-Net [100] only declines $\sim 1\%$ when occluded applied, while seeing a large margin of decline $\sim 13\%$ during measuring text spotting metric for F-measure. This performance brings us to conclude that occlusion affects more on the recognition models more than detection. However, Our proposed

Table 8.1: Comparison of STR methods with our proposed method on the OST [264] dataset using WRA metric. Weak and Heavy mean weak and heavy occluded of characters. For evaluating of without occlusion, we use the average WRA of the [71, 77, 79, 81–83] datasets using the pre-trained models. "Weak" and "heavy" denote the degree of random occlusion using one and two lines to cover the characters, respectively. The best performance is highlighted in **bold**.

Method	**Weak occlusion**	**Heavy occlusion**	**without occlusion**
CRNN [63]	53.43%	37.33%	78.13%
RARE [64]	56.29%	38.20%	79.14%
ROSETTA [66]	55.50%	34.06%	79.70%
STAR-Net [65]	63.07%	42.05%	82.59%
CLOVA [67]	66.47%	47.55%	84.37%
ASTER [27]	60.90%	40.80%	86.10%
VisionLAN [264]	70.30%	50.30%	**89.11%**
2DSPE [203]	70.70%	55.80%	87.49%
2LSPE [243]	71.88%	57.04%	88.16%
Baseline [268]	69.20%	50.82%	86.20%
Ours	**75.32%**	**62.40%**	87.60%

end-to-end scene text spotting model by using the word instances of rectangular bounding boxes outperformed the SOTA model [100] in the OCTT dataset.

We also study the performance of end-to-end text spotting at the character level using the OCTT dataset. As shown in Table 8.3, the AP performance of recent detection models [201, 244, 245, 250] decrease $\sim 4\%$ compare to OCTT without occlusion. However, our proposed end-to-end text spotting model outperformed the SOTA detectors in terms of AP performance for both cases of occluded or not occluded characters of the OCTT dataset.

8.4.5 Qualitative Results

Figure 8.7 shows the qualitative results of our recognition model in §8.3.1 on the OST [264] dataset. As shown, the proposed model recognize correctly the occluded cropped word instances, where most of the SOTA methods [27, 63–67] failed to recognize these occluded word instances

Table 8.2: Detection and end-to-end spotting results on the OCTT dataset. E2E denotes end-to-end. P, R, H denote Precision, Recall, and H-mean, respectively. The best performance is highlighted in **bold**. Occluded results are marked in gray.

Model	Occlusion	Detection			E2E
		P	R	H	F-measure
ABC [100]	-	82.3	**86.9**	84.5	63.0
	yes	80.8	85.6	83.1	50.8
APT [275]	-	**89.1**	86.4	**87.8**	-
	yes	**87.8**	**85.7**	**86.7**	-
Ours	-	-	-	-	**76.3**
	yes	-	-	-	**68.5**

Table 8.3: Results of end-to-end character spotting on OCTT dataset. AP means average precision.

Model	AP (Original)	AP (Occluded)
Baseline [201]	0.43	0.39
PVT [244]	0.44	0.41
Sparse R-CNN [245]	0.45	0.41
E2ESC [250]	0.46	0.42
Ours	**0.51**	**0.49**

Table 8.4: Classification accuracy results in the occluded characters of the OCTT dataset.

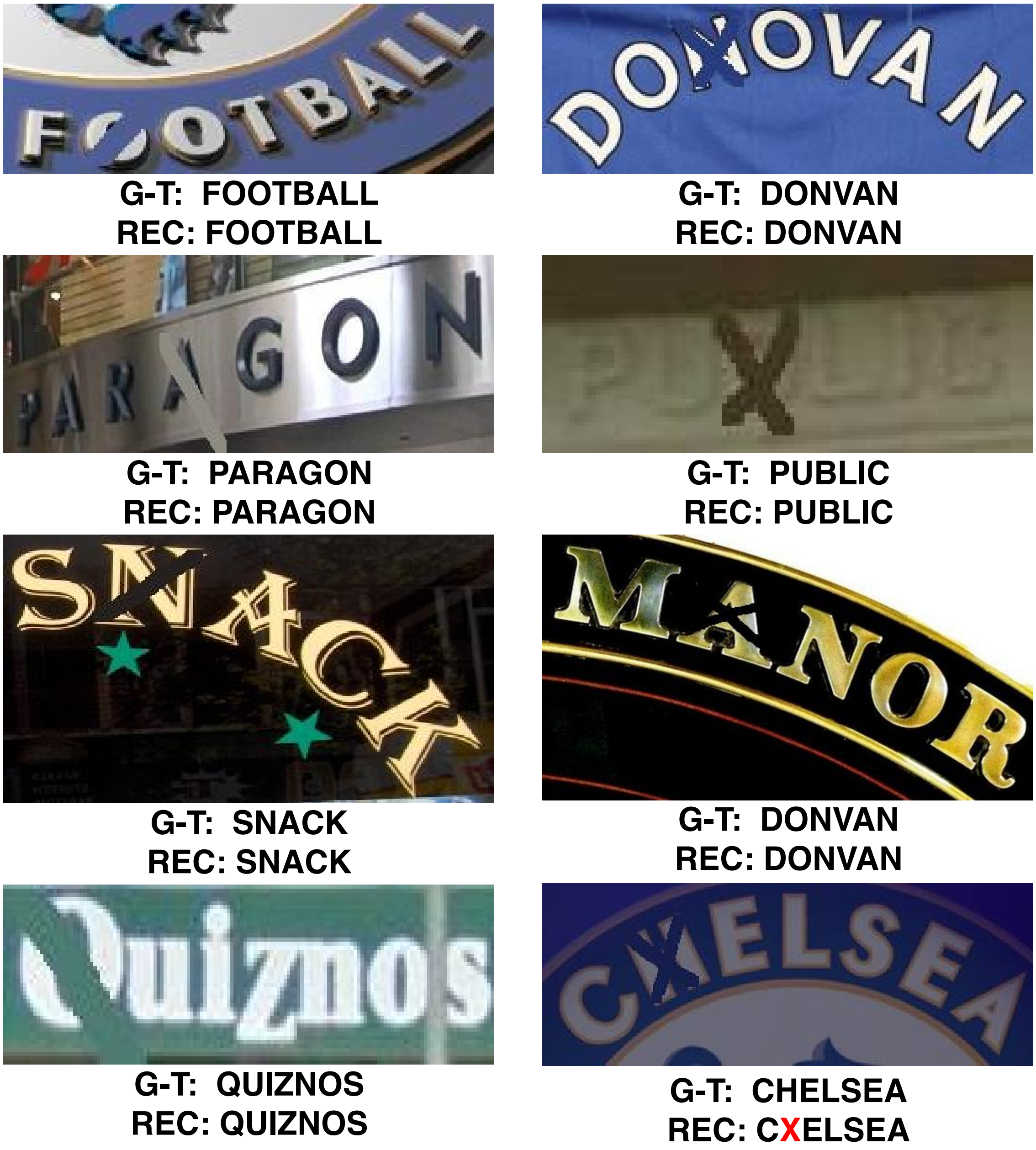
Model	Accuracy
ResNet	83.2
Fine-Tuned MAE	**92.5**

especially the heavy occluded in right column. We also provide qualitative results of our proposed end-to-end scene text spotting model (§8.3.2) on the OCTT dataset in Figure 8.6. As it can be seen, the proposed method performs well on the occluded text instances with arbitrary shapes.

8.4.6 Ablation Study

Effect of MAE on Occluded Characters. For this experiment, we use MAE that utilizes a ViT-Base (ViT-B/16) [273] as the backbone in our ablation study. We fine-tune the MAE [7] using cropped characters of SynthText [87] and applied it to the occluded characters. We also compare the classification accuracy of ResNet [164] and our fine-tuned model on cropped characters of OCTT. Table 8.4 demonstrates the results. The fine-tuned MAE outperformed the ResNet50 model with a large margin on this dataset. The qualitative results on some occluded characters of the OCTT dataset in Figure 8.8 demonstrate how a fine-tuned MAE model can minimize the occluders' effect.

Classification Alphanumeric Characters. To see how masked autoencoders are capable of restoring the masked digits and characters, we apply an MAE with the setting shown in Table 8.5. Increasing the number of encoder and decoder layers increases the classification accuracy for both digits and cropped characters. We then use a pre-trained model of [7] and fine-tune it on the cropped characters of [72] dataset; This improved the classification accuracy of both MNIST [281] and the cropped characters of the OCTT dataset. Figure 8.9 illustrates the qualitative results of fine-tuning on the MNIST [281] and cropped characters using a masked autoencoder model



G-T: FOOTBALL
REC: FOOTBALL

G-T: DONVAN
REC: DONVAN

G-T: PARAGON
REC: PARAGON

G-T: PUBLIC
REC: PUBLIC

G-T: SNACK
REC: SNACK

G-T: DONVAN
REC: DONVAN

G-T: QUIZNOS
REC: QUIZNOS

G-T: CHELSEA
REC: CXELSEA

Figure 8.6: Experiment results of the proposed model that successfully recognized the occluded cropped words of OST [264] dataset. "G-T" and "REC" mean ground-truth and recognition, respectively. The images in the left and right columns are selected from weak and heavy samples in OST, respectively. The missed characters are shown in red. The result images are reproduced from publicly available benchmark dataset [72].

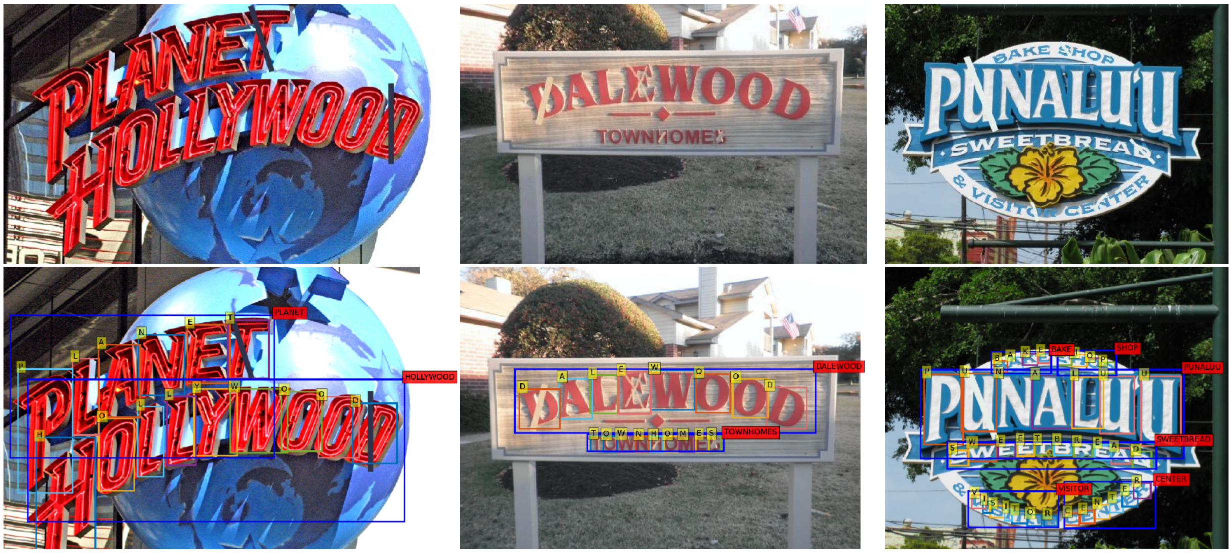


Figure 8.7: Experiment results of the proposed model that successfully spotted the occluded text instances of the occluded input images. The top images are the original images with occlusion, and the bottom are our end-to-end text spotting results (Best viewed in color when zoomed). The result character images are reproduced from publicly available benchmark dataset [72].

with six encoders and two decoder layers of Table 8.5.

