University of Zurich[UZH]

Scene Text Extraction for Retrieval of Visual Multimedia

Thesis July 11, 2021

Alexander Theus
of Wädenswil ZH, Switzerland

Student-ID: 18-737-171
alexander.theus@uzh.ch

Advisor: **Dr. Luca Rossetto**

Prof. Abraham Bernstein, PhD
Institut für Informatik
Universität Zürich
http://www.ifi.uzh.ch/ddis

Acknowledgements

First, I want to express my deepest gratitude to my supervisor, Dr. Luca Rossetto, for his continuous support and advice-giving throughout this thesis. A special thanks also goes to Dr. Silvan Keller and Dr. Ralph Gasser, who, together with Dr. Luca Rossetto, were so kind to review my code, and allow me to contribute to the fascinating multimedia information retrieval stack, vitrivr. I would also like to thank Prof. Abraham Bernstein for the opportunity to work on this thesis.

Finally, I want to thank my family and friends for their encouragement and support throughout this thesis and my studies in general.

Zusammenfassung

Mit der Vergrößerung von multimedialen Sammlungen wurde die Suche nach dem darin enthaltenen Wissen immer aufwändiger und eine manuelle Annotation unzumutbar. Als Konsequenz wurde vitrivr entwickelt, das ein inhaltsbasiertes Retrieval über Methoden wie Query-by-Sketch, Query-by-Example und viele mehr ermöglicht. Ein noch unerforschter Teil des in visuellen Multimedia-Inhalten enthaltenen Wissens ist der Szenentext. Textinformationen, die in visuellem Multimedia eingebettet sind, liefern hochrangige semantische Informationen über den Inhalt und den Kontext der Medien und können für ein besseres Retrieval genutzt werden.

Zu diesem Zweck wurden in dieser Arbeit bestehende Methoden zur Szenentextextraktion in Standbildern untersucht und evaluiert. Darüber hinaus wurde ein neuartiger Szenentextextraktor für Videos namens HyText entwickelt, der in meiner Evaluierung sehr hohe Leistung erzielte. Die Neuartigkeit der vorgeschlagenen Methode liegt in der Hybridisierung von Tracking-by-Detection und Partikelfilterung, um eine verbesserte Inferenzzeit zu ermöglichen. Die vorgeschlagene Methode ist in vitrivr implementiert, um die Extraktion und Abfrage von Szenentext zu ermöglichen.

Abstract

The expansion of multimedia collections has made the quest for accessing the knowledge contained within them ever more onerous, and has rendered prior annotation unfeasible. As a consequence, vitrivr was developed which enables content-based retrieval via methods such as Query-by-Sketch, Query-by-Example, and many more. A yet unexplored piece of knowledge contained in visual multimedia is scene text. Textual information embedded in visual multimedia provides high-level semantic information about the content and context of the media, and can be leveraged for superior retrieval.

For this purpose, this thesis explored and evaluated existing methods for scene text extraction in still images. Furthermore, a novel scene text extractor for videos called HyText is developed, which achieved state-of-the-art performance in my evaluation. The novelty of the proposed method relies on hybridizing tracking-by-detection and particle filtering to allow for enhanced inference time. The proposed method is implemented in vitrivr to enable the extraction and retrieval of scene text.

Contents

1

Introduction

1.1 Motivation

As multimedia collections grow larger in terms of size, heterogeneity, and variety of media types, the quest for accessing the knowledge contained within them becomes more onerous. The traditional approach of manually annotating media objects, and retrieving them at a later point based on this metadata has various deficiencies. Firstly, the sheer size of media collections and their progressive growth renders prior annotation unfeasible. Secondly, textual descriptions tend to be subjective due to language, culture, expertise, and personal experience. Lastly, temporal evolution is strenuous to describe, e.g. in video-based media, in a way that enables others to retrieve the desired object later [Gasser et al., 2019].

Figure 1.1: Example of the difficulty of retrieving shops of a particular kind. In this case, tea and coffee shops are visually heterogeneous within their category, and visually indifferentiable from other shops. Images taken from [Tripadvisor, 2021], [Foursquare-City-Guide, 2021], [Natura-Coffee and Tea, 2021], and [Galerie, 2021]

.

For this reason, a content-based multimedia information retrieval stack called "vitrivr" was introduced. Instead of solely relying on metadata, vitrivr allows for content-based retrieval such as with "Query-by-Example" (QbE) and "Query-by-Sketch" (QbS). As the names suggest, QbE is performed by providing an image similar to the desired object, whereas QbS allows the user to sketch a similar image [Gasser et al., 2019]. While these features are certainly useful, they are intrinsically inapt for certain retrieval scenarios. A prime example of that is the differentiation of different kinds of shops. A tea shop, and

a coffee shop, are meaningfully different, but differentiating them without referring to storefront signs, is very challenging even to humans themselves (see Figure 1.1). In fact, research has shown that shop classifiers end up learning to interpret textual content since it is the most meaningful way to distinguish them [Movshovitz-Attias et al., 2015]. Thus, neither QbE nor QbS could sufficiently retrieve the desired shop as most shops would be clustered together due to their inter-categorical homogeneity. However, providing access to the textual content of the storefront signs would allow for appropriate retrieval.

This type of naturally occurring text is termed "scene text". Due to the abundance of textual content in natural scenes, and especially in urban settings [Lin et al., 2014], the task of scene text extraction (STE) has received a lot of interest from the computer vision community, and has been propelled by advances in deep learning. While STE in still images remains a problem, several approaches have emerged which show a sufficiently high performance to be useful for general purpose applications. STE in videos, however, is still a virgin field, even though it is of equal importance.

1.2 Goals

Due to the insufficiency of QbE and QbS to handle certain retrieval scenarios, and the advances in STE performance, the goal of this thesis is to extend the existing vitrivr with the ability to extract scene text from visual multimedia and to later retrieve said objects by referring to the extracted textual content.

Since there has already been a lot of research on STE in still images, the first goal will be to explore and evaluate existing STE methods to find the most suitable one for vitrivr. The evaluation of said methods will be done with suitable metrics and encompass both accuracy and inference time.

STE in videos is a very young field, and to my knowledge no published code exists for said purpose. Thus, the goal will be to develop a novel scene text extractor for video-based material. A literature review about existing methods will be done for the sake of inspiration, and the developed method will be evaluated according to accuracy and inference time.

Last but not least, the method for STE in still images and videos will be implemented in vitrivr, and evaluated to ensure its performance.

1.3 Outline

In order to meaningfully arrive at the aforementioned goals, the thesis is structured as follows: Section 2 provides the reader with the necessary background in text in visual multimedia, STE, and vitrivr itself to clarify terminology and the task at hand. Section 3 will introduce existing research for scene text retrieval. Subsequently, Section 4 and 5 deal with the exploration and evaluation of the sub-components which make up the STE task. The most promising sub-components are then matched in Section 6 to create end-to-end STE pipelines for still images, which are then evaluated to arrive at optimal

matches. In Section 7, existing methods for STE in videos are investigated, and my novel video-based STE method called HyText is introduced and evaluated. Afterwards, the STE method will be implemented in vitrivr in Section 8. At last, the insights and contributions of this thesis are summarized in Section 9, and proposals for future work are made as they relate to the STE performance, invariance to textual prominence, and visuo-textual interplay.

2

Background

In order to soundly progress with this thesis, a few concepts have to be cleared up. Firstly, I will introduce the different kinds of text that can appear in visual multimedia, and subsequently state which type this thesis will focus on. Secondly, the task of STE and its sub-components will be explained in Section 2.2 to provide context for upcoming chapters. Lastly, the multimedia retrieval stack, vitrivr, its functions, and in particular the extension that this paper introduces will be elucidated.

2.1 Text in Visual Multimedia

(a) Graphic text (b) Scene text

Figure 2.1: Difference between graphic text and scene text. Images taken from [Nguyen et al., 2014] and [Karatzas et al., 2015]

Text can appear differently in visual multimedia, and typically be described by three characteristics: Form, language, and generation. Text can either be in handwritten or printed form. Text in handwritten form is usually more strenuous to recognize due to the diversity of handwriting styles. Depending on the language of the text, text in visual multimedia can comprise varying characters such as Chinese, Latin, or Arabic. This introduces a wide diversity of text characteristics as the reading order, and text categories may vary substantially in different languages. Lastly, text can be categorized into “graphic text” and “scene text”. Graphic text is text that is digitally added as an

overlay to images and videos as can be seen in Figure 2.1a. Scene text, on the other hand, refers to text on objects captured in their natural environment [Chen et al., 2020]. A prime example of that would be a picture in which an advertisement appears with text on it as shown in Figure 2.1b. Scene text is particularly hard to detect and recognize as it can appear on any surface, and may be hardly distinguishable from its background [Chen et al., 2020].

With these categories in mind, this thesis will focus on the extraction of printed Latin scene text from visual multimedia.