8.5 Conclusion

In this work, we have utilized the recent transformers-based backbone, MAE, in our STR and scene text spotting at character level frameworks to address the occlusion problem in the wild

Table 8.5: The effect of different number of encoder, decoder layers and masking ratios on recognition accuracy in MNIST [281] and cropped characters of Total-Text.

Pre-train	encoder	decoder	mask ratio	MNIST	CTT
-	6	2	0.75	96.0	85.49
-	6	4	0.75	97.3	87.35
Image-Net	12	6	0.75	99.9	94.52

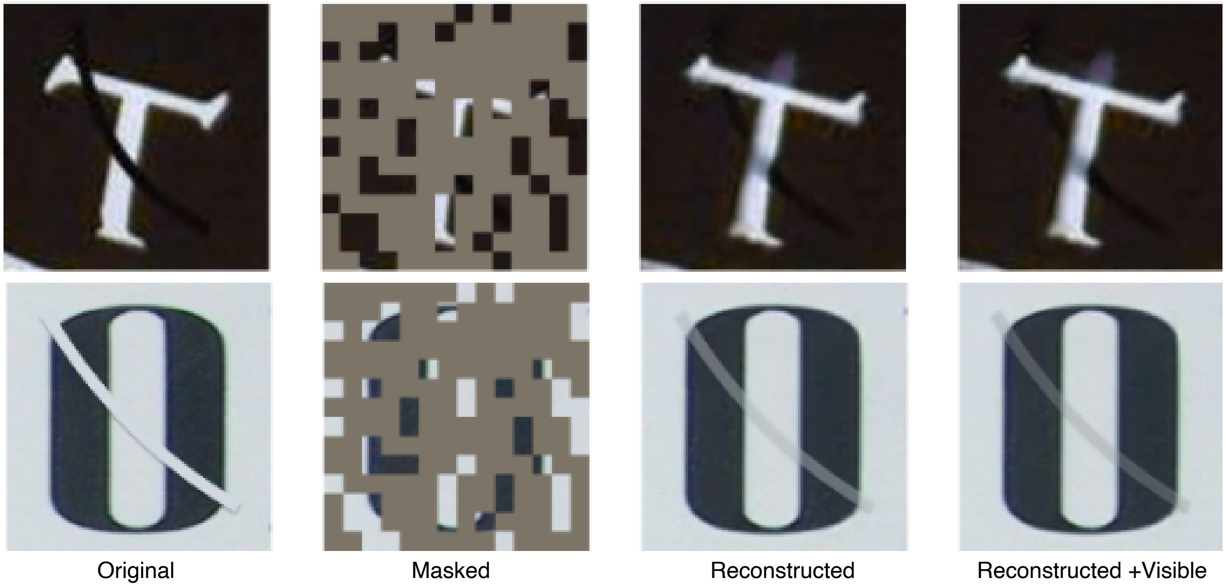


Figure 8.8: Effect of MAE backbone on some partially occluded cropped character samples. The sample characters are from the OCTT dataset. As shown, the MAE decreased the effect of occluded line in both images, which later makes easier recognizing the occluded characters.

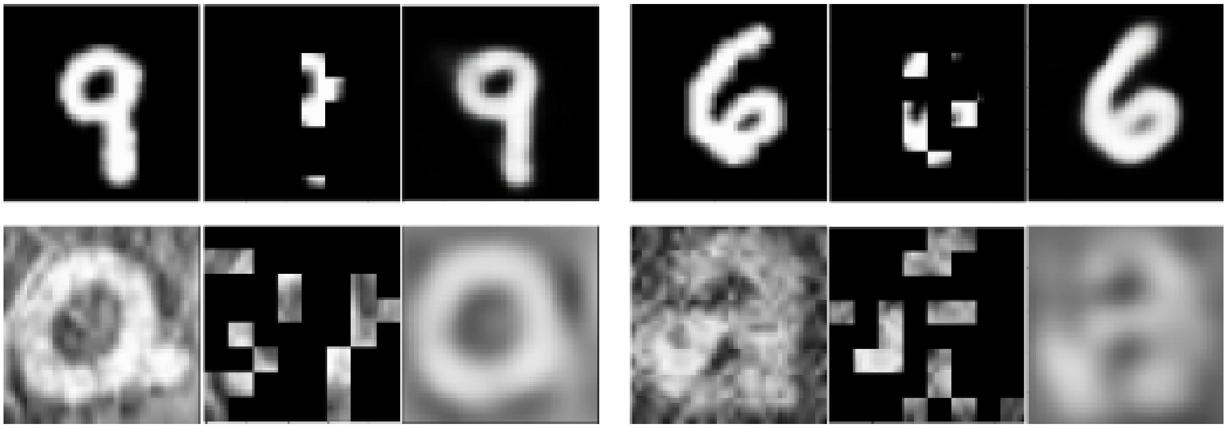


Figure 8.9: Sample masked image (middle) with masking ratio is 75%, MAE reconstruction (right), and the ground-truth (left). The images on top are from the MNIST dataset [281], and down images are cropped scene text characters from OCTT dataset reproduced from public dataset in [72].

images. In addition, we have proposed an end-to-end scene text spotting at the character level that directly reads characters in a given image and aggregates them into a word. To evaluate our proposed method and compare its performance with SOTA techniques in partially occluded characters in scene text recognition and end-to-end scene text spotting, we have used the OST dataset and our new proposed OCTT datasets, respectively. The experimental results have shown that our proposed models achieved SOTA performance on the OST and OCTT datasets that confirm the effectiveness of the MAE backbone in addressing occluded text recognition and spotting.

In the last chapter of this thesis (Chapter 9), we summarize and conclude the proposed techniques for detecting and recognizing of irregular and occluded text challenges in the wild images. We also discuss some interesting future directions that can be used for improving the detection and recognition performance of the mentioned challenges.

Chapter 9

Conclusion

Scene text detection and recognition methods that utilized deep-learning in their pipelines have witnessed tremendous progress in recent years. These methods achieve superior performance when the text's shape in the image is a regular and clean background. However, several important remaining challenges limit the performance of the current state-of-the-art methods, such as irregular text (e.g., curved, multi-oriented, and vertical text) and the presence of occlusion.

In this thesis, we have presented a detailed review of the recent advancement in scene text detection and recognition fields, focusing on deep learning-based techniques and architectures. We have leveraged the transformer's architecture for both detection and recognition for tackling the irregular-text problem. For scene text detection, we have designed a new predictor that aims to infer n-vertices of a polygon or the degree of a Bezier curve to better represent irregular-text regions and a loss function that is more precise in measuring the changes in scales and aspect ratios of the detected text regions.

For scene text recognition, we have used a 2D positional encoder with the transformer architecture, which better preserves the spatial information in 2D images than the prior methods for irregular text recognition. Furthermore, we have proposed a new feed-forward-network layer in the encoder module making it more robust in capturing the features generated by the encoder's self-attention mechanism. We also have extended the capabilities of a recently proposed positional encoder with learnable sinusoidal frequencies from one-dimensions to a two-dimensions format. Moreover, we have shown how to incorporate the learned positions within the multi-head self-attention (MHSA) of the transformer's architecture for scene text recognition.

For end-to-end scene text spotting, we have leveraged a new end-to-end transformer-based architecture for character spotting in the wild images. The proposed method has leveraged Deformable-Patch (DPT) as a feature extraction backbone and a bounding box loss function for reading characters with different sizes, scales, and aspect ratios in the wild images. In addition, we have proposed an end-to-end arbitrary shaped text spotting architecture using a multi-scale vision transformer encoder as a backbone followed by an upgraded transformer-based detector capable of outputting characters and words as well as their rectangular and polygon bounding boxes representations.

To address partially occluded text in the wild images, we have used a pre-trained masked autoencoder pipeline integrated into our current text recognition and spotting networks. We have prepared a partially occluded text (OCTT) dataset annotated at the character level explicitly designed for occluded text spotting.

We also have conducted experiments on different challenging benchmark datasets to compare the performance of our proposed methods with state-of-the-art scene text detection and recognition methods. Our quantitative and qualitative experimental results can be summarized as follows:

- For scene text detection, we have presented a transformer-based architecture for multi-oriented and curved text detection in the wild. Our best proposed model that uses a 3 splits rotated rectangular loss function achieves the best H-mean performance of 87.8% and 87.2% for Total-Text and CTW-1500 datasets, respectively. Our system also exhibits

SOTA performance in Recall (85.0%) and H-mean (88.1%) on the MSRA-TD500 dataset and yield competitive results for ICDAR15 benchmarks.

- For scene text recognition, it has been shown that our proposed model outperformed the state-of-the-art methods in terms of word recognition accuracy on several challenging datasets. Furthermore, the effect of different PE schemes on the transformer's architecture has been studied. The proposed 2D sinusoidal PE technique with learnable frequencies has outperformed the baseline method that uses fixed PE frequencies in terms of recognition accuracy in all cases.
- For end-to-end recognition, experimental results have shown that the proposed method outperforms the state-of-the-art methods, including recent transformer based detectors, in terms of mean average precision.
- For the text occlusion problem, our scent text recognition and scene text spotting models have shown the best performance in terms of average WRA and F-measure in the OST (68.86%) and OCTT (68.5%) datasets by outperforming the best methods by a large margin of $\sim 5\%$ and $\sim 18\%$, respectively.

In this thesis, we have taken a few steps towards addressing the detection and recognition of irregular and occluded text challenges in the wild images; however, we believe there is still much room for improvement in our proposed models. In future work, we are considering further optimization of our algorithm's shortcomings. The direction of possible investigation in existing open areas of text detection, recognition, and end-to-end text spotting frameworks is as follows:

(1) Similar to other text detection and text spotting methods, our proposed frameworks are not *generalizable* (i.e., training on one dataset and testing on a different dataset). The main reason for this is that the current real-world datasets for text detection have too few images for training (1k - 2k). One solution for addressing this problem could be designing a large set of training datasets suitable for real-world challenges.

(2) In contrast to our scent text recognition models that only require *synthetic datasets* for training, scene text detection and end-to-end text spotting models require both synthetic samples for pre-training and real-world images for fine-tuning. However, preparing a large real-world dataset is expensive. Researchers can use generative adversarial network [192] based methods or 3D proposal based [282] models for producing more realistic text images that can be a better way of generating synthetic datasets for training text detectors. For example, by utilizing the recent work in [283], we can generate real-world like text images, which can be used directly for the training of our models.

The other way could be utilizing *unsupervised* or *weakly-supervised* techniques in existing current architectures to alleviate the need for annotation of a real-world dataset. For example, in a weakly-supervised setting, we can benefit from the existing partial transcript annotation provided for the image during training as in [272] without complete expensive annotations. Furthermore, in a weakly-supervised manner, we can train our model with a more straightforward detection representation like a rectangular bounding box instead of polygon or segmentation masks annotation as proposed in [284].