2.2 Scene Text Extraction

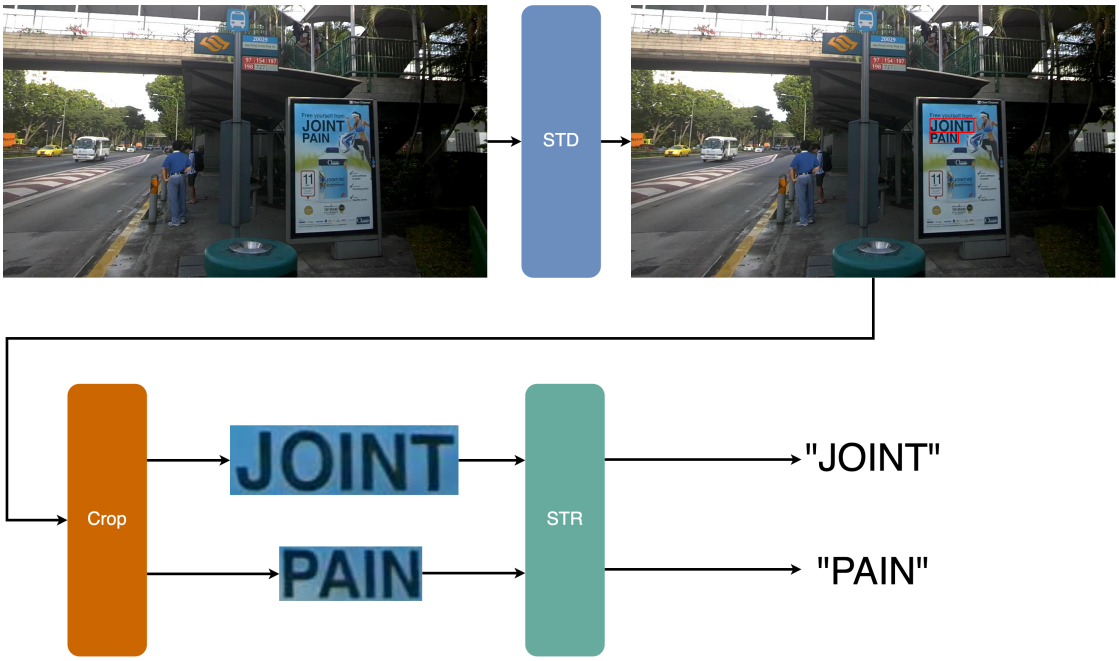


Figure 2.2: Architecture of a typical STE pipeline. Image taken from [Karatzas et al., 2015].

The task of STE is to take an image as input and output the text that appears in it. In Figure 2.2, an input image is fed into the pipeline and two strings - “JOINT” and “PAIN” - exit it since these two text instances appear in the input image. In order to accomplish this task, a typical pipeline is comprised of three consecutive steps. At first, text instances have to be detected and localized. This step is commonly referred to as “scene text detection” (STD), and has the job to take an input image, and output bounding boxes enveloping individual text instances. This is illustrated in Figure 2.2 where two detected text instances are enveloped by red rectangular boxes. After the STD task, the bounding boxes have to be cropped. Namely, the bounding boxes containing

the text have to be separated from their superfluous surroundings. As can be seen in Figure 2.2, after the cropping the original image containing the two text instances is transformed into two individual images which contain their respective text. The last module of the pipeline is typically called "scene text recognition" (STR), and has the purpose of taking a cropped image containing text, and outputting said text as a string.

Apart from content-based retrieval, STE has a plethora of additional applications enabled by automatic scene text understanding. It can contribute to intelligent transportation by constructing automatic geocoding systems [Xie et al., 2018], which not only alleviates traveling, but also enables users to overcome language barriers, e.g. by automatically detecting road signs and translating them into the native language of the driver [Chen et al., 2004]. Moreover, STE can aid the visually impaired. According to the World Health Organization [WHO, 2021], at least 2.2 billion people live with a visual impairment or blindness. STE technology can improve their life, e.g. by text-to-speech devices to help understand menus, books, signs, newspapers, and so on [Gómez et al., 2018].

2.3 Vitrivr

The progressive increase in multimedia collections has necessitated new ways in storing, organizing, and searching data. Especially for large multimedia collections, providing a simple one-size-fits all approach with just one query option is insufficient due to varying user intentions in different applications. Despite the inaptness, most multimedia search platforms focus on tag-based keyword searches [Rossetto et al., 2016].

Multimedia Information Retrieval (MMIR), is an area of research which intends to improve current multimedia search solutions by extracting semantic information from large collections of media [Rossetto et al., 2016]. The vitrivr stack is a MMIR system with a particular focus on similarity search such as QbS and QbE across different media types [Gasser et al., 2019].

2.3.1 Functionality

Vitrivr can be parted into three subdivisions: the feature database ADAM$_{pro}$, the retrieval engine Cineast, and the browser-based vitrivr frontend. The whole architecture is illustrated in Figure 2.3. ADAM$_{pro}$ is a storage engine which is able to persistently store and retrieve multimedia object on a large scale. More specifically, it allows for fast and scalable k-Nearest-Neighbour search in high-dimensional vector spaces which is crucial for content-based multimedia retrieval [Gasser et al., 2019].

Cineast is a modular retrieval engine implemented in Java and serves as the core of the vitrivr architecture. It supports two types of workflows: the offline "ingest workflow" and the online "retrieval workflow". The offline workflow consists of decoding and segmenting the multimedia file into meaningfully distinguished scenes. These derived segments are then fed to one or more feature modules. The online workflow parses the

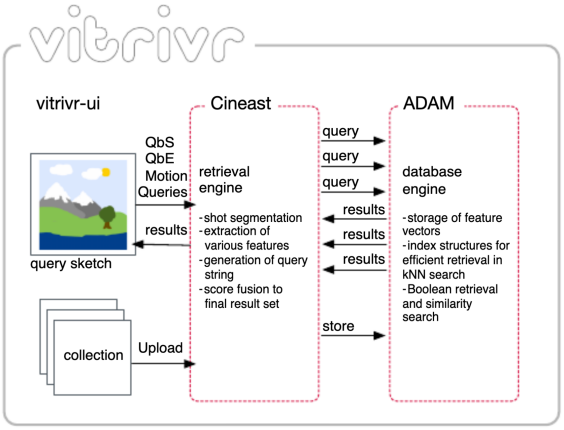


Figure 2.3: The vitrivr stack architecture. Figure taken from [Rossetto et al., 2016].

user provided queries to retrieve relevant documents. The feature modules are responsible for extracting relevant features, generating feature vectors, and to persist the feature vectors in the storage layer (offline), or to perform a look-up (online). For example, when QbS or QbE is used, the provided reference document (e.g. an image or a sketch) is utilized to extract relevant features using the aforementioned feature modules. The feature modules thus represent the core functionality of Cineast both for the offline and online workflow [Gasser et al., 2019].

The vitrivr frontend is a user-interface which assists the user in forming queries. These queries consist of two building blocks: a "query component" and a "query term". A query component contains multiple query terms that can be activated. At least one query term has to be active in a component and there cannot be multiple active query terms of the same type within a component. Individual query terms can differ in terms of the reference document used, e.g. the image query term allows users to upload an image or to make a sketch, whereas the audio query term provides the user with the ability to upload short query terms [Gasser et al., 2019].

2.3.2 Scene Text Retrieval Extension

As already mentioned in Section 1.1, retrieval of visual multimedia via QbS and QbE is inapt for certain retrieval scenarios. For this reason, vitrivr will be extended with the ability to search for visual multimedia by looking for text that appears in it. Namely, the user should be able to provide one or more strings and be provided with images or video segments in which the provided text appears. Since the frontend already provides for the possibility to search for scene text, this thesis will extend Cineast with a feature module called "OCRSearch", which is able to take an image, or a segment of a video, and extract the text that appears in it. Since the retrieval itself (online), does not require the extraction of features such as with QbS or QbE, the extraction will happen offline only. Thus, inference time for STE is not a primary concern. However, it cannot be neglected as vitrivr is used to extract features from videos that can span thousands

of hours [Gasser et al., 2019]. The text retrieval will be supported by Apache Lucene [Apache, 2021]. Since the recognition of scene text instances is a particularly onerous task, small errors can be expected. Consequently, fuzzy searches should be enabled by default to retrieve extracted text that closely resembles the query string. However, the user should also be able to specify that the retrieved text should completely match the query string by providing quotation marks around it. Since fuzzy searches are enabled by default, the method evaluation should not only take into account complete matches, but also relative matches. Last but not least, as mentioned in Section 2.1, the method should focus on the extraction of printed Latin scene text.

3

Related Work

Research which deals with STE as it relates to retrieval is rather scarce. Nevertheless, over the years a few of such methods have been introduced. The two most prominent approaches in this field of research will briefly be illuminated.

Gómez et al. [Gómez et al., 2018] developed a real-time STE method that detects and recognizes text in a single shot. The architecture of the method is based on YOLO, and is recast as PHOC (Pyramidal Histogram Of Characters) predictor, which enables performing detection and recognition at the same time. Another novel method introduced by Wang et al. [Wang et al., 2021] performs scene text retrieval without the need for STR. Namely, they introduce a new deep learning framework for text retrieval that combines the stages of text detection and cross-modal similarity learning. The network is optimized by two tasks: a text detection branch which proposes bounding boxes of candidate text regions, and a similarity learning branch to capture the cross-modal similarity between a query text and a bounding box. Thus, this method is specifically optimized for scene text retrieval and can achieve superior inference times by avoiding the STR task.

4

Scene Text Detection

The objective of scene text detection (STD) is to take an image and to predict the position of text instances by a bounding box (see Figure 4.1). The shape of such a bounding box is usually a rectangle, oriented rectangle or a quadrilateral which can be described by the following parameters respectively: (x, y, w, h), (x, y, w, h, θ) and (x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4). Akin to the development of other computer vision tasks, STD historically relied on hand-crafted features. Since around 2015, deep learning-based methods have become mainstream, in which features specific to text are learned rather than manually obtained.

This chapter aims to give the reader a holistic overview of the STD task, and to subsequently evaluate and select the most appropriate detectors for the feature module. In Section 4.1, STD methods relying on hand-crafted features will briefly be explored for the sake of context. Then, in Section 4.2 state of the art methods will be investigated within their respective category. Lastly, the most promising approaches will be evaluated in Section 4.3.

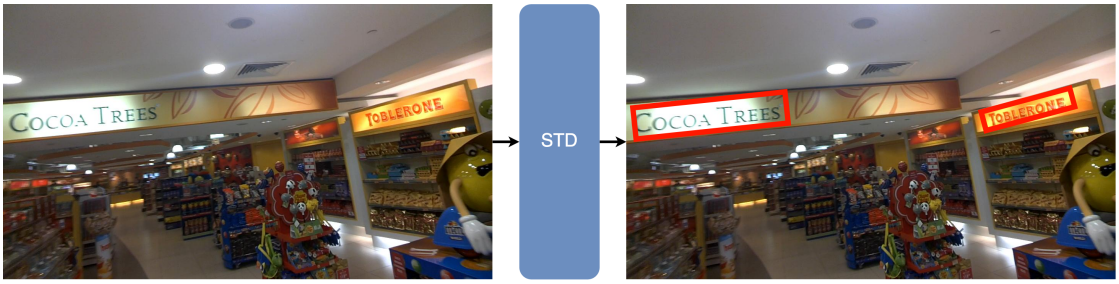


Figure 4.1: High-level illustration of the STD task with rotated rectangle bounding boxes (x, y, w, h, θ). Image taken from [Karatzas et al., 2015].

4.1 Hand-Crafted Feature Extraction Era

STD methods relying on hand-crafted features aim to find and develop hand-crafted low-level features which uniquely describe text. Most of these methods can be classified into connected component (CC) based and sliding window (SW) based methods [Pan et al., 2011].

SW-based methods detect text by moving a multi-scale window through all possible regions in the image. From these regions a feature vector is extracted which is then fed into a classifier for text/non-text discrimination. Last but not least, in order to create text blocks, neighboring regions of text are merged [Pan et al., 2011]. A concrete example of such a procedure is the one proposed by Wang et al. [Wang et al., 2011], who slide a multi-scale window through the image in which they extract features for each one to later classify the existence or absence of a character via a RandomFerns classifier. The features extracted consist of randomly chosen thresholds on randomly chosen entries in the HOG descriptor.

On the other hand, CC-based methods extract candidate components and then filter out non-text ones using classifiers or heuristic rules [Pan et al., 2011]. One of the most prominent methods of this category is stroke-width transform (SWT) [Epshtein et al., 2010], which takes advantage of the fact that text tends to have a consistent stroke width. Thus, it computes the stroke width variance per component and subsequently prunes components with a variance above the threshold defined [Epshtein et al., 2010].

SW and CC methods have their respective strengths and weaknesses. SW methods are relatively slow and sensitive to text orientation. In contrast, CC methods cannot effectively segment text components without prior knowledge of text scale and position. Some hybrid methods have been proposed which intend to exploit the strengths of both methods while avoiding its weaknesses such as the one proposed by Pan et al. [Pan et al., 2011].

4.2 Deep Learning Era

Contrary to hand-crafted feature methods, deep learning-based methods automatically learn and extract discriminative features by training a model and are used in a wide range of tasks such as speech recognition, object detection, semantic segmentation and other pattern recognition problems [Lin et al., 2020]. They have achieved great success relative to their hand-crafted counterparts and have thus become mainstream within the STD domain.

Deep learning-based methods can be classified into segmentation-based and object detection-based methods [Lin et al., 2020]. STD methods from these categories will be examined thoroughly and conclusions will be drawn from both an intra- and inter-categorical perspective. A hierarchal overview of these categories is presented in Figure 4.2.

4.2.1 Segmentation-Based Methods

The aim of image segmentation is to classify an image on a per-pixel basis, determine the category of each point, and divide the image correspondingly. Specific to STD, segmentation-based detectors generally first compute a segmentation from a fully-convolutional network (FCN), extract the text blocks and lastly generate bounding boxes via complex post-processing [Lin et al., 2020].

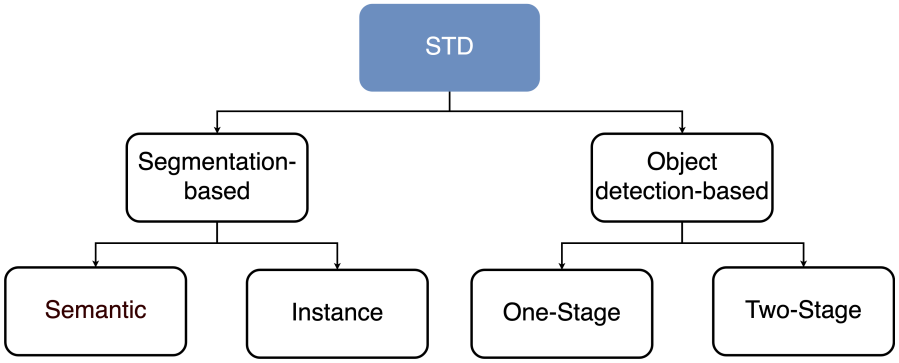


Figure 4.2: Hierarchical overview of deep learning-based STD categories.

Zhang et al. [Zhang et al., 2016] were one of the early adopters of image segmentation for STD. They propose a Text-Block FCN which first generates a salient map and then segments it into candidate text boxes. Then the text boxes are divided into character components for the sake of estimating the orientation. Using the cues of the character component's orientation and the candidate text boxes, bounding boxes (oriented rectangles) are generated and later filtered out to avoid false positives. Later, a method called SegLink [Shi et al., 2017] aimed to detect long and oriented text with segments and links. They argue that general object detection methods are not well suited for scene text since, in contrast to other objects, text has a very large aspect ratio and is multi-oriented. In order to avoid this, they propose to detect smaller components such as individual characters, which do not suffer from extreme aspect ratios or multi-orientation, and then connect them by links to create the final bounding boxes. An inherent flaw to semantic segmentation is that it may fail to separate two text instances when they are relatively close. In order to address this, Wang et al. [Wang et al., 2019] introduce a novel instance segmentation-based method called PSENet. This method incorporates the advantage of traditional segmentation-based method of being able to locate text with arbitrary shapes, while still being able to distinguish between two close text instances. To do this, they generate multiple segmentation areas for a text instance, which are similar in terms of shape but differ in scale. Then they produce the final detection result with a Breadth-First-Search (BFS) based algorithm which starts with the segmentation of the smallest scale and progressively includes pixels of segmentation areas of larger scales. Recently, Yang et al. [Yang et al., 2020] introduced an instance segmentation-based method that simultaneously generates prototype masks and per-instance mask coefficients. Furthermore, to refine the detection accuracy and avoid over-fitting, they apply self-distillation to train the model and ensure generalizability. Another inherent disadvantage of segmentation-based methods is the intensive post-processing binarization required to convert the probability map produced by the segmentation network into detected text instances. For example, the previously introduced PSENet [Wang et al., 2019] uses the progressive scale expansion algorithm to enhance detection accuracy. Liao et al. [Liao et al., 2020] intend to overcome this flaw

by integrating the binarization into the segmentation network by a module they call Differentiable Binarization (DB). Since the standard binarization procedure is not differentiable, they propose an approximate function for binarization that is fully differentiable and thus optimizable. Baek et al. [Baek et al., 2019] detect three difficulties with current STD methods which they intend to solve. Namely, the STD task is obscured by extreme aspect ratios, shape distortion, and variety of character sizes. These factors, however, are not inherent to STD in general, but specific to word-/line level detection. Character-level detection, on the other hand, mitigates these difficulties substantially. Thus, they introduce a character-level STD method named CRAFT, in which a FCN is trained to predict character regions and the affinity between characters. Although character-based methods such as CRAFT do well for text with arbitrary shapes, they are often time-consuming due to the large number of candidate characters generated. Word-based methods are more efficient and simpler, however, they do not do well with text of arbitrary shapes [Ye et al., 2020]. Ye et al. [Ye et al., 2020] intend to take advantage of both character- and word-based methods by extracting both character-, word- and global features to obtain richer fused features.

Segmentation-based methods for STD have become mainstream, largely due to their insensitivity to font variation, noise, blur, and orientation. However, as previously discussed, they rely on complex post-processing to deal with cases in which text instances are close to each other. In order to overcome this, promising approaches such as instance segmentation and integrated binarization have been proposed.

4.2.2 Object Detection-Based Methods

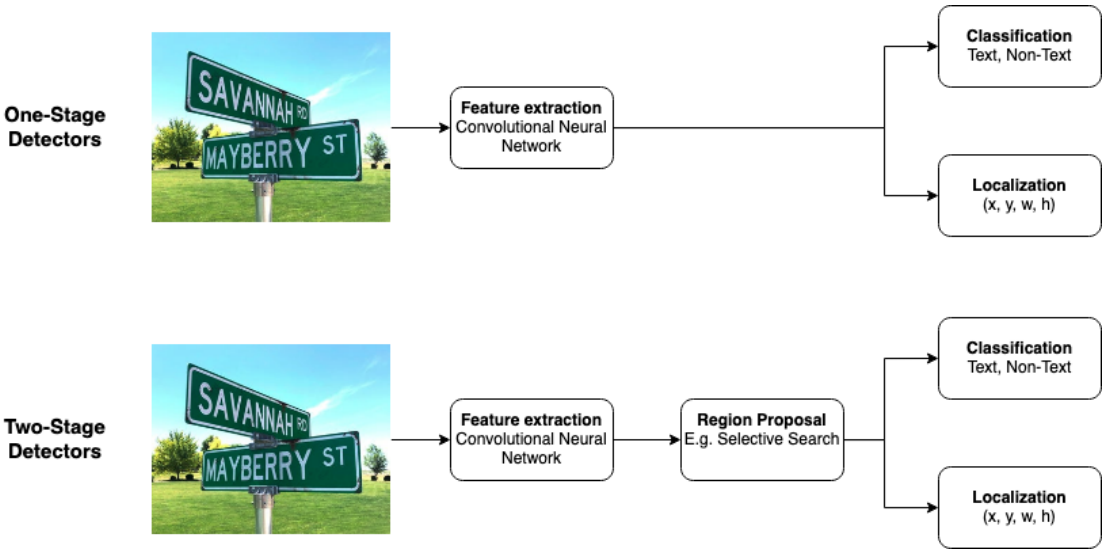


Figure 4.3: Abstract illustration of one- and two-stage object detection architectures. Image taken from [Etsy, 2021].