(3) One of the shortcomings in our irregular text detection and end-to-end spotting models is *efficiency*. Our models require multiple powerful GPUs to train and at least one GPU to obtain a high FPS during inference. Furthermore, with the availability of handheld devices, it is difficult to use these models in real-world applications via smartphones; for example, blind navigation, assisting tourists or drivers on streets, label reading of products for costumers in supermarkets, and reading the hand-writing of a doctor or classroom teacher. It is worth while for researchers to utilize model compression techniques or design lightweight models that can be used on handheld devices.

Concerning the *occlusion* challenge, our work in this thesis only evaluated partially occluded text instances prepared with human interaction on one/two characters. However, text affected by heavy occlusion on *more characters* in real-world scenarios may significantly undermine the performance of our model. In addition, to the best of our knowledge, there is not a *publicly available benchmark dataset* for the occluded text detection/recognition in the wild; we discuss some possible solutions as follows:

(i) Since occlusion mainly happens on characters, designing a sizeable *real-world occluded dataset* annotated at the character-level with additional occlusion information will help future researchers to design more robust algorithms for addressing the occlusion problem. In addition, as we mentioned in §8.1, one way to tackle the occlusion problem is leveraging the *compositionality* idea [255] in our current deep-learning models may help solve the occlusion problem. However, compositionality requires a specific type of dataset with human interaction that are not available in the text detection/recognition field for researchers.

(ii) As humans, we usually use *prior-knowledge* to guess the missed parts of text instances. Current NLP models have superior performance in predicting missed words/text in a sentence. Therefore, designing a text recognition scheme based on a solid *semantic NLP model* can help better predict occluded characters. For example, we can feed the extracted visual features of an input image into a pre-trained language model in an iterative feedback setting to refine more contextual features and improve the final output's confidence on character prediction [263, 285–287].

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[2] Z. Raisi, M. A. Naiel, G. Younes, S. Wardell, and J. S. Zelek, “Transformer-based text detection in the wild,” in *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2021, pp. 3162–3171.

[3] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2015, pp. 3431–3440.

[4] W. Wang, E. Xie, X. Song, Y. Zang, W. Wang, T. Lu, G. Yu, and C. Shen, “Efficient and accurate arbitrary-shaped text detection with pixel aggregation network,” in *Proc. IEEE Int. Conf. on Comp. Vision*, 2019, pp. 8440–8449.

[5] T.-Y. Lin, P. Dollar, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit. (CVPR)*, July 2017.

[6] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[7] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, “Masked autoencoders are scalable vision learners,” *arXiv preprint arXiv:2111.06377*, 2021.

[8] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, “Pyramid scene parsing network,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2017, pp. 2881–2890.

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

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

[11] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," in *Proc. Int. Conf. on Neural Inf. Process. Syst. - Volume 2*. MIT Press, 2015, pp. 2017–2025.

[12] Q. Ye and D. Doermann, "Text detection and recognition in imagery: A survey," *IEEE Trans. Pattern Anal. Mach. Intell*, vol. 37, no. 7, pp. 1480–1500, 2015.

[13] S. Long, X. He, and C. Yao, "Scene text detection and recognition: The deep learning era," *CoRR*, vol. abs/1811.04256, 2018.

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

[15] C. Case, B. Suresh, A. Coates, and A. Y. Ng, "Autonomous sign reading for semantic mapping," in *Proc. IEEE Int. Conf. on Robot. and Automation*, 2011, pp. 3297–3303.

[16] I. Kostavelis and A. Gasteratos, "Semantic mapping for mobile robotics tasks: A survey," *Robot. and Auton. Syst.*, vol. 66, pp. 86–103, 2015.

[17] Y. K. Ham, M. S. Kang, H. K. Chung, R.-H. Park, and G. T. Park, "Recognition of raised characters for automatic classification of rubber tires," *Optical Eng.*, vol. 34, no. 1, pp. 102–110, 1995.

[18] V. R. Chandrasekhar, D. M. Chen, S. S. Tsai, N.-M. Cheung, H. Chen, G. Takacs, Y. Reznik, R. Vedantham, R. Grzeszczuk, J. Bach *et al.*, "The stanford mobile visual search data set," in *Proc. ACM Conf. on Multimedia Syst.*, 2011, pp. 117–122.

[19] S. S. Tsai, H. Chen, D. Chen, G. Schroth, R. Grzeszczuk, and B. Girod, "Mobile visual search on printed documents using text and low bit-rate features," in *Proc. IEEE Int. Conf. on Image Process.*, 2011, pp. 2601–2604.

[20] D. Ma, Q. Lin, and T. Zhang, "Mobile camera based text detection and translation," *Stanford University*, 2000.

[21] E. Cheung and K. H. Purdy, "System and method for text translations and annotation in an instant messaging session," Nov. 11 2008, US Patent 7,451,188.

[22] W. Wu, X. Chen, and J. Yang, "Detection of text on road signs from video," *IEEE Trans. on Intell. Transp. Sys.*, vol. 6, no. 4, pp. 378–390, 2005.

[23] S. Messelodi and C. M. Modena, "Scene text recognition and tracking to identify athletes in sport videos," *Multimedia Tools and Appl.*, vol. 63, no. 2, pp. 521–545, Mar 2013.

[24] P. J. Somerville, "Method and apparatus for barcode recognition in a digital image," Feb. 12 1991, US Patent 4,992,650.

[25] D. Chen, "Text detection and recognition in images and video sequences," Tech. Rep., 2003.

[26] A. Chaudhuri, K. Mandaviya, P. Badelia, and S. K. Ghosh, *Optical Character Recognition Systems for Different Languages with Soft Computing*, ser. Studies in Fuzziness and Soft Computing. Springer, 2017, vol. 352.

[27] B. Shi, M. Yang, X. Wang, P. Lyu, C. Yao, and X. Bai, "Aster: An attentional scene text recognizer with flexible rectification," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2018.

[28] K. I. Kim, K. Jung, and J. H. Kim, "Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 12, pp. 1631–1639, 2003.

[29] X. Chen and A. L. Yuille, "Detecting and reading text in natural scenes," in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit (CVPR)* ., vol. 2, 2004, pp. II–II.

[30] S. M. Hanif and L. Prevost, "Text detection and localization in complex scene images using constrained adaboost algorithm," in *Proc. Int. Conf. on Doc. Anal. and Recognit.*, 2009, pp. 1–5.

[31] K. Wang, B. Babenko, and S. Belongie, "End-to-end scene text recognition," in *Proc. Int. Conf. on Comp. Vision*, 2011, pp. 1457–1464.

[32] J.-J. Lee, P.-H. Lee, S.-W. Lee, A. Yuille, and C. Koch, "Adaboost for text detection in natural scene," in *Proc. Int. Conf. on Document Anal. and Recognition*, 2011, pp. 429–434.

[33] A. Bissacco, M. Cummins, Y. Netzer, and H. Neven, "PhotoOCR: Reading text in uncontrolled conditions," in *Proc. IEEE Int. Conf. on Comp. Vision*, 2013, pp. 785–792.

[34] K. Wang and J. A. Kangas, "Character location in scene images from digital camera," *Pattern recognition*, vol. 36, no. 10, pp. 2287–2299, 2003.

[35] C. Mancas Thillou and B. Gosselin, "Spatial and color spaces combination for natural scene text extraction," in *Proc. IEEE Int. Conf. on Image Process.* IEEE, 2006, pp. 985–988.

[36] C. Mancas-Thillou and B. Gosselin, "Color text extraction with selective metric-based clustering," *Comp. Vision and Image Understanding*, vol. 107, no. 1-2, pp. 97–107, 2007.

[37] Y. Song, A. Liu, L. Pang, S. Lin, Y. Zhang, and S. Tang, "A novel image text extraction method based on k-means clustering," in *Seventh IEEE/ACIS International Conference on Computer and Information Science (icis 2008).* IEEE, 2008, pp. 185–190.

[38] W. Kim and C. Kim, "A new approach for overlay text detection and extraction from complex video scene," *IEEE transactions on image processing*, vol. 18, no. 2, pp. 401–411, 2008.

[39] T. E. De Campos, B. R. Babu, M. Varma *et al.*, "Character recognition in natural images," in *Proc. Int. Conf. on Comp. Vision Theory and App. (VISAPP)*, vol. 7, 2009.

[40] Y.-F. Pan, X. Hou, and C.-L. Liu, "Text localization in natural scene images based on conditional random field," in *Proc. Int. Conf. on Document Anal. and Recognition*, 2009, pp. 6–10.

[41] W. Huang, Y. Qiao, and X. Tang, "Robust scene text detection with convolution neural network induced mser trees," in *Proc. Eur. Conf. on Comp. Vision*. Springer, 2014, pp. 497–511.

[42] Z. Zhang, W. Shen, C. Yao, and X. Bai, "Symmetry-based text line detection in natural scenes," in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2015, pp. 2558–2567.

[43] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, "Reading text in the wild with convolutional neural networks," *Int. J. of Comp. Vision*, vol. 116, no. 1, pp. 1–20, 2016.

[44] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman., "Deep structured output learning for unconstrained text recognition," 2015.

[45] Z. Tian, W. Huang, T. He, P. He, and Y. Qiao, "Detecting text in natural image with connectionist text proposal network," in *Proc. Eur. Conf. on Comp. Vision*. Springer, 2016, pp. 56–72.

[46] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, and J. Liang, "EAST: an efficient and accurate scene text detector," in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2017, pp. 5551–5560.

[47] M. Liao, B. Shi, X. Bai, X. Wang, and W. Liu, "Textboxes: A fast text detector with a single deep neural network," in *Proc. AAAI Conf. on Artif. Intell.*, 2017.

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

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

[50] Z. Zhang, C. Zhang, W. Shen, C. Yao, W. Liu, and X. Bai, "Multi-oriented text detection with fully convolutional networks," in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2016, pp. 4159–4167.

[51] C. Yao, X. Bai, N. Sang, X. Zhou, S. Zhou, and Z. Cao, "Scene text detection via holistic, multichannel prediction," *arXiv preprint arXiv:1606.09002*, 2016.

[52] Y. Wu and P. Natarajan, "Self-organized text detection with minimal post-processing via border learning," in *Proc. IEEE Int. Conf. on Comp. Vision*, 2017, pp. 5000–5009.

[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 *Proc. Eur. Conf. on Comp. Vision (ECCV)*, 2018, pp. 20–36.

[54] D. Deng, H. Liu, X. Li, and D. Cai, “Pixellink: Detecting scene text via instance segmentation,” in *Proc. AAAI Conf. on Artif. Intell.*, 2018.

[55] P. Lyu, C. Yao, W. Wu, S. Yan, and X. Bai, “Multi-oriented scene text detection via corner localization and region segmentation,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2018, pp. 7553–7563.

[56] H. Qin, H. Zhang, H. Wang, Y. Yan, M. Zhang, and W. Zhao, “An algorithm for scene text detection using multibox and semantic segmentation,” *Applied Sciences*, vol. 9, no. 6, p. 1054, 2019.

[57] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, “Character region awareness for text detection,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2019.

[58] J. Liu, X. Liu, J. Sheng, D. Liang, X. Li, and Q. Liu, “Pyramid mask text detector,” *CoRR*, vol. abs/1903.11800, 2019.

[59] 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 *Proc. Eur. Conf. on Comp. Vision (ECCV)*, 2018, pp. 67–83.

[60] X. Liu, D. Liang, S. Yan, D. Chen, Y. Qiao, and J. Yan, “FOTS: Fast oriented text spotting with a unified network,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2018, pp. 5676–5685.

[61] T. He, Z. Tian, W. Huang, C. Shen, Y. Qiao, and C. Sun, “An end-to-end textspotter with explicit alignment and attention,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2018, pp. 5020–5029.

[62] M. Busta, L. Neumann, and J. Matas, “Deep textspotter: An end-to-end trainable scene text localization and recognition framework,” in *Proc. IEEE Int. Conf. on Comp. Vision*, 2017, pp. 2204–2212.

[63] B. Shi, X. Bai, and C. Yao, “An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 11, pp. 2298–2304, 2016.

[64] B. Shi, X. Wang, P. Lyu, C. Yao, and X. Bai, “Robust scene text recognition with automatic rectification,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2016, pp. 4168–4176.

[65] W. Liu, C. Chen, K.-Y. K. Wong, Z. Su, and J. Han, "STAR-Net: A spatial attention residue network for scene text recognition," in *Proc. Brit. Mach. Vision Conf. (BMVC)*. BMVA Press, September 2016, pp. 43.1–43.13.

[66] F. Borisyuk, A. Gordo, and V. Sivakumar, "Rosetta: Large scale system for text detection and recognition in images," in *Proc. ACM SIGKDD Int. Conf. on Knowledge Discovery & Data Mining*, 2018, pp. 71–79.

[67] J. Baek, G. Kim, J. Lee, S. Park, D. Han, S. Yun, S. J. Oh, and H. Lee, "What is wrong with scene text recognition model comparisons? dataset and model analysis," in *Proc. Int. Conf. on Comp. Vision (ICCV)*, 2019.

[68] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image and vision computing*, vol. 22, no. 10, pp. 761–767, 2004.

[69] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in natural scenes with stroke width transform," in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2010, pp. 2963–2970.

[70] Z. Huang, K. Chen, J. He, X. Bai, D. Karatzas, S. Lu, and C. Jawahar, "ICDAR2019 competition on scanned receipt ocr and information extraction," in *Int. Conf. on Doc. Anal. and Recognit. (ICDAR)*, 2019, pp. 1516–1520. [Online]. Available: https://rrc.cvc.uab.es/?ch=13&com=tasks

[71] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, S. Ghosh, A. Bagdanov, M. Iwamura, J. Matas, L. Neumann, V. R. Chandrasekhar, S. Lu *et al.*, "ICDAR 2015 competition on robust reading," in *Proc. Int. Conf. on Document Anal. and Recognition (ICDAR)*, 2015, pp. 1156–1160. [Online]. Available: https://rrc.cvc.uab.es/?ch=4

[72] C. K. Ch'ng and C. S. Chan, "Total-text: A comprehensive dataset for scene text detection and recognition," in *Proc. IAPR Int. Conf. on Document Anal. and Recognit. (ICDAR)*, vol. 1, 2017, pp. 935–942. [Online]. Available: https://github.com/cs-chan/Total-Text-Dataset

[73] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2016, pp. 779–788.

[74] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. on Comp. Vision*, 2017, pp. 2961–2969.

[75] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[76] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young, "ICDAR 2003 robust reading competitions," in *Seventh Int. Conf. on Document Anal. and Recognition, 2003. Proceedings.*, 2003, pp. 682–687. [Online]. Available: http://www.iapr-tc11.org/mediawiki/index.php/ICDAR_2003_Robust_Reading_Competitions

[77] K. Wang and S. Belongie, “Word spotting in the wild,” in *Proc. Eur. Conf. on Comp. Vision*. Springer, 2010, pp. 591–604. [Online]. Available: http://vision.ucsd.edu/~kai/svt/

[78] A. Shahab, F. Shafait, and A. Dengel, “ICDAR 2011 robust reading competition challenge 2: Reading text in scene images,” in *Proc. Int. Conf. on Doc. Anal. and Recognit.*, 2011, pp. 1491–1496.

[79] A. Mishra, K. Alahari, and C. V. Jawahar, “Scene text recognition using higher order language priors,” in *BMVC*, 2012. [Online]. Available: https://cvit.iiit.ac.in/research/projects/cvit-projects/the-iiit-5k-word-dataset

[80] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, “Detecting texts of arbitrary orientations in natural images,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2012, pp. 1083–1090.

[81] T. Quy Phan, P. Shivakumara, S. Tian, and C. Lim Tan, “Recognizing text with perspective distortion in natural scenes,” in *Proc. IEEE Intl. Conf. on Comput. Vision*, 2013, pp. 569–576. [Online]. Available: http://vision.ucsd.edu/~kai/svt/

[82] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L. G. i Bigorda, S. R. Mestre, J. Mas, D. F. Mota, J. A. Almazan, and L. P. De Las Heras, “ICDAR 2013 robust reading competition,” in *Proc. Int. Conf. on Document Anal. and Recognition*, 2013, pp. 1484–1493. [Online]. Available: https://rrc.cvc.uab.es/?ch=2

[83] A. Risnumawan, P. Shivakumara, C. S. Chan, and C. L. Tan, “A robust arbitrary text detection system for natural scene images,” *Expert Syst. with Appl.*, vol. 41, no. 18, pp. 8027–8048, 2014. [Online]. Available: http://cs-chan.com/downloads_cute80_dataset.html

[84] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft coco: Common objects in context,” in *Proc. Eur. Conf. on Comp. Vision*. Springer, 2014, pp. 740–755.

[85] M. Iwamura, N. Morimoto, K. Tainaka, D. Bazazian, L. Gomez, and D. Karatzas, “ICDAR2017 robust reading challenge on omnidirectional video,” in *Proc. IAPR Int. Conf. on Document Anal. and Recognition (ICDAR)*, vol. 1, 2017, pp. 1448–1453.

[86] L. Yuliang, J. Lianwen, Z. Shuaitao, and Z. Sheng, “Detecting curve text in the wild: New dataset and new solution,” in *arXiv preprint arXiv:1712.02170*, 2017. [Online]. Available: https://github.com/Yuliang-Liu/Curve-Text-Detector

[87] A. Gupta, A. Vedaldi, and A. Zisserman, “Synthetic data for text localisation in natural images,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2016, pp. 2315–2324. [Online]. Available: https://rrc.cvc.uab.es/?ch=2&com=downloads

[88] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, “Synthetic data and artificial neural networks for natural scene text recognition,” *arXiv preprint arXiv:1406.2227*, 2014. [Online]. Available: https://www.robots.ox.ac.uk/~vgg/data/text/

[89] W. Wang, E. Xie, X. Li, W. Hou, T. Lu, G. Yu, and S. Shao, “Shape robust text detection with progressive scale expansion network,” in *Proc. IEEE/CVF Conf. on Comp. Vision and Pattern Recognit.*, 2019, pp. 9336–9345.

[90] Y. Liu, S. Zhang, L. Jin, L. Xie, Y. Wu, and Z. Wang, “Omnidirectional scene text detection with sequential-free box discretization,” *arXiv preprint arXiv:1906.02371*, 2019.

[91] Z. Liu, G. Lin, S. Yang, F. Liu, W. Lin, and W. L. Goh, “Towards robust curve text detection with conditional spatial expansion,” in *Proc. IEEE/CVF Int. Conf. on Comp. Vision* (ICCV), June 2019.

[92] 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 *Proc. IEEE/CVF Int. Conf. on Comp. Vision* (ICCV), June 2019.

[93] C. Luo, L. Jin, and Z. Sun, “Moran: A multi-object rectified attention network for scene text recognition,” *Pattern Recognition*, vol. 90, pp. 109–118, 2019.

[94] F. Zhan and S. Lu, “Esir: End-to-end scene text recognition via iterative image rectification,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2019, pp. 2059–2068.

[95] M. Yang, Y. Guan, M. Liao, X. He, K. Bian, S. Bai, C. Yao, and X. Bai, “Symmetry-constrained rectification network for scene text recognition,” in *Proc. IEEE/CVF Int. Conf. on Comp. Vision* (ICCV), October 2019.

[96] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, “End-to-end object detection with transformers,” *arXiv preprint arXiv:2005.12872*, 2020.

[97] X. Qin, Y. Zhou, D. Wu, Y. Yue, and W. Wang, “FC2RN: A Fully Convolutional Corner Refinement Network for Accurate Multi-Oriented Scene Text Detection,” *arXiv e-prints*.

[98] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[99] D. Soydaner, “Attention mechanism in neural networks: Where it comes and where it goes,” *arXiv preprint arXiv:2204.13154*, 2022.

[100] 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 *Proc. IEEE/CVF Conf. on Comput. Vision and Pattern Recognit.*, 2020, pp. 9809–9818.

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

[102] Z. Raisi, M. A. Naiel, P. Fieguth, S. Wardell, and J. Zelek, "Challenges of deep learning-based text detection in the wild," *Journal of Computational Vision and Imaging Systems*, vol. 6, no. 1, pp. 1–5, 2021.

[103] S. Ferreira, V. Garin, and B. Gosselin, "A text detection technique applied in the framework of a mobile camera-based application," in *Proceedings of the First International Workshop on Camera-based Document Analysis and Recognition (CBDAR)*, 2005, pp. 133–139.

[104] Z. Qi, M. Kimachi, Y. Wu, and T. Aziwa, "Using adaboost to detect and segment characters from natural scenes," in *Proceedings of CBDAR, ICDAR Workshop*, 2005.

[105] S. M. Hanif and L. Prevost, "Text detection in natural scene images using spatial histograms," in *2nd Workshop on Camera Based Document Analysis and Recognition*, 2007, pp. 122–129.

[106] M. Iwamura, T. Tsuji, A. Horimatsu, and K. Kise, "Real-time recognition of camera-captured characters in complex layouts," in *Proc. Third International Workshop on Camera-Based Document Analysis and Recognition (CBDAR2009)*, 2009, pp. 53–60.

[107] L. Neumann and J. Matas, "A method for text localization and recognition in real-world images," in *Proc. Asian Conf. on Comp. Vision*. Springer, 2010, pp. 770–783.

[108] A. Roy Chowdhury, U. Bhattacharya, and S. K. Parui, "Text detection of two major indian scripts in natural scene images," in *International Workshop on Camera-Based Document Analysis and Recognition*. Springer, 2011, pp. 42–57.

[109] R. Nagy, A. Dicker, and K. Meyer-Wegener, "Neocr: A configurable dataset for natural image text recognition," in *International Workshop on Camera-Based Document Analysis and Recognition*. Springer, 2011, pp. 150–163.

[110] C. Merino-Gracia, K. Lenc, and M. Mirmehdi, "A head-mounted device for recognizing text in natural scenes," in *International Workshop on Camera-Based Document Analysis and Recognition*. Springer, 2011, pp. 29–41.

[111] C. Yi and Y. Tian, "Text string detection from natural scenes by structure-based partition and grouping," *IEEE Trans. on Image Process.*, vol. 20, no. 9, pp. 2594–2605, Sep. 2011.