In contrast to segmentation-based methods, object detection-based methods predict candidate bounding boxes directly by regarding text as objects [Sun et al., 2018]. The majority of these methods can further be classified into one-stage and two-stage architectures. In contrast to the one-stage model, the two-stage model includes a region proposal step, which proposes regions of interest, and at last, the final predictions are generated on those regions (see Figure 4.3). The most prominent two-stage model is Faster R-CNN [Ren et al., 2015]. On the other hand, the one-stage model generates the final predictions on the image directly, and various methods have been created within this domain such as the single-shot detector [Liu et al., 2016a] and YOLO [Redmon et al., 2016].

Jiang et al. [Jiang et al., 2017] introduce a method called R^2CNN, which is based on the two-stage Faster R-CNN model and optimized for detecting multi-oriented scene text. Since text in scenes is usually small relative to general objects, the anchor boxes in Faster R-CNN are modified to be smaller to take advantage of this text characteristic. Furthermore, Faster R-CNN does ROI-Pooling on the feature map with a pooled size of 7x7. However, since text usually has a significantly larger width than height, two different pool sizes are used: 3x11 (for horizontal-leaning text) and 11x3 (for vertical leaning-text). The final output bounding box is a rotated rectangle. Later, Liu and Jin [Liu and Jin, 2017] remark that prior techniques devised to detect multi-oriented text rely on rotated rectangles such as the previously introduced R^2CNN. However, scene text cannot always be satisfactorily bounded by a rotated rectangle due to character distortion. This may cause unnecessary overlap between rectangular boxes, redundant information and may result in marginal text not being able to be localized accurately. In order to address this, they introduce quadrilateral sliding windows of multiple shapes to more accurately represent text. Similarly, Liao et al. [Liao et al., 2018] optimize the one-stage single-shot detector (SSD) for scene text. For one, they propose to represent oriented text with quadrilaterals or oriented rectangles. In their prior paper, they used "long" anchor boxes to better suit scene text which tend to have a large aspect ratio [Liao et al., 2016]. However, since long bounding boxes are not suitable for oriented text, they now also introduce anchor boxes of inverse aspect ratios. It is important to note that the previously introduced methods rely on hand-crafted anchor boxes and are thus not learned. Instead of choosing priors manually, Busta et al. [Busta et al., 2017] run k-means clustering on the training set to acquire their anchor boxes. With the requirement that each ground truth box shall have an intersection-over-union (IoU) of at least 60% with one anchor box, they came up with k=14 boxes. Since one-stage models directly compute their final predictions on the unmodified anchor boxes, they rely heavily on how densely the anchor boxes cover the target box. Thus there has been a trend to use multiple anchor boxes of various scales, aspect ratios, and orientations to achieve higher coverage. For instance, the previously introduced TextBoxes++ method used seven specific aspect ratios. DPMNet added several oriented rectangles (six regular and six inclined). DeepTextSpotter uses fourteen. Deng et al [Deng et al., 2019] intend to decrease the number of anchors whilst maintaining similar performance by utilizing learned anchors which are generated through a regression operation to replace the original anchor box in the final prediction. Although one-stage detectors are very fast relative to two-stage models, they usually have lower localization and object recogni-

tion accuracy [Jiao et al., 2019]. Recently, Wang et al. [Wang et al., 2020] introduced a method which similar to TextBoxes++ and DPMNet can detect text with quadrilateral boundaries. However, their proposed method is a two-stage method and thus does not suffer from sub-optimal performance. Namely, they introduce a method called QRPN for generating quadrilateral region proposals based on a new quadrilateral regression algorithm. In the second stage of the architecture, a novel weighted RoI pooling module with learned weight masks to pool the features is introduced and at last, the proposals are classified and the shapes refined with the quadrilateral regression algorithm.

Object detection-based methods for STD have become popular due to their high accuracy and recall rates. Much work has been done to adapt general object detectors such as SSD [Liu et al., 2016a], YOLO [Redmon et al., 2016] and Faster R-CNN [Ren et al., 2015] for scene text. However, due to the multi-oriented nature of scene text, many methods design anchors of various aspect ratios, scales and orientations to better capture text, which renders object detection-based methods complicated and inefficient.

4.3 Evaluation

EAST [Zhou et al., 2017], PixelLink [Deng et al., 2018], CRAFT [Baek et al., 2019], and the detection module of EasyOCR [JaidedAI, 2021] will be evaluated according to two datasets that carry distinct characteristics. The novel tightness-aware intersection-over-union metric [Liu et al., 2019] will be used to calculate the correspondence between ground truth bounding boxes and the detected ones. Furthermore, the inference time is captured to put the computed accuracy in perspective.

4.3.1 Methods

Name	Type	Focus	Training set	Year	Code
EAST	Semantic	Speed	ICDAR2015	2017	[argman, 2021]
PixelLink	Instance	Dense text	ICDAR2015	2018	[ZJULearning, 2021]
CRAFT	Semantic	Perspective distortion	ICDAR2015	2019	[ClovaAI, 2021a]
EasyOCR (CRAFT)	Semantic	Perspective distortion	N/A	2019	[JaidedAI, 2021]

Table 4.1: Information on different STD methods

As previously mentioned, EAST [Zhou et al., 2017], PixelLink [Deng et al., 2018] and CRAFT [Baek et al., 2019] will be evaluated. Furthermore, the detection module in EasyOCR will be evaluated additionally to see how the aforementioned state-of-the-art methods perform relative to an already established and widely used end-to-end pipeline.

EAST is a segmentation-based method that aims to be faster than its competitors by eliminating superfluous intermediate steps. Even though inference time is not a primary concern, it still plays an important role in the decision-making process. If EAST's simple

architecture does not come at the cost of accuracy, it could be a good candidate for the final pipeline.

A salient characteristic in natural images is the density at which text occurs. For semantic segmentation-based methods, the distinction between closely adjacent text is very difficult and sometimes even impossible. This is especially problematic since STD is a peculiar type of object detection that is done for the sake of recognizing text. Thus, the presence of multiple text instances within a bounding box can obstruct the recognition process (see Section 4.3.2 for more details). In order to handle this, Deng et al. [Deng et al., 2018] propose an instance segmentation-based method called PixelLink which is superior in distinguishing dense text.

CRAFT aims to overcome the difficulty of multi-oriented text and perspective distortion by predicting character regions and the affinity between characters. CRAFT is also used in the mainstream STE pipeline EasyOCR. While EasyOCR uses the same method, the official CRAFT method will still be evaluated separately due to the fact that EasyOCR does not reveal how their detection module was trained and if the original code was altered.

With this group of methods a broad spectrum of characteristics and problems of STD are addressed. Namely, methods which deal with speed, dense text, and extreme distortions. All the methods are trained on ICDAR2015 [Karatzas et al., 2015]. ICDAR2015 is specifically chosen due to its abundance of irregular and blurred scene text, which simulate natural images and especially frames in videos.

4.3.2 Evaluation Metric

(a) Incomplete detection

(b) Complete detection

Figure 4.4: Comparison of complete and incomplete STD. Image taken from ICDAR2015 [Karatzas et al., 2015]. Red: ground-truth. Yellow: detection.

(a) Incompact detection

(b) Compact detection

Figure 4.5: Comparison of compact and incompact STD. Image taken from ICDAR2015 [Karatzas et al., 2015]. Red: ground-truth. Yellow: detection.

Mainstream metrics to evaluate the performance of STD methods have been adopted from the object detection PASCAL VOC metric [Everingham et al., 2014], which was

Figure	IoU	TIoU	Recognition
Figure 4.4a: Incomplete detection	0.8	0.64	"JOIN"
Figure 4.4b: Complete detection	0.8	0.8	"JOINT"
Figure 4.5a: Incompact detection	0.8	0.72	"JOINT-"
Figure 4.5b: Compact detection	0.8	0.8	"JOINT"

Table 4.2: Comparison on how the completeness and compactness of a detection influence IoU and TIoU respectively.

designed for generic object detection. However, STD is a special kind of detection task since it is performed specifically to recognize text in a later step. Thus, generic object detection metrics cannot be directly applied to STD. Namely, generic object detection metrics are completeness- and compactness-invariant.

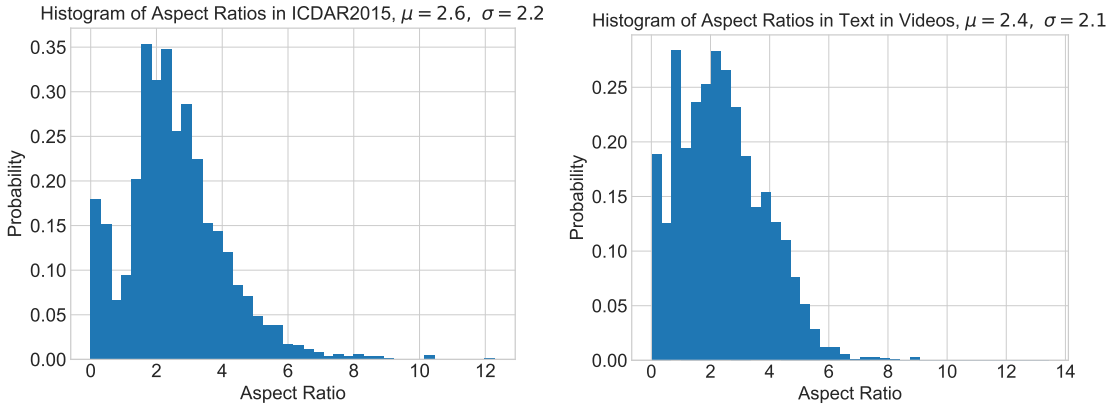
In Figure 4.4a an incomplete detection is shown. In other words, the detection does not encompass the entirety of the ground-truth bounding box and in this example even misses a whole letter. In Figure 4.4b a complete detection is shown in which the detection envelops the whole ground-truth box. While both detections are not ideal since they do not perfectly match the ground-truth, the complete detection is highly preferable since it does not cut-off a letter and thus is more suitable to recognize the contained text. However, traditional IoU methods cannot distinguish this case and thus lead to the same IoU of 0.8 (see Table 4.2).