[112] T. Wang, D. J. Wu, A. Coates, and A. Y. Ng, "End-to-end text recognition with convolutional neural networks," in *Proc. Int. Conf. on Pattern Recognit. (ICPR)*, 2012, pp. 3304–3308.

[113] A. Mishra, K. Alahari, and C. Jawahar, "Top-down and bottom-up cues for scene text recognition," in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2012, pp. 2687–2694.

[114] Q. Ye and D. Doermann, “Scene text detection via integrated discrimination of component appearance and consensus,” in *International Workshop on Camera-Based Document Analysis and Recognition*. Springer, 2013, pp. 47–59.

[115] J. Zhang and R. Kasturi, “Sign detection based text localization in mobile device captured scene images,” in *International Workshop on Camera-Based Document Analysis and Recognition*. Springer, 2013, pp. 71–82.

[116] R. Gao, F. Shafait, S. Uchida, and Y. Feng, “A hierarchical visual saliency model for character detection in natural scenes,” in *International Workshop on Camera-Based Document Analysis and Recognition*. Springer, 2013, pp. 18–29.

[117] X.-C. Yin, X. Yin, K. Huang, and H.-W. Hao, “Robust text detection in natural scene images,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 5, pp. 970–983, 2014.

[118] B. Shi, X. Bai, and S. Belongie, “Detecting oriented text in natural images by linking segments,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2017, pp. 2550–2558.

[119] 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. [Online]. Available: https://cocodataset.org/#download

[120] N. Dalal and B. Triggs, “Histograms of Oriented Gradients for Human Detection,” in *Int. Conf. on Comp. Vision & Pattern Recognit. (CVPR)*, vol. 1, Jun. 2005, pp. 886–893.

[121] Y.-F. Pan, X. Hou, and C.-L. Liu, “A hybrid approach to detect and localize texts in natural scene images,” *IEEE Trans. on Image Process.*, vol. 20, no. 3, pp. 800–813, 2010.

[122] S. Tian, Y. Pan, C. Huang, S. Lu, K. Yu, and C. Lim Tan, “Text flow: A unified text detection system in natural scene images,” in *Proc. IEEE Int. Conf. on Comp. Vision*, 2015, pp. 4651–4659.

[123] A. Bosch, A. Zisserman, and X. Munoz, “Image classification using random forests and ferns,” in *Proc. IEEE Int. Conf. on Comp. vision*, 2007, pp. 1–8.

[124] X. Zhao, K.-H. Lin, Y. Fu, Y. Hu, Y. Liu, and T. S. Huang, “Text from corners: a novel approach to detect text and caption in videos,” *IEEE Trans. on Image Process.*, vol. 20, no. 3, pp. 790–799, 2010.

[125] H. Chen, S. S. Tsai, G. Schroth, D. M. Chen, R. Grzeszczuk, and B. Girod, “Robust text detection in natural images with edge-enhanced maximally stable extremal regions,” in *Proc. IEEE Int. Conf. on Image Process.*, 2011, pp. 2609–2612.

[126] X. Yin, Z. Zuo, S. Tian, and C. Liu, “Text detection, tracking and recognition in video: A comprehensive survey,” *IEEE Trans. on Image Process.*, vol. 25, no. 6, pp. 2752–2773, June 2016.

[127] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” in *Adv. in Neural Info. Processing Sys.*, 2012, pp. 1097–1105.

[128] Y. Zhou, Q. Ye, Q. Qiu, and J. Jiao, “Oriented response networks,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2017, pp. 519–528.

[129] W. He, X.-Y. Zhang, F. Yin, and C.-L. Liu, “Deep direct regression for multi-oriented scene text detection,” in *Proc. IEEE Int. Conf. on Comp. Vision*, 2017, pp. 745–753.

[130] P. He, W. Huang, T. He, Q. Zhu, Y. Qiao, and X. Li, “Single shot text detector with regional attention,” in *Proc. IEEE Int. Conf. on Comp. Vision*, 2017, pp. 3047–3055.

[131] H. Hu, C. Zhang, Y. Luo, Y. Wang, J. Han, and E. Ding, “Wordsup: Exploiting word annotations for character based text detection,” in *Proc. IEEE Int. Conf. on Comp. Vision*, 2017, pp. 4940–4949.

[132] Y. Jiang, X. Zhu, X. Wang, S. Yang, W. Li, H. Wang, P. Fu, and Z. Luo, “R2CNN: Rotational region cnn for orientation robust scene text detection,” *arXiv preprint arXiv:1706.09579*, 2017.

[133] M. Liao, Z. Zhu, B. Shi, G.-s. Xia, and X. Bai, “Rotation-sensitive regression for oriented scene text detection,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2018, pp. 5909–5918.

[134] P. Lyu, C. Yao, W. Wu, S. Yan, and X. Bai, “Multi-oriented scene text detection via corner localization and region segmentation,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2018, pp. 7553–7563.

[135] 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*, 2019.

[136] Z. Huang, Z. Zhong, L. Sun, and Q. Huo, “Mask r-cnn with pyramid attention network for scene text detection,” in *Proc. IEEE Winter Conf. on Appls. of Comp. Vision (WACV)*, 2019, pp. 764–772.

[137] E. Xie, Y. Zang, S. Shao, G. Yu, C. Yao, and G. Li, “Scene text detection with supervised pyramid context network,” in *Proc. AAAI Conf. on Artif. Intell.*, vol. 33, 2019, pp. 9038–9045.

[138] C. Zhang, B. Liang, Z. Huang, M. En, J. Han, E. Ding, and X. Ding, “Look more than once: An accurate detector for text of arbitrary shapes,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2019, pp. 10 552–10 561.

[139] H. Bunke and P. S.-p. Wang, *Handbook of character recognition and document image Anal.* World scientific, 1997.

[140] J. Zhou and D. Lopresti, “Extracting text from www images,” in *Proc. Int. Conf. on Document Anal. and Recognition*, vol. 1, 1997, pp. 248–252.

[141] M. Sawaki, H. Murase, and N. Hagita, “Automatic acquisition of context-based images templates for degraded character recognition in scene images,” in *Proc. Int. Conf. on Pattern Recognit. (ICPR)*, vol. 4, 2000, pp. 15–18.

[142] N. Arica and F. T. Yarman-Vural, “An overview of character recognition focused on off-line handwriting,” *IEEE Trans. on Syst. , Man, and Cybernetics, Part C (Appl. and Reviews)*, vol. 31, no. 2, pp. 216–233, 2001.

[143] S. M. Lucas, “ICDAR 2005 text locating competition results,” in *Proc. Int. Conf. on Document Anal. and Recognition (ICDAR'05)*, Aug 2005, pp. 80–84 Vol. 1.

[144] L. Neumann and J. Matas, “Real-time scene text localization and recognition,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2012, pp. 3538–3545.

[145] C.-Y. Lee and S. Osindero, “Recursive recurrent nets with attention modeling for ocr in the wild,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2016, pp. 2231–2239.

[146] J. Wang and X. Hu, “Gated recurrent convolution neural network for ocr,” in *Advances in Neural Inf. Process. Syst.*, 2017, pp. 335–344.

[147] X. Yang, D. He, Z. Zhou, D. Kifer, and C. L. Giles, “Learning to read irregular text with attention mechanisms,” in *Proc. IJCAI*, vol. 1, no. 2, 2017, p. 3.

[148] Z. Cheng, F. Bai, Y. Xu, G. Zheng, S. Pu, and S. Zhou, “Focusing attention: Towards accurate text recognition in natural images,” in *Proc. IEEE Int. Conf. on Comp. Vision*, 2017, pp. 5076–5084.

[149] W. Liu, C. Chen, and K.-Y. K. Wong, “Char-net: A character-aware neural network for distorted scene text recognition,” in *Proc. AAAI Conf. on Artif. Intell.*, 2018.

[150] Z. Cheng, Y. Xu, F. Bai, Y. Niu, S. Pu, and S. Zhou, “Aon: Towards arbitrarily-oriented text recognition,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2018, pp. 5571–5579.

[151] F. Bai, Z. Cheng, Y. Niu, S. Pu, and S. Zhou, “Edit probability for scene text recognition,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2018, pp. 1508–1516.

[152] Y. Liu, Z. Wang, H. Jin, and I. Wassell, “Synthetically supervised feature learning for scene text recognition,” in *Proc. Eur. Conf. on Comp. Vision (ECCV)*, 2018, pp. 435–451.

[153] Z. Xie, Y. Huang, Y. Zhu, L. Jin, Y. Liu, and L. Xie, “Aggregation cross-entropy for sequence recognition,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2019, pp. 6538–6547.

[154] P. Wang, L. Yang, H. Li, Y. Deng, C. Shen, and Y. Zhang, “A simple and robust convolutional-attention network for irregular text recognition,” *ArXiv*, vol. abs/1904.01375, 2019.

[155] K. Negishi, M. Iwamura, S. Omachi, and H. Aso, “Isolated character recognition by searching features in scene images,” in *First International Workshop on Camera-Based Document Analysis and Recognition*, 2005, pp. 140–147.

[156] K. Kuramoto, W. Ohyama, T. Wakabayashi, and F. Kimura, “Accuracy improvement of viewpoint-free scene character recognition by rotation angle estimation,” in *International Workshop on Camera-Based Document Analysis and Recognition*. Springer, 2013, pp. 60–70.

[157] D. G. Lowe, “Distinctive image features from scale-invariant keypoints,” *Int. J. of Comp. Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[158] J. A. Suykens and J. Vandewalle, “Least squares support vector machine classifiers,” *Neural Process. letters*, vol. 9, no. 3, pp. 293–300, 1999.

[159] N. S. Altman, “An introduction to kernel and nearest-neighbor nonparametric regression,” *The Amer. Statistician*, vol. 46, no. 3, pp. 175–185, 1992.

[160] Y. Zhu, C. Yao, and X. Bai, “Scene text detection and recognition: Recent advances and future trends,” *Frontiers of Comp. Science*, vol. 10, no. 1, pp. 19–36, 2016.

[161] J. Almazán, A. Gordo, A. Fornés, and E. Valveny, “Word spotting and recognition with embedded attributes,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 12, pp. 2552–2566, 2014.

[162] Y.-C. Wu, F. Yin, X.-Y. Zhang, L. Liu, and C.-L. Liu, “Scan: Sliding convolutional attention network for scene text recognition,” *arXiv preprint arXiv:1806.00578*, 2018.

[163] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *CoRR*, vol. abs/1409.1556, 2014.

[164] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit. (CVPR)*, pp. 770–778, 2015.

[165] C.-Y. Lee and S. Osindero, “Recursive recurrent nets with attention modeling for ocr in the wild,” *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit. (CVPR)*, pp. 2231–2239, 2016.

[166] P. He, W. Huang, Y. Qiao, C. C. Loy, and X. Tang, “Reading scene text in deep convolutional sequences,” in *Proc. AAAI Conf. on Artif. Intell.*, 2016.

[167] M. Liao, J. Zhang, Z. Wan, F. Xie, J. Liang, P. Lyu, C. Yao, and X. Bai, “Scene text recognition from two-dimensional perspective,” *ArXiv*, vol. abs/1809.06508, 2018.

[168] Z. Wan, F. Xie, Y. Liu, X. Bai, and C. Yao, “2D-CTC for scene text recognition,” 2019.

[169] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks,” in *Proc. 23rd Int. Conf. on Mach. learning*, 2006, pp. 369–376.

[170] F. Yin, Y.-C. Wu, X.-Y. Zhang, and C.-L. Liu, “Scene text recognition with sliding convolutional character models,” *arXiv preprint arXiv:1709.01727*, 2017.

[171] B. Su and S. Lu, “Accurate scene text recognition based on recurrent neural network,” in *Asian Conference on Computer Vision*. Springer, 2014, pp. 35–48.

[172] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *arXiv preprint arXiv:1409.0473*, 2014.

[173] Z. Wojna, A. N. Gorban, D.-S. Lee, K. Murphy, Q. Yu, Y. Li, and J. Ibarz, “Attention-based extraction of structured information from street view imagery,” in *Int. Conf. on Doc. Anal. and Recognit. (ICDAR)*, vol. 1, 2017, pp. 844–850.

[174] Y. Deng, A. Kanervisto, and A. M. Rush, “What you get is what you see: A visual markup decompiler,” *arXiv preprint arXiv:1609.04938*, vol. 10, pp. 32–37, 2016.

[175] X. Yang, D. He, Z. Zhou, D. Kifer, and C. L. Giles, “Learning to read irregular text with attention mechanisms.” in *IJCAI*, vol. 1, no. 2, 2017, p. 3.

[176] H. Li, P. Wang, C. Shen, and G. Zhang, “Show, attend and read: A simple and strong baseline for irregular text recognition,” in *Proc. AAAI Conf. on Artificial Intel.*, vol. 33, 2019, pp. 8610–8617.

[177] Q. Wang, W. Jia, X. He, Y. Lu, M. Blumenstein, and Y. Huang, “Faclstm: Convlstm with focused attention for scene text recognition,” *arXiv preprint arXiv:1904.09405*, 2019.

[178] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, “Show, attend and tell: Neural image caption generation with visual attention,” in *Proc. Int. Conf. on Machine Learning*, 2015, pp. 2048–2057.

[179] F. Wang, M. Jiang, C. Qian, S. Yang, C. Li, H. Zhang, X. Wang, and X. Tang, “Residual attention network for image classification,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit. (CVPR)*, July 2017, pp. 6450–6458.

[180] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, “Convolutional lstm network: A machine learning approach for precipitation nowcasting,” in *Advances in neural information Process. systems*, 2015, pp. 802–810.

[181] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, “Detecting texts of arbitrary orientations in natural images,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2012, pp. 1083–1090. [Online]. Available: http://www.iapr-tc11.org/mediawiki/index.php/MSRA_Text_Detection_500_Database_(MSRA-TD500)

[182] S. M. Lucas, A. Panaretos, L. Sosa, A. Tang, S. Wong, and R. Young, “ICDAR 2003 robust reading competitions,” in *Proc. Int. Conf. on Document Anal. and Recognition, 2003. Proceedings.*, Aug 2003, pp. 682–687.

[183] A. Gupta, A. Vedaldi, and A. Zisserman, “Synthetic data for text localisation in natural images,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2016, last retrieved March 11, 2020. [Online]. Available: https://www.robots.ox.ac.uk/~vgg/data/scenetext/

[184] Y. Liu and L. Jin, “Deep matching prior network: Toward tighter multi-oriented text detection,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2017, pp. 1962–1969.

[185] X. Liu, H.-F. Yu, I. Dhillon, and C.-J. Hsieh, “Learning to encode position for transformer with continuous dynamical model,” in *Int. Conf. on Machine Learning*, 2020, pp. 6327–6335.

[186] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018.

[187] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, “Language models are unsupervised multitask learners,” *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.

[188] L. Dong, S. Xu, and B. Xu, “Speech-transformer: a no-recurrence sequence-to-sequence model for speech recognition,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2018, pp. 5884–5888.

[189] N. Parmar, A. Vaswani, J. Uszkoreit, Ł. Kaiser, N. Shazeer, A. Ku, and D. Tran, “Image transformer,” *arXiv preprint arXiv:1802.05751*, 2018.

[190] R. Girdhar, J. Carreira, C. Doersch, and A. Zisserman, “Video action transformer network,” in *Proc. IEEE Conf. on Computer Vision and Pattern Recognit.*, 2019, pp. 244–253.

[191] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, “Self-attention generative adversarial networks,” *arXiv preprint arXiv:1805.08318*, 2018.

[192] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[193] Y. Tay, M. Dehghani, D. Bahri, and D. Metzler, “Efficient transformers: A survey,” *arXiv preprint arXiv:2009.06732*, 2020.

[194] H. W. Kuhn, “The hungarian method for the assignment problem,” *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.

[195] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, “Generalized intersection over union,” June 2019.

[196] N. Piasco, D. Sidibé, C. Demonceaux, and V. Gouet-Brunet, “A survey on visual-based localization: On the benefit of heterogeneous data,” *Pattern Recognition*, vol. 74, pp. 90–109, 2018.

[197] L. Deng, Y. Gong, X. Lu, Y. Lin, Z. Ma, and M. Xie, “STELA: A real-time scene text detector with learned anchor,” *IEEE Access*, vol. 7, pp. 153 400–153 407, 2019.

[198] X. Wang, S. Zheng, C. Zhang, R. Li, and L. Gui, “R-YOLO: A real-time text detector for natural scenes with arbitrary rotation,” *Sensors*, vol. 21, no. 3, p. 888, 2021.

[199] R. Endo, Y. Kawai, H. Sumiyoshi, and M. Sano, “Scene-text-detection method robust against orientation and discontiguous components of characters,” in *Proc. IEEE Conf. on Comput. Vision and Pattern Recognit. Workshops*, 2017, pp. 1–9.

[200] W. Chan, C. Saharia, G. Hinton, M. Norouzi, and N. Jaitly, “Imputer: Sequence modelling via imputation and dynamic programming,” *arXiv preprint arXiv:2002.08926*, 2020.

[201] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, “Deformable detr: Deformable transformers for end-to-end object detection,” *arXiv preprint arXiv:2010.04159*, 2020.

[202] J. Lee, S. Park, J. Baek, S. Joon Oh, S. Kim, and H. Lee, “On recognizing texts of arbitrary shapes with 2D self-attention,” in *IEEE CVPR*, 2020, pp. 546–547.

[203] Z. Raisi, M. Naiel, P. Fieguth, S. Wardell, and J. Zelek, “2d positional embedding-based transformer for scene text recognition,” *Journal of Computational Vision and Imaging Systems*, vol. 6, no. 1, p. 1–4, 2021.

[204] S. Khan, M. Naseer, M. Hayat, S. W. Zamir, F. S. Khan, and M. Shah, “Transformers in vision: A survey,” *arXiv preprint arXiv:2101.01169*, 2021.

[205] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” in *Proc. IEEE Conf. on Comput. Vision and Pattern Recognit.*, 2014, pp. 580–587.

[206] R. Girshick, “Fast R-CNN,” in *Proc. IEEE Int. Conf. on Comput. Vision*, 2015, pp. 1440–1448.

[207] M. Van Kreveld, O. Schwarzkopf, M. de Berg, and M. Overmars, *Computational geometry algorithms and applications*. Springer, 2000.

[208] W. H. Beyer, *Standard mathematical tables and formulae*. CRC press, 1991.

[209] A. M. Andrew, “Another efficient algorithm for convex hulls in two dimensions,” *Information Processing Letters*, vol. 9, no. 5, pp. 216–219, 1979.

[210] A. Laaksonen, “Competitive programmer’s handbook,” *Preprint*, 2017.

[211] I. Loshchilov and F. Hutter, “Decoupled weight decay regularization,” *arXiv preprint arXiv:1711.05101*, 2017.

[212] F. Zhan, S. Lu, and C. Xue, “Verisimilar image synthesis for accurate detection and recognition of texts in scenes,” in *Proc. of the European Conf. on Comput. Vision (ECCV)*, 2018, pp. 249–266.

[213] S. Long and C. Yao, “Unrealtext: Synthesizing realistic scene text images from the unreal world,” *arXiv preprint arXiv:2003.10608*, 2020.

[214] G. Bradski, “The OpenCV Library,” *Dr. Dobb’s Journal of Software Tools*, 2000.

[215] S. Long, X. He, and C. Yao, “Scene text detection and recognition: The deep learning era,” *International Journal of Computer Vision*, vol. 129, no. 1, pp. 161–184, 2021.

[216] Y. Bi and Z. Hu, “Disentangled contour learning for quadrilateral text detection,” in *Proc. IEEE/CVF Winter Conf. on Appl. of Comput. Vision*, 2021, pp. 909–918.

[217] C. Zhang, B. Liang, Z. Huang, M. En, J. Han, E. Ding, and X. Ding, “Look more than once: An accurate detector for text of arbitrary shapes,” in *Proc. IEEE Conf. on Comp. Vision and Pattern Recognit.*, 2019, pp. 10 552–10 561.

[218] 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 Trans. on Image Process.*, vol. 28, no. 11, pp. 5566–5579, 2019.

[219] F. Wang, Y. Chen, F. Wu, and X. Li, “Textray: Contour-based geometric modeling for arbitrary-shaped scene text detection,” in *Proc. ACM International Conference on Multimedia*, 2020, pp. 111–119.

[220] Y. Zhu, J. Chen, L. Liang, Z. Kuang, L. Jin, and W. Zhang, “Fourier contour embedding for arbitrary-shaped text detection,” in *Proc. IEEE/CVF Confon. Comput. Vision and Pattern Recognit.*, 2021, pp. 3123–3131.

[221] Y. Liu, C. Shen, L. Jin, T. He, P. Chen, C. Liu, and H. Chen, “Abcnet v2: Adaptive bezier-curve network for real-time end-to-end text spotting,” *arXiv preprint arXiv:2105.03620*, 2021.

[222] M. Liao, Z. Wan, C. Yao, K. Chen, and X. Bai, “Real-time scene text detection with differentiable binarization,” in *Proc. AAAI Conf. on Artif. Intell.*, vol. 34, no. 07, 2020, pp. 11 474–11 481.

[223] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.

[224] K. Han, Y. Wang, H. Chen, X. Chen, J. Guo, Z. Liu, Y. Tang, A. Xiao, C. Xu, Y. Xu *et al.*, “A survey on visual transformer,” *arXiv preprint arXiv:2012.12556*, 2020.

[225] C. Joshi, “Transformers are graph neural networks,” *The Gradient*, 2020.

[226] S. Tuli, I. Dasgupta, E. Grant, and T. L. Griffiths, “Are convolutional neural networks or transformers more like human vision?” *arXiv preprint arXiv:2105.07197*, 2021.

[227] J. Tang, Z. Yang, Y. Wang, Q. Zheng, Y. Xu, and X. Bai, “Seglink++: Detecting dense and arbitrary-shaped scene text by instance-aware component grouping,” *Pattern Recognit.*, vol. 96, p. 106954, 2019.

[228] S.-X. Zhang, X. Zhu, J.-B. Hou, C. Liu, C. Yang, H. Wang, and X.-C. Yin, “Deep relational reasoning graph network for arbitrary shape text detection,” in *Proc. IEEE/CVF Confon. Comput. Vision and Pattern Recognit.*, 2020, pp. 9699–9708.