As previously mentioned, IoU-based methods can also not take compactness into account. Namely, if a detection envelops not just one, but two ground-truth bounding boxes, then that can negatively influence the subsequent recognition step. In Figure 4.5b a detection is shown in which the detection envelops only one ground-truth. In Figure 4.5a, however, the detection envelops two ground-truth boxes. Thus, the detection in Figure 4.5b should be superior. However, IoU cannot distinguish between the two cases and gives both an IoU value of 0.8 (see Table 4.2). Since the STD methods specified in Section 4.3.1 are almost all semantic-based ones, which have difficulties distinguishing dense text, compactness is an especially important factor to capture.

In order to extend object detection metrics with the particularities of STD, Liu et al [Liu et al., 2019] develop the Tightness-aware Intersect-over-Union (TIoU) metric that can take both completeness and compactness into account. As can be seen in Table 4.2, the scores for the incomplete and incompact detections are lower than their counterparts. Furthermore, they use the TIoU value directly instead of binarizing the score with a threshold as mainstream IoU metrics do. For the purpose of evaluating the methods defined in Sections 4.3.1, the TIoU metric will be used to calculate recall, precision, and f-measure. Furthermore, the inference time will be captured to put the results in perspective.

Name	Irregular Text	Blur	Still Image	Amount	Avg. Aspect Ratio
ICDAR2015	✓	✓	✓	500	2.6
Text in Videos	✓	✓	×	538	2.4

Table 4.3: Overview of datasets used for STD evaluation.



(a) ICDAR2015

(b) Text in Videos

Figure 4.6: Aspect ratio histograms of ICDAR2015 and Text in Videos.

4.3.3 Datasets

Two distinct datasets called ICDAR2015 [Karatzas et al., 2015] and Text in Videos [Iwamura et al., 2021] were chosen to evaluate the STD methods. ICDAR2015 is a dataset based on still images. In contrast to other mainstream detection datasets, ICDAR2015 is characterized by its particular presence of blurred and low-resolution text. Text in Videos is, as the name already suggests, a dataset based on video footage. It contains 15 videos which all in all make up almost 10'000 frames. In order to adapt the dataset to something more manageable for frame-by-frame detection, random frames from each video were selected to create a new and more manageable dataset which is comprised of 538 images. These two datasets were specifically chosen to simulate the detection of scene text in visual multimedia. Moreover, the groundtruth of both datasets are quadrilateral bounding boxes of the form (x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4), which is particularly important in this case since the output of the chosen STD methods are also quadrilateral bounding boxes. If the ground-truth were rectangular boxes or rotated rectangular boxes, the increased tightness of quadrilateral bounding boxes could not be captured. Last but not least, the dataset Text in Videos also serves the function of capturing the generalizability of the STD methods since all of them were trained on ICDAR2015.

A salient characteristic of scene text which has made the STD task challenging is the presence of varying aspect ratios that arise from diverse word lengths and multi-orientation of text. In order to shed a light on how this is manifested in ICDAR2015

and Text in Videos, the aspect ratios of ground-truth bounding boxes were calculated and visualized in Figure 4.6. The aspect ratio is defined as

$$AspectRatio = \frac{width}{height} \tag{4.1}$$

where *width* and *height* are

$$Width = Distance(leftCenter, rightCenter) \tag{4.2}$$

$$Height = Distance(topCenter, bottomCenter) \tag{4.3}$$

Thus, a high aspect ratio may be indicative of particularly long words, whereas low aspect ratios either indicate short words or multi-oriented text. ICDAR2015 and Text in Videos do not differ significantly in either average aspect ratio or its standard deviation. Both, however, have rather low aspect ratios than what may be expected from horizontal text. Thus, this may be indicative of the presence of text instances that are not horizontally aligned and thereby hard to detect.

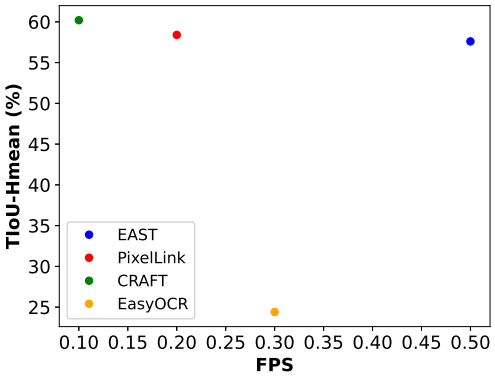
4.3.4 Results

Name	FPS	FPS rank	TiOU-Hmean	TiOU-Hmean rank	IoU-Hmean	IoU-Hmean rank
EAST	0.5	**1**	52.7%	2	76.0%	3
EasyOCR	0.3	2	23.6%	4	36.73%	4
PixelLink	0.2	3	52.6%	3	76.2%	2
CRAFT	0.1	4	56.6%	**1**	79.3%	**1**

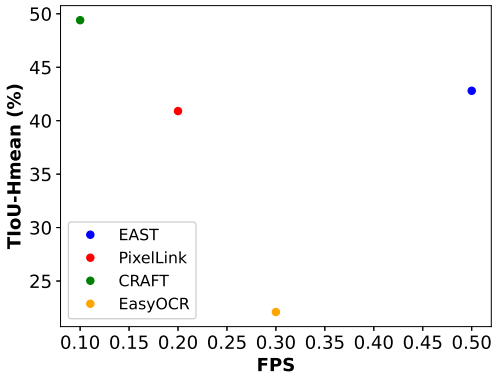
Table 4.4: Overview of weighted average STD results. The results on ICDAR2015 have the weight of 0.75, and results on Text in Videos of 0.25. This is done because ICDAR2015 is a more reliable and commonly used dataset for text detection.

As mentioned in Section 4.3.2 and 4.3.3, the STD methods will be evaluated according to speed and the TiOU-Hmean on the datasets ICDAR2015 [Karatzas et al., 2015] and Text in Videos [Iwamura et al., 2021]. As can be observed in Figure 4.7, in comparison to the results on ICDAR2015 there is a general drop in the Hmean for all evaluated metrics. However, the performance did not drop equally among all methods. The Hmean of CRAFT only dropped by around 10% whereas it dropped 15% for EAST and even 17% for PixelLink.

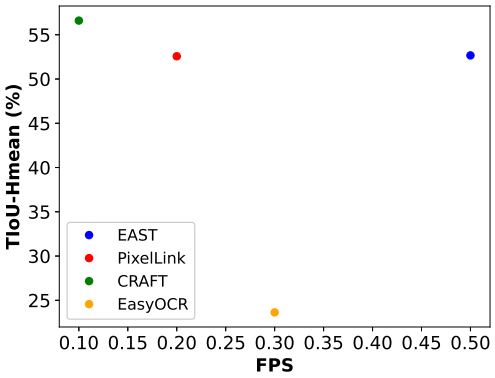
In the combined results one can see that CRAFT is the method with the highest TIoU-Hmean, with a weighted average Hmean of 56.6%. However, it is also the slowest method with a FPS score of 0.1. EAST represents a strong contrast to CRAFT. Whereas CRAFT was designed to handle extreme distortions of text, EAST was designed to have short inference times. With a FPS score of 0.5 it clearly sticks out as the fastest method. Moreover, its Hmean is the second highest with a weighted average Hmean of 52.7%. PixelLink is faster than CRAFT, but also slower than EAST. Furthermore, its weighted average Hmean is almost exactly the same as the one in EAST. The detection module



(a) ICDAR2015



(b) Text in Videos



(c) Combined results

Figure 4.7: TIoU-Hmean and FPS results for selected STD methods on the datasets ICDAR2015 and Text in Videos.

of EasyOCR is a definite outlier among the methods evaluated. Namely, its Hmean is consistently below PixelLink - the method with the second lowest Hmean - by more than 25%.

As can be seen in Table 4.4, the ranking of the IoU-Hmean more or less reflects the ranking of the TIoU-Hmean. Interestingly, the specialization of PixelLink in distinguishing dense text, which one might assume would lead to a more favorable TiOU result, actually resulted in a bigger drop than for EAST. This could either indicate a lack of success in distinguishing dense text or may indicate a tendency to cut ground-truth text, which would negatively affect TIoU-recall and thus also the Hmean value.

Given these results, CRAFT and EAST will be candidates for the final text extraction pipeline, where CRAFT is specialized in accuracy and EAST in inference time. Since the Hmean of PixelLink is almost the same as the one of EAST, but with a lower FPS score, PixelLink is deemed inferior to EAST. EasyOCR will still be evaluated in upcoming chapters for the sake of reference.

5

Scene Text Recognition

The goal of scene text recognition (STR) is to take a cropped image containing text and to translate it into a target string (see Figure 5.1). STR can also be used in a jointly-trained end-to-end text spotting system, in which STR operates directly on feature maps. However, this variant will be examined in Chapter 6.

Similar to the development of STD, STR also experienced the transition from using hand-crafted features to features learned by CNNs. Namely, traditional STR methods used hand-crafted features such as connected components, stroke width transform, and histogram of oriented gradient descriptors. Current STR methods use neural networks which automatically learn relevant features from the input image with substantial improvements relative to their traditional counterparts.

State-of-the-art methods can be divided into segmentation-based and segmentation-free methods. Methods from both types will be thoroughly examined within their respective category. The objective of this chapter is to analyze state-of-the-art methods in STR and to draw relevant conclusions. At last, the most promising approaches will be evaluated to find suitable candidates for the final STE pipeline.