[229] Y. Sun, Z. Ni, C.-K. Chng, Y. Liu, C. Luo, C. C. Ng, J. Han, E. Ding, J. Liu, D. Karatzas *et al.*, “ICDAR 2019 competition on large-scale street view text with partial labeling–RRC-LSVT,” *arXiv preprint arXiv:1909.07741*, 2019.

[230] G. G. Lorentz, *Bernstein polynomials*. American Mathematical Soc., 2013.

[231] 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 *Proc. IEEE/CVF Confon. Comput. Vision and Pattern Recognit.*, 2019, pp. 9076–9085.

[232] 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 *Proc. IEEE/CVF Confon. Comput. Vision and Pattern Recognit.*, 2020, pp. 11 753–11 762.

[233] K. Choromanski, V. Likhosherstov, D. Dohan, X. Song, A. Gane, T. Sarlos, P. Hawkins, J. Davis, A. Mohiuddin, L. Kaiser *et al.*, “Rethinking attention with performers,” *arXiv preprint arXiv:2009.14794*, 2020.

[234] B. Wang, L. Shang, C. Lioma, X. Jiang, H. Yang, Q. Liu, and J. G. Simonsen, “On position embeddings in {bert},” in *International Conference on Learning Representations*, 2021.

[235] P. Shaw, J. Uszkoreit, and A. Vaswani, “Self-attention with relative position representations,” *arXiv preprint arXiv:1803.02155*, 2018.

[236] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[237] L. Kang, P. Riba, M. Rusiñol, A. Fornés, and M. Villegas, "Pay attention to what you read: Non-recurrent handwritten text-line recognition," *arXiv preprint arXiv:2005.13044*, 2020.

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

[239] Y. Li, Y. Tian, J. Tian, and F. Zhou, "An efficient method for dpm code localization based on depthwise separable convolution," *IEEE Access*, 2019.

[240] Y. Sun, C. Zhang, Z. Huang, J. Liu, J. Han, and E. Ding, "Textnet: Irregular text reading from images with an end-to-end trainable network," in *Asian Conf. on Comput. Vision*, 2018, pp. 83–99.

[241] A. Singh, G. Pang, M. Toh, J. Huang, W. Galuba, and T. Hassner, "Textocr: Towards large-scale end-to-end reasoning for arbitrary-shaped scene text," in *Proc. IEEE/CVF Confon. Comput. Vision and Pattern Recognit.*, 2021, pp. 8802–8812.

[242] S. Chaudhari, G. Polatkan, R. Ramanath, and V. Mithal, "An attentive survey of attention models," *arXiv preprint arXiv:1904.02874*, 2019.

[243] Z. Raisi, M. A. Naiel, G. Younes, S. Wardell, and J. Zelek, "2lspe: 2d learnable sinusoidal positional encoding using transformer for scene text recognition," in *Proc. Conf. on Robots and Vision (CRV)*, 2021, pp. 119–126.

[244] W. Wang, E. Xie, X. Li, D.-P. Fan, K. Song, D. Liang, T. Lu, P. Luo, and L. Shao, "Pyramid vision transformer: A versatile backbone for dense prediction without convolutions," *arXiv preprint arXiv:2102.12122*, 2021.

[245] P. Sun, R. Zhang, Y. Jiang, T. Kong, C. Xu, W. Zhan, M. Tomizuka, L. Li, Z. Yuan, C. Wang *et al.*, "Sparse r-cnn: End-to-end object detection with learnable proposals," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14 454–14 463.

[246] Z. Chen, Y. Zhu, C. Zhao, G. Hu, W. Zeng, J. Wang, and M. Tang, "DPT: Deformable patch-based transformer for visual recognition," in *Proceedings of the 29th ACM International Conference on Multimedia*, 2021, pp. 2899–2907.

[247] Y. Liu, Y. Zhang, Y. Wang, F. Hou, J. Yuan, J. Tian, Y. Zhang, Z. Shi, J. Fan, and Z. He, "A survey of visual transformers," *arXiv preprint arXiv:2111.06091*, 2021.

[248] J. He, S. M. Erfani, X. Ma, J. Bailey, Y. Chi, and X.-S. Hua, "Alpha-iou: A family of power intersection over union losses for bounding box regression," in *Proc. Neural Information Processing Systems*, 2021.

[249] A. Geovanna Soares, B. Leite Dantas Bezerra, and E. Baptista Lima, "How far deep learning systems for text detection and recognition in natural scenes are affected by occlusion?" in *International Conference on Document Analysis and Recognition*. Springer, 2021, pp. 198–212.

[250] Z. Raisi and J. S. Zelek, "End-to-end scene text spotting at character level," in *Proc. Annual Conference on Vision and Intelligent Systems* CVIS, 2021.

[251] L. Chang, J. Sun, M. Suwa, H. Takebe, Y. He, and S. Naoi, "Occluded text restoration and recognition," in *Proc. IAPR Intl. Workshop on Doc. Anal. Sys.*, 2010, pp. 151–158.

[252] T. T. Pham, H.-I. Choi, and G.-Y. Kim, "A matching strategy to recognize occluded number," in *Proc. Korean Society of Computer Information Conference*, 2011, pp. 55–58.

[253] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," in *European conference on computer vision*, 2006, pp. 404–417.

[254] B. M. Lake, R. Salakhutdinov, and J. B. Tenenbaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.

[255] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman, "Building machines that learn and think like people," *Behavioral and brain sciences*, vol. 40, 2017.

[256] A. Stone, H. Wang, M. Stark, Y. Liu, D. Scott Phoenix, and D. George, "Teaching compositionality to cnns," in *Proc. IEEE Conf. on Comput. Vision and Pattern Recognit.*, 2017, pp. 5058–5067.

[257] A. Kortylewski, Q. Liu, H. Wang, Z. Zhang, and A. Yuille, "Combining compositional models and deep networks for robust object classification under occlusion," in *The IEEE Winter Conference on Applications of Computer Vision*, 2020, pp. 1333–1341.

[258] A. Kortylewski, Q. Liu, A. Wang, Y. Sun, and A. Yuille, "Compositional convolutional neural networks: A robust and interpretable model for object recognition under occlusion," *arXiv preprint arXiv:2006.15538*, 2020.

[259] A. Wang, Y. Sun, A. Kortylewski, and A. L. Yuille, "Robust object detection under occlusion with context-aware compositionalnets," in *Proc. IEEE/CVF Conf. on Comput. Vision and Pattern Recognit.*, 2020, pp. 12 645–12 654.

[260] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[261] J. Lu, D. Batra, D. Parikh, and S. Lee, "Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks," *arXiv preprint arXiv:1908.02265*, 2019.

[262] Y. Li, S. Xie, X. Chen, P. Dollar, K. He, and R. Girshick, “Benchmarking detection transfer learning with vision transformers,” *arXiv preprint arXiv:2111.11429*, 2021.

[263] S. Fang, H. Xie, Y. Wang, Z. Mao, and Y. Zhang, “Read like humans: Autonomous, bidirectional and iterative language modeling for scene text recognition,” in *Proc. IEEE/CVF Conf. on Comput. Vision and Pattern Recognit.*, 2021, pp. 7098–7107.

[264] Y. Wang, H. Xie, S. Fang, J. Wang, S. Zhu, and Y. Zhang, “From two to one: A new scene text recognizer with visual language modeling network,” in *Proc. IEEE/CVF Intl. Conf. on Comput. Vision*, 2021, pp. 14 194–14 203.

[265] B. Na, Y. Kim, and S. Park, “Multi-modal text recognition networks: Interactive enhancements between visual and semantic features,” *arXiv preprint arXiv:2111.15263*, 2021.

[266] S. Qin, A. Bissacco, M. Raptis, Y. Fujii, and Y. Xiao, “Towards unconstrained end-to-end text spotting,” in *Proc. IEEE/CVF Intl. Conf. on Computer Vision*, 2019, pp. 4704–4714.

[267] M. Liao, G. Pang, J. Huang, T. Hassner, and X. Bai, “Mask textspotter v3: Segmentation proposal network for robust scene text spotting,” in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*, 2020, pp. 706–722.

[268] R. Atienza, “Vision transformer for fast and efficient scene text recognition,” in *Document Analysis and Recognition – ICDAR 2021.* Springer International Publishing, 2021, pp. 319–334.

[269] H. Li, P. Wang, and C. Shen, “Towards end-to-end text spotting with convolutional recurrent neural networks,” *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 5248–5256, 2017.

[270] L. Qiao, Y. Chen, Z. Cheng, Y. Xu, Y. Niu, S. Pu, and F. Wu, “Mango: A mask attention guided one-stage scene text spotter,” *arXiv preprint arXiv:2012.04350*, 2020.

[271] Y. Baek, S. Shin, J. Baek, S. Park, J. Lee, D. Nam, and H. Lee, “Character region attention for text spotting,” *ArXiv*, vol. abs/2007.09629, 2020.

[272] Y. Kittenplon, I. Lavi, S. Fogel, Y. Bar, R. Manmatha, and P. Perona, “Towards weakly-supervised text spotting using a multi-task transformer,” *arXiv preprint arXiv:2202.05508*, 2022.

[273] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020.

[274] S. Cao, P. Xu, and D. A. Clifton, “How to understand masked autoencoders,” *arXiv preprint arXiv: Arxiv-2202.03670*, 2022.

[275] Z. Raisi, G. Younes, and J. Zelek, “Arbitrary shape text detection using transformers,” in *Proc. Intl. Conf. on Pattern Recognit.* (ICPR), 2022, p. Under Review.

[276] Z. Raisi, M. A. Naiel, G. Younes, S. Wardell, and J. S. Zelek, “Transformer-based text detection in the wild,” in *Proc.* IEEE/CVF *Conf. on Comp. Vision and Pattern Recognit. Workshops* (CVPRW), 2021, pp. 3156–3165.

[277] Y. Wu and K. He, “Group normalization,” *arXiv preprint arXiv: Arxiv-1803.08494*, 2018.

[278] D. Hendrycks and K. Gimpel, “Gaussian error linear units (gelus),” *arXiv preprint arXiv: Arxiv-1606.08415*, 2016.

[279] A. El-Nouby, H. Touvron, M. Caron, P. Bojanowski, M. Douze, A. Joulin, I. Laptev, N. Neverova, G. Synnaeve, J. Verbeek, and H. Jegou, “Xcit: Cross-covariance image transformers,” *arXiv preprint arXiv: Arxiv-2106.09681*, 2021.

[280] R. Atienza, “Data augmentation for scene text recognition,” in *Proc. IEEE/CVF Intl. Conf. on Comp. Vision (ICCV) Workshops*, October 2021, pp. 1561–1570.

[281] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[282] X. Chen, K. Kundu, Y. Zhu, A. G. Berneshawi, H. Ma, S. Fidler, and R. Urtasun, “3d object proposals for accurate object class detection,” in *Advances in Neural Information Processing Systems*, 2015, pp. 424–432.

[283] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. Denton, S. K. S. Ghasemipour, B. K. Ayan, S. S. Mahdavi, R. G. Lopes *et al.*, “Photorealistic text-to-image diffusion models with deep language understanding,” *arXiv preprint arXiv:2205.11487*, 2022.

[284] M. Zhao, W. Feng, F. Yin, X.-Y. Zhang, and C.-L. Liu, “Weakly-supervised arbitrary-shaped text detection with expectation-maximization algorithm,” *arXiv preprint arXiv:2012.00424*, 2020.

[285] A. Aberdam, R. Ganz, S. Mazor, and R. Litman, “Multimodal semi-supervised learning for text recognition,” *arXiv preprint arXiv:2205.03873*, 2022.

[286] X. Chu and Y. Wang, “Itervm: Iterative vision modeling module for scene text recognition,” *arXiv preprint arXiv:2204.02630*, 2022.

[287] P. Lyu, C. Zhang, S. Liu, M. Qiao, Y. Xu, L. Wu, K. Yao, J. Han, E. Ding, and J. Wang, “Maskocr: Text recognition with masked encoder-decoder pretraining,” *arXiv preprint arXiv:2206.00311*, 2022.

[288] Y. LeCun and A. Canziani, “Lectures of deep learning course,” 2018. [Online]. Available: https://cds.nyu.edu/deep-learning/

[289] L. Weng, “Attention? attention!” *lilianweng.github.io*, 2018. [Online]. Available: https://lilianweng.github.io/posts/2018-06-24-attention/

[290] V. A. Stuart, “Biomedical knowledge discovery in networks through language/graphical models and machine learning,” Persagen Consulting (Persagen.com), Tech. Rep., Aug. 2018. [Online]. Available: https://persagen.com/resources/biokdd-review.html

APPENDICES

Appendix A

Loss Functions And Detailed Architectures

A.1 Losses

A.1.1 Smooth-L1 Loss Function

This loss is leveraged for the tasks with bounding box regression. The smooth egression loss (L_{reg})is defined as:

$$L_{reg} = \sum_{i \in S} \text{smooth}_{L1}(p_i, p^*) \tag{A.1}$$

in which,

$$\text{Smooth}_{L1}(x) = \begin{cases} 0.5(\sigma x)^2 \, if |x| < 1/\sigma^2 \\ |x| - 0.5/\sigma^2 \, otherwise \end{cases} \tag{A.2}$$

where x represents the error between the predicted bounding boxes (p) and the ground-truth (p^*) The Smooth-ln loss is also an another continuous function of the above piece-wise equation *et* *al.* [184], defined as follow:

$$\text{smooth}_{ln}(x) = (|x| + 1)\ln(|x| + 1) - |x| \tag{A.3}$$

A.2 Detailed Architecture

A.2.1 Transformer for text detection

Figure A.1 shows architecture of object detection using transformer.

A.3 Self-attention

Let $x_i \in \mathbb{R}^n$ denotes an n-dimensional vector. So for a set of t input x's, we have [288]:

$$\{x_i\}_{i=1}^{t} = \{x_1, x_2, ..., x_t\}, \qquad X = [x_1, x_2, ..., x_t] \in \mathbb{R}^{n \times t} \tag{A.4}$$

where $X \in \mathbb{R}^{n \times t}$ represents the set as a matrix form. With self-attention, the hidden representation h is a linear combination of the inputs:

$$h = \alpha_1 x_1 + \alpha_2 x_2 +, ..., \alpha_t x_t \in \mathbb{R}^n \tag{A.5}$$

Using the matrix representation described above, we can write the hidden layer as the matrix product:

$$h = Xa \tag{A.6}$$

where $a \in R^n$ is a column vector with components α_i. With *hard-attention*, we impose the following constraint on the alphas: $||a||_0 = 1$. This means α is a one-hot vector. Therefore, all

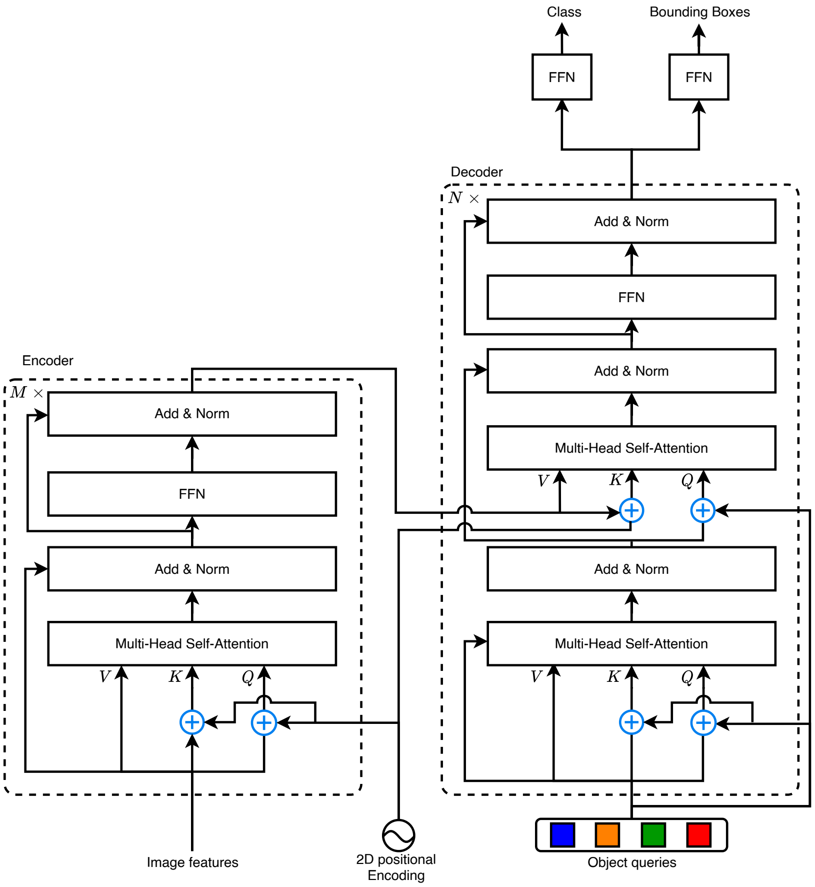


Figure A.1: Architecture of object detection using transformer. The above architecture is reproduced from [96].

but one of the coefficients in the linear combination of the inputs equals zero, and the hidden representation reduces to the input x_i corresponding to the element $\alpha_i = 1$. With *soft-attention*, we impose that $||a||_1 = 1$. The hidden representations is a linear combination of the inputs where the coefficients sum up to 1. In Eq. A.6, a is defined as follows:

$$a = \mathtt{Softmax}_\beta(X^T x) \in \mathbb{R}^t, \quad \beta = \frac{1}{\sqrt{d}} \tag{A.7}$$

since large similarities cause `Softmax` to saturate and give vanishing gradients, so for a d-dimension vector, β is used to control the temperature constant of it. For set of x, we have set of a which we can call it matrix $A \in \mathbb{R}^{t \times t}$. Similarly, by having a set of a we have a set of h as H from Eq. A.6. Generally, the matrix format of Eq. A.6 can be defined as follows:

$$H = XA \in \mathbb{R}^{n \times t} \tag{A.8}$$

A.3.1 Queries, Keys and Values

The input set x is projected by three matrices, q, k, and v, which are referred to as the *queries*, *keys* and *values*, respectively as [288]:

$$q = W_q x \in \mathbb{R}^{d'}, \quad k = W_k x \in \mathbb{R}^{d'}, \quad v = W_v x \in \mathbb{R}^{d''} \tag{A.9}$$

In order to compare the query against all possible keys, q and k must have the same dimensionality, i.e. $q, k \in \mathbb{R}^{d'}$. However, v can be of any dimension. But for simplicity we can set $d' = d'' \triangleq d$. for set of x, we have a set of q, k, and v as follows:

$$\{x_i\}_{i=1}^{t} \rightsquigarrow \{q_i\}_{i=1}^{t}, \{k_i\}_{i=1}^{t}, \{v_i\}_{i=1}^{t} \rightsquigarrow Q, K, V \in \mathbb{R}^{d \times t} \tag{A.10}$$

By considering the above equation, we can define a and the hidden layer (h) as follows:

$$a = \texttt{Softmax}_{\beta}(K^T q) = \texttt{Softmax}(\frac{K^T q}{\sqrt{d}}) \in \mathbb{R}^t \qquad h = Va \in \mathbb{R}^d \tag{A.11}$$

where one query (q) is compared with all keys (K). β in the above equation is used for controlling the temperature constant of `Softmax`, which for d-dimensions can be defined as follows [288]:

$$\beta = \frac{1}{\sqrt{d}} \tag{A.12}$$

large similarities will cause `Softmax` to saturate and give vanishing gradients. For example, $a.b = |a||b|cos(\theta)$ and suppose a and b are constant vectors of dimension d then $|a| = (\sum_i a_i^2) = a\sqrt{d}$. Finally, for a set of q and a the final set of hidden layer (H) can be defined as follows [288]:

$$\{q_i\}_{i=1}^{t} \rightsquigarrow \{a_i\}_{i=1}^{t} \rightsquigarrow A \in \mathbb{R}^{t\times t}, \qquad H = VA \in \mathbb{R}^{d\times t} \tag{A.13}$$

A.3.2 Multi-Head Self Attention

Multi-Head Self Attention mechanism MHA learns an alignment in which each element in the sequence learns to gather from other elements in the sequence [193, 288–290]. By considering h as heads we have a vector in $\mathbb{R}^{3hd}$ as follows:

$$\begin{bmatrix} q \\ k \\ v \end{bmatrix} = \begin{bmatrix} W_q \\ W_k \\ W_v \end{bmatrix} x \in \mathbb{R}^{3d} \tag{A.14}$$

And for h-head we can extend the above equation as follows:

$$\begin{bmatrix} q^1 \\ q^2 \\ \vdots \\ q^h \end{bmatrix} = \begin{bmatrix} W_q^1 \\ W_q^2 \\ \vdots \\ W_q^h \end{bmatrix} x, \quad \begin{bmatrix} k^1 \\ k^2 \\ \vdots \\ k^h \end{bmatrix} = \begin{bmatrix} W_k^1 \\ W_k^2 \\ \vdots \\ W_k^h \end{bmatrix} x, \quad \begin{bmatrix} v^1 \\ v^2 \\ \vdots \\ v^h \end{bmatrix} = \begin{bmatrix} W_v^1 \\ W_v^2 \\ \vdots \\ W_v^h \end{bmatrix} x, \quad \rightsquigarrow \begin{bmatrix} q^h \\ k^h \\ v^h \end{bmatrix} = \begin{bmatrix} W_q^h \\ W_k^h \\ W_v^h \end{bmatrix} x \in \mathbb{R}^{d \times hd}$$

so by using $W_h \in \mathbb{R}^{d \times hd}$ to get back to $\mathbb{R}^d$.

A.4 One-Dimensional (1D) Positional Encoding

The positional encoding (PE) is first utilized in natural language processing in [1]. To explain, let $X \in \mathbb{R}^{n \times t}$ be an example of the embedding input, where n and t denote the sequence length and embedding size, respectively. The PE block in Figure 3.1 encodes the position of X and outputs $PE + X$, and it can be defined as follows:

$$\begin{aligned} \text{PE}(p, 2i) &= \sin\left(\frac{p}{10000^{2i/t}}\right), \\ \text{PE}(p, 2i+1) &= \cos\left(\frac{p}{10000^{2i/t}}\right). \end{aligned} \tag{A.15}$$

where p in the above equation show input sequence's order, in which $(p = 0, \ldots, n-1)$. The i illustrates the position along the embedding vector dimension and it varies as follows: $(i = 0, \ldots, \lfloor (t-1)/2 \rfloor)$.