5.1 Segmentation-Based Methods

Segmentation-based methods view STR as an image classification problem. Specifically, they try to locate each character in the text instance individually, apply a character classifier to the detected character components and then group the recognized characters into words/text lines.

This group of methods has largely fallen out of favor and segmentation-free/sequence-to-sequence recognizers have become mainstream, mainly because the pipelines require accurate detection of individual characters, which is a very challenging task. Furthermore, in contrast to sequence-to-sequence methods, segmentation-based methods fail to

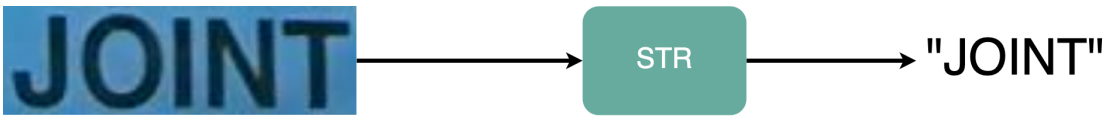


Figure 5.1: High-level illustration of the scene text recognition task on a cropped image

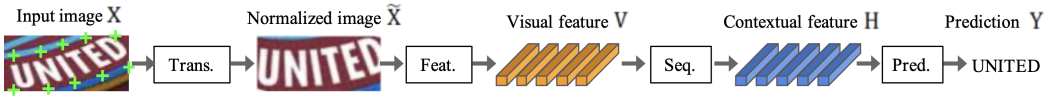


Figure 5.2: Overview of an example of the segmentation-free STR approach that includes all four stages. Image taken from [Baek et al., 2019]. The transformation and sequence modeling stage are not necessarily required.

take into account context beyond individual characters. However, there has very recently been a resurgence in segmentation-based recognizers. Namely, while the introduction of sequence-to-sequence based recognition has substantially simplified the task of recognizing text in the wild and lead to great performance on regular text, the performance on irregular text is unsatisfactory and has thus attracted a lot of attention from the computer vision community. This is largely due to the fact that sequence-to-sequence based methods were originally designed for one-dimensional prediction problems (see Section 5.2.1 for detailed information). Segmentation-based methods, however, are architecturally suitable for two-dimensional prediction problems. Taking advantage of this, Liao et al. [Liao et al., 2019] introduce a novel segmentation-based scene text recognizer that regards scene text as a two-dimensional distribution of features. Specifically, they introduce a Character Attention Fully Convolutional Network (CA-FCN) to predict characters at a pixel level. Last but not least, the word and the location of characters can be obtained by a formation module.

5.2 Segmentation-Free Methods

Segmentation-free methods, in contrast to segmentation-based ones, do not regard text as an accumulation of individual characters, but as a text line as a whole with intra-contextual dependencies. Specifically, segmentation-free methods try to detect whole text-lines and to translate said text-line into a target sequence via an encoder-decoder framework. Thus, it avoids the problematic character segmentation step in segmentation-based methods.

In this chapter, the architecture of segmentation-free methods will first be explored, which will serve as a foundation for Section 5.2.2 in which state-of-the-art segmentation-free methods will be investigated.

5.2.1 Architecture

Most segmentation-free methods can be divided into four stages: transformation, feature extraction, sequence modeling, and prediction stage [Baek et al., 2019]. It is important to note, however, that not all stages are required, and subsequently not all methods include all four stages. However, the feature extraction and prediction stage are essential for this STR category. The four stages are further illustrated in Figure 5.2.

1. Transformation Stage

   Text in the wild can come in multi-oriented, curved, and distorted forms, which may obstruct the recognition process. Consequently, this irregularity of text may cause two fundamental problems. For one, the feature extraction stage would have to learn to be invariant to these geometries [Baek et al., 2019]. Furthermore, in case the text is not horizontal, the recognition performance would suffer significantly since the two-dimensional text is treated one-dimensionally, which may lead to the loss of key information as well undesired noise [Liao et al., 2019].

   In order to alleviate these problems, the transformation stage aims to take an image as input and to output a normalized form. The most prominent approach is rectification, which aims to remove distortion and mitigate the difficulty of irregular text [Chen et al., 2020]. An early version of this form was the Spatial Transformation Network (STN), which aims to improve spatial invariance to large transformations of input [Jaderberg et al., 2015]. STN is differentiable and thus can be trained end-to-end. It is important to note that STN also works on feature maps [Jaderberg et al., 2015], and thus the transformation stage does not necessarily have to be the first step, but can for example be integrated within the feature extraction stage. Later, this network was improved upon by the Thin-Plate-Spline (TPS) to handle more complex and radical distortions [Shi et al., 2019]. Recently, Symmetry-constrained Rectification Network (ScRN) was introduced, which uses the center line of text and symmetrical constraints to further improve performance [Yang et al., 2019].

   As will be visible in Section 5.2.2, in which state-of-the-art methods will be explored, complex rectification modules to handle irregular text have become a hot topic. However, by nature of being an optional step, this step necessarily comes with a trade-off between accuracy and speed/memory consumption. Thus, employers of STR must carefully consider what the best trade-off is in regards to its application.

2. Feature Extraction Stage

   The feature extraction stage aims to extract features such that they reflect attributes relevant for character recognition, while suppressing features that are irrelevant to it such as font, lightning, color, size, and background [Chen et al., 2020]. More concretely, it takes an input image (or its transformed form), and produces a feature map [Baek et al., 2019]. Each column in the feature map represents a receptive field along the horizontal line of the input image [Baek et al., 2019].

   Nowadays, CNNs are usually used to extract relevant features. Among its widely used forms is the VGGNet, which uses smaller filters and more convolutional layers relative to its prior counterparts, in order to use fewer parameters and introduce more non-linearity [Simonyan and Zisserman, 2014]. In order to have a more powerful feature representation, some STR methods use ResNet, which is a particularly deep neural network. However, deep networks are more difficult to train due to the degradation problem [He et al., 2015]. In order to address this, they introduce

residual connections to facilitate the training [He et al., 2015]. Some methods such as R2AM use RCNN to capture longer contextual dependencies while not increasing the number of parameters needed [Lee and Osindero, 2016].

While deeper and more advanced feature extraction networks may result in better representation of relevant features, they also come at a cost of computation and computation consumption. Similar to the choice of using or not using transformation, this trade-off also has to be considered when choosing an STR method.

3. Sequence Modeling Stage

   The features that were extracted from the feature extraction stage are reshaped to be a sequence of features [Baek et al., 2019]. However, this sequence does not take advantage of its contextual information. Thus, the sequence modeling stage serves as an intermediate step between the feature extraction- and prediction stage to capture contextual information, which is more stable than treating each character independently [Chen et al., 2020].

   A widely used model for the sequence modeling stage is the multiple bidirectional long short term memory (BiLSTM) [Graves et al., 2009], which is able to capture long-term dependencies [Chen et al., 2020]. The reason why a bidirectional LSTM is used and not a normal (unidirectional) LSTM, is because in text (and in image-based sequences in general), contexts from both directions of the sequence are useful and complementary to each other [Shi et al., 2015]. Recently, a deeper BiLSTM was introduced which aims to improve the encoding of contextual cues [Litman et al., 2020]. However, BiLSTM has been explicitly excluded in some STR pipelines, as is the case with Rosetta [Borisyuk et al., 2018], whose authors argue that the step is too time-consuming and computationally intensive. Furthermore, it can lead to gradient exploding/vanishing which obstructs the training.

4. Prediction Stage

   The objective of the prediction stage is to take a sequence of features (either from the feature extraction- or the sequence modeling stage), and to produce a target string sequence [Baek et al., 2019] that has the highest probability among all possible sequences of characters. The most prominent approaches for this task is the Connectionist Temporal Classification (CTC) [Graves et al., 2006] [Shi et al., 2019] and the attention-based sequence prediction [Bahdanau et al., 2014].

   CTC was introduced in 2014 to train RNNs to label unsegmented sequences. That is to say, it takes a fixed number of feature columns and predicts an unfixed number of sequence labels. It does this by predicting a character (or blank symbol) for each column. Then each duplicated character and blank symbol is deleted to get the final target sequence. The blank symbol is introduced to keep wanted duplicated letters in a word (e.g. the two consecutive “e”’s in “cheese”) and to effectively distinguish between two words that exist with two consecutive characters as well as a single one (e.g. “be” and “bee”). This prediction technique was first successfully applied to speech recognition and later introduced to STR

[Shi et al., 2015]. However, while CTC shows promising and stable transcription, CTC suffers from three major problems. For one, the methodology of CTC is highly sophisticated, which causes high computational cost when applied to particularly long texts. Secondly, it suffers from overfitting, which causes highly peaky and overconfident distributions [Liu et al., 2018]. Last but not least, while CTC performs successfully for one-dimensional predictions such as speech recognition and regular text, it performs poorly when applied to two-dimensional predictions as is the case with irregular text [Wan et al., 2019]. In order to address this, a 2D-CTC was proposed which adds another dimension along the height dimension [Wan et al., 2019]. However, while this model improves the recognition performance for irregular text, the prediction in two dimensions still remains a problem and might rely on complex rectification networks.

The attention mechanism proposed by Bahdanau et al. [Bahdanau et al., 2014] was originally designed to handle neural machine translation, and has been adopted to perform prediction for STR. Namely, it learns the alignment between the input image and the ground truth text sequences by referring to the history of the target characters and the encoded feature vectors [Chen et al., 2020].

Even though this mechanism has become mainstream among STR, it still has its drawbacks. Firstly, similar to CTC, the attention mechanism was designed for one-dimensional sequences (machine translation) and is not directly generalizable to two-dimensional sequences such as irregular scene text. Recently, this issue has been addressed by various research papers such as Li et al. [Li et al., 2019], who proposed to improve the vanilla 1D attention mechanism to an attention mechanism that can handle two-dimensional sequences. Furthermore, the attention mechanism is computationally intensive and time-consuming. In order to address this, the Transformer [Vaswani et al., 2017] was employed in [Yu et al., 2020] to reduce computational complexity [Chen et al., 2020].

5.2.2 Method Overview

Name	Year	Transformation	Feature Extraction	Sequence Modeling	Prediction
CRNN	2015	×	VGG	BiLSTM	CTC
STAR-Net	2016	STN	ResNet	BiLSTM	CTC
R2AM	2016	×	RCNN	×	Attn
RARE	2016	STN	VGG	BiLSTM	Attn
ASTER	2019	TPS	ResNet	BiLSTM	Bi-Attn
SAR	2019	×	ResNet	BiLSTM	2D-Attn

Table 5.1: Overview of segmentation-free STR methods.

Now that the architecture of segmentation-free STR methods has been investigated thoroughly, this chapter serves as a concrete method overview of segmentation-free STR techniques. The architecture of most of these methods is captured in Table 5.1.

The Convolutional Recurrent Neural Network (CRNN) was one of the early adopters of the segmentation-free method. Architecturally, it does not include a transformation step, uses a VGG network (without the fully connected layers) as its backbone for the feature extraction, and uses BiLSTM for the sequence modeling stage. For the prediction stage, it uses CTC. Even though this method is rather old, it is still widely in use such as in EasyOCR [JaidedAI, 2021], Textboxes [Liao et al., 2016] [Liao et al., 2018] and PaddleOCR [Du et al., 2020]. However, this method has its limitations when it comes to irregular text. As was described in the previous section, vanilla CTC was not made for two-dimensional predictions such as irregular text. While rectification techniques such as STN [Jaderberg et al., 2015] can alleviate these problems, CRNN does not take advantage of that and skips the transformation stage. A year later, the SpaTial Attention Residue Network (STAR-Net) was introduced, which relative to CRNN contains an STN within its transformation stage and adopts a deep residual network for the feature extraction [Liu et al., 2016b]. Due to the fact that it tries to rectify images before feeding it into the feature extraction, and the fact that it adopts a very deep neural network that is more stable to distortions, it is expected to perform better than CRNN when it comes to irregular text. However, it is also harder to train [Cheng et al., 2018] and slower.

Around 2016, attention-based methods became mainstream. The Recursive Recurrent Nets with Attention Modeling (R2AM) was introduced in 2016 [Lee and Osindero, 2016]. It notes that one key success of STR with a non-recursive CNN as the backbone is the ability to capture contextual dependencies during character prediction by applying multiple convolutional layers that operate on the whole input image. One possibility to capture even longer contextual dependencies would be to increase the receptive field, which can be done by increasing the kernel size or by adding additional convolutional layers (f.ex. ResNet [He et al., 2015]). This option, however, systematically increases the number of parameters and increases the complexity, which leads to generalization and training issues. The alternative proposed would be to use a recursive or recurrent neural network, which would make the network arbitrarily deep without substantially increasing the number of parameters. While this method is easy to train, has low memory consumption, and is very fast, it is not expected to be as accurate as other state-of-the-art methods [Baek et al., 2019] especially when it comes to irregular text [Yang et al., 2017]. Shi et al. [Shi et al., 2016] introduce an attention-based method with STN and BiLSTM called “RARE”. This model was later improved upon in the method called “ASTER”, where the STN module was exchanged for an improved TPS rectification method which can handle more complex distortions [Shi et al., 2019]. Furthermore, they note that the traditional attention mechanism only captures output dependencies unidirectionally. Thus, they introduce an enhanced bidirectional attention mechanism with two decoders in opposite directions. Yang et al. [Yang et al., 2019] further improve the rectification module with their proposed Symmetry-constrained Rectification Network (ScRN). The conventional rectification module TPS predicts control points of the text outlines. These control points should ideally be evenly spread along the upper and lower edges of the text region and should be symmetrical to the center line of the text. However, these obvious constraints are not effectively utilized due to

the control points being predicted separately, which may lead to sub-optimal performance when text is highly curved and/or distorted. In order to improve this, ScRN uses the center line of text and imposes symmetrical constraints with computed geometrical attributes. However, in practice, training STN-based methods are extremely difficult without human-labeled geometric ground truth especially for complicated scene text such as curved, arbitrarily-oriented, or perspective text [Cheng et al., 2018]. Consequently, Cheng et al [Cheng et al., 2018] introduce a method called AON which can detect irregular text accurately without the need for STN-based modules. Specifically, their feature extraction phase can be divided into three major components: 1) The basal convolutional neural network for extracting low-level information; 2) The arbitrary orientation network (AON) for producing horizontal, vertical, and character clues; 3) the filter gate to systematically neglect irrelevant features [Cheng et al., 2018]. Li et al. [Li et al., 2019] also exclude the rectification network and instead try to improve the vanilla 1D attention mechanism to handle two dimensions. Moreover, for the feature extraction stage, they employ a 31-layer ResNet and BiLSTM for the sequence modeling. Other methods such as the one proposed by Luo et al. [Luo et al., 2021] introduce a transformation module that is not intended to perform rectification, but background removal. That is, in order to improve text recognition performance, they intend to separate text from complex backgrounds. In order to do this, they propose an adversial learning framework for the generation and subsequent recognition of multiple characters within an image. Baek et al. [Baek et al., 2019] take the architecture described in Section 5.2.1, create various module combinations (24 in total), and compare them with other state-of-the-art methods. In respect to the accuracy versus time trade-off, one of their proposed combination with TPS in their transformation stage, ResNet as the backbone for the feature extraction, BiLSTM for the sequence modeling and attention mechanism for the prediction (TPS-ResNet-BiLSTM-Attn) achieved particularly high accuracy, however, this also came with a decrease in speed. Conversely, their combination None-VGG-None-CTC was particularly fast, but achieved low accuracy. In regards to accuracy versus memory, the TPS-ResNet-BiLSTM-Attn combination also achieved the highest accuracy, but with the highest memory consumption. The previously introduced R2AM method [Lee and Osindero, 2016] (None-RCNN-None-Attn), used little memory but also achieved relatively low accuracy.

5.3 Evaluation

Five novel STR methods will be evaluated according to the word recognition accuracy (WRA) and inference time. Furthermore, the STR module in EasyOCR will be evaluated to compare how the aforementioned STR methods perform relative to a mainstream STR module. The evaluation will be carried out on three distinct datasets.

Name	Type	Training set	Year	Code
CRNN	Sequence-based	Synthtext	2015	[ClovaAI, 2021b]
EasyOCR (CRNN)	Sequence-based	N/A	2015	[JaidedAI, 2021]
TRBC	Sequence-based	Synthtext	2019	[ClovaAI, 2021b]
TRBA	Sequence-based	Synthtext	2019	[Media-Smart, 2021]
SARN	Sequence-based	Synthtext	2019	[Media-Smart, 2021]
CSTR	Classification-based	Synthtext	2021	[Media-Smart, 2021]

Table 5.2: Overview of STR methods used for evaluation.

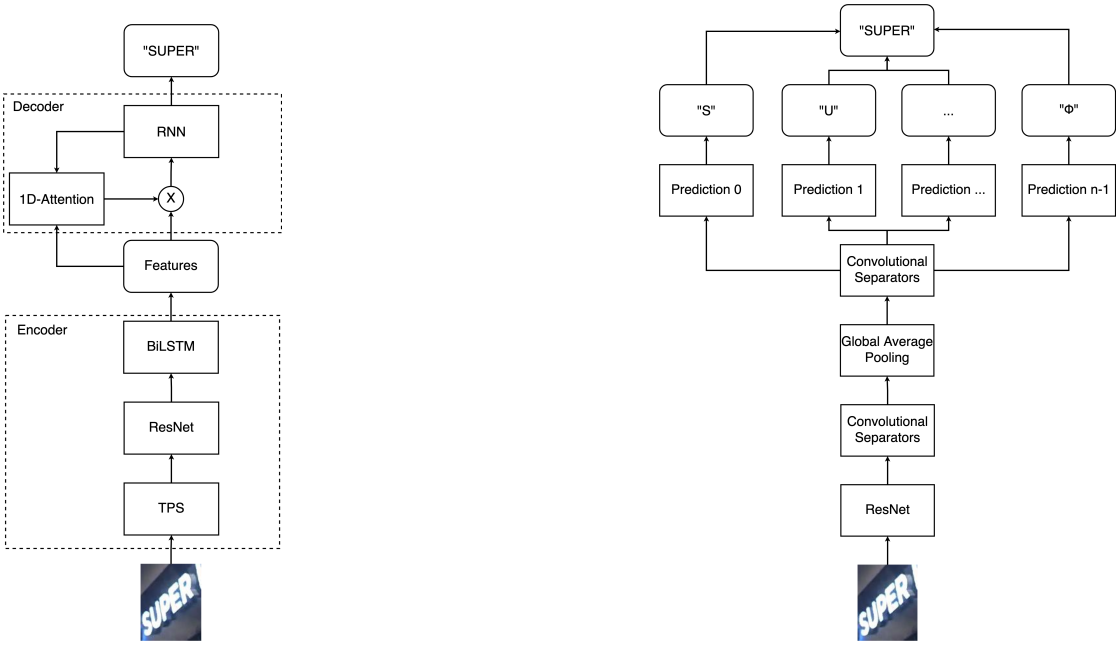
5.3.1 Methods

A distinct characteristic of scene text is its multi-oriented nature and the presence of curved text. This in particular has made STR a rather challenging task and has brought into question whether or not sequence-based methods are even suitable to begin with. To address this, five methods will be evaluated which are either widely used or specifically targeted at dealing with irregular text. Namely, CRNN [Shi et al., 2015], TPS-ResNet-BiLSTM-CTC (TRBC), TPS-ResNet-BiLSTM-Attn (TRBA), SARN [Lee et al., 2019], CSTR [Cai et al., 2021] and the previously mentioned EasyOCR module (CRNN) will be evaluated. All these methods except for the EasyOCR recognition module are trained on Synthtext [Gupta et al., 2016] to allow for a fair comparison. EasyOCR does unfortunately not share how and with which dataset their recognition module was trained on, nor does it give the possibility to train the module oneself.

CRNN is a sequence-based recognition method developed in 2015. It is comprised of a CNN that follows the VGG network to devise high-level features from the input image. The one-dimensional features are then directly fed to the CTC decoder which produces the final output string. Although it is a rather old method and does not make an effort in its architecture to handle non-horizontal text, it is still the most widely used method and is used in STE pipelines such as EasyOCR, Textboxes++, and PaddleOCR. Even though EasyOCR also uses CRNN for its recognition module, I still find it important to evaluate the EasyOCR method since the CRNN module does itself not conclusively reveal how well mainstream recognition modules perform.

TRBC and TRBA are the two methods that were shown to have the highest word-recognition accuracy in the paper by Baek et al. [Baek et al., 2019]. As their names suggest, they are sequence-based methods that are only distinguished by their decoder. Namely, one uses CTC whereas the other uses the one-dimensional attention mechanism. They both use TPS to rectify the image, ResNet to extract features, and BiLSTM to model contextual dependencies. Given the fact that they include more steps than CRNN, and use a deeper network to extract features, higher inference times are expected, but also higher accuracies.

There are two ways to deal with the reality of irregular text. On one hand, one can rectify the image beforehand such that the original text is transformed to be horizontally aligned with characters of equal dimensions. The two aforementioned recognition



(a) Sequence-based (b) Classification-based

Figure 5.3: Architectural comparison of a sequence-based STR method (TRBA) and a classification-based one (CSTR).

methods follow this philosophy by rectifying the image via TPS before it reaches the feature-extraction stage. This approach, however, suffers from the fact that one has to specify the family of transformations beforehand [Lee et al., 2019]. Thus, another approach was developed which takes the unmodified input image, learns 2D feature maps, and sequentially retrieves characters from the 2D feature map. A concrete example of this approach is the Self-Attention Text Recognition Network (SARN) developed by Lee et al. [Lee et al., 2019]. With the inclusion of this method, two different sequence-based approaches to handle irregular text are covered.

All the aforementioned methods are sequence-based. Although sequence-based methods are dominant among top-performing STR methods, they have been criticized for their complexity, long pipelines, and their architectural unsuitability for irregular text. As a response, novel segmentation-based methods have been developed such as the one introduced by Liao et al. [Liao et al., 2019]. Unfortunately, no available implementations for segmentation-based recognition methods could be found. However, a novel classification-based method named CSTR was recently introduced which drastically simplifies the pipeline. The method assumes that a word has a maximum of n characters. At first, it extracts features from the input image via the ResNet architecture. Then convolutional separators are applied to divide the feature map. Global average pooling is then applied to introduce global semantic information. At last, convolutional separators are applied. Each produced output channel represents a character prediction

which ranges from the symbols that can be contained in a word to the symbol ϕ which represents the absence of a character. See Figure 5.3 to see how CSTR compares to a sequence-based method.

5.3.2 Evaluation Metric

In order to evaluate the performance of STR methods, the widely used word recognition accuracy (WRA) will be used which is defined as

$$WRA = \frac{W_r}{W} \tag{5.1}$$

where W_r is the amount of correctly recognized words and W the amount of total words. Inspired by how Liu et al. [Liu et al., 2019] view the correspondence between detection and ground-truth not as a binary matter ("matching" or "not-matching"), but as a spectrum which can range from "matching" to "not-matching", I will extend the mainstream WRA metric to be tightness-aware. Namely, instead of classifying a recognition as being "correct" or "incorrect", the Jaro-Winkler Distance will be used to measure the correspondence between the predicted and the ground-truth string. This new metric will be termed Tightness-aware Word Recognition Accuracy ($TWRA$) and is defined as

$$TWRA = \frac{1}{N}\sum_{i=1}^{N} JaroWinkler(s_{target_i}, s_{pred_i}) \tag{5.2}$$

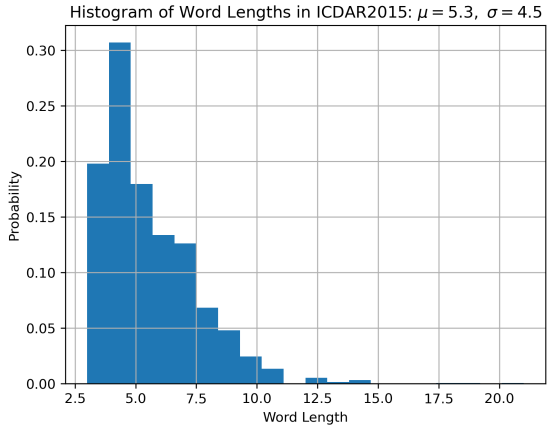
where s_{target} is the ground-truth string and s_{pred} the predicted string. $JaroWinkler$ is short for Jaro-Winkler Distance, which captures the normalized similarity between strings where 1 describes the case in which s_{target} and s_{pred} are the same and 0 the case in which they are not matching at all. Furthermore, the inference time will be captured.

5.3.3 Datasets

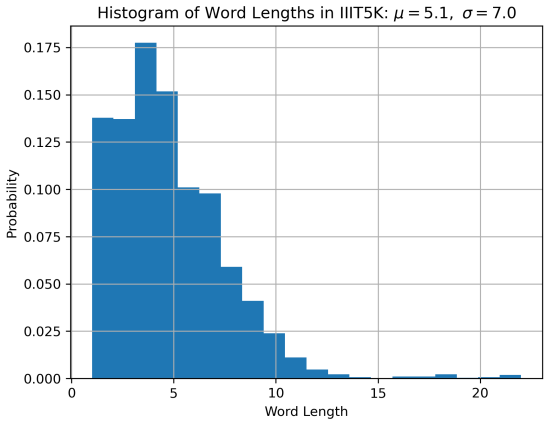
Name	Alphabet	Irregular text	Blur	Still image	Amount	Average word length
ICDAR2015	Latin	✓	✓	✓	2077	5.3
Text in Videos	Latin	✓	✓	×	2188	4.4
IIIT5K	Latin	✓	×	✓	3000	5.1

Table 5.3: Overview of datasets used for STR evaluation.

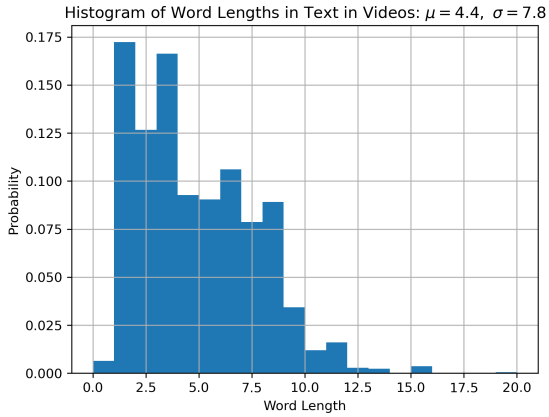
The datasets chosen to evaluate the STR methods are ICDAR2015 [Karatzas et al., 2015], IIIT5K [Mishra et al., 2012], and Text in Videos [Iwamura et al., 2021]. They were specifically chosen because each of them carries distinct characteristics that can affect the performance of STR in visual multimedia. Namely, the images in ICDAR2015 and Text in Videos are often heavily blurred, which is a particular characteristic of natural images and especially frames in videos. IIIT5K is a dataset with more regular text. However, even though the images are rarely blurred, the fonts are often very peculiar,



(a) ICDAR2015



(b) IIIT5K



(c) Text in Videos

Figure 5.4: Word length histograms of ICDAR2015, IIIT5K and Text in Videos.

which is also a characteristic of natural images. For example brand names often appear in natural images, and brands often do not choose regular fonts. The company symbol for "Coca Cola" is a primary example of that. Text in Videos also distinguishes itself from the other two datasets by being based on videos.

As can be seen in Figure 7.3, the average word lengths for ICDAR2015 and IIIT5K are quite similar with 5.3 characters and 5.1 characters per word respectively. Moreover, the standard deviation of word lengths in IIIT5K and Text in Videos are rather large with Text in Videos having a standard deviation of 7.8 characters. Given the heavy blurring, the unusually short word lengths and the high standard deviation, Text in Videos may pose a particularly hard challenge to the STR methods.

Name	FPS	FPS rank	WRA	WRA rank	TWRA	TWRA rank
CRNN	42.1	**1**	54.4%	5	81.3%	5
EasyOCR	24.6	2	30.4%	6	65.0%	6
TRBC	8.4	3	60.7%	4	84.1%	4
TRBA	6.2	4	66.6%	2	85.8%	3
SARN	2.7	5	67.9%	**1**	87.3%	**1**
CSTR	1.1	6	65.9%	3	86.7%	2

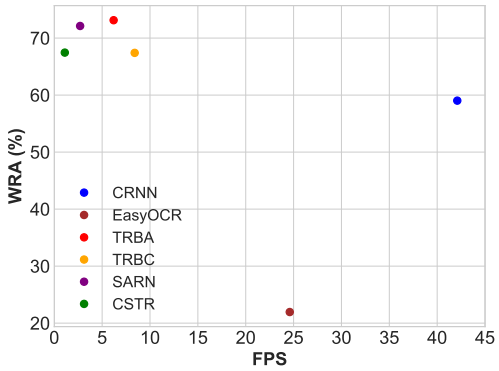
Table 5.4: Overview of the weighted average STR results. Results on ICDAR2015 and IIIT5K have weights two times as high as Text in Videos, because the former two are more commonly used and reliable datasets for STR.

5.3.4 Results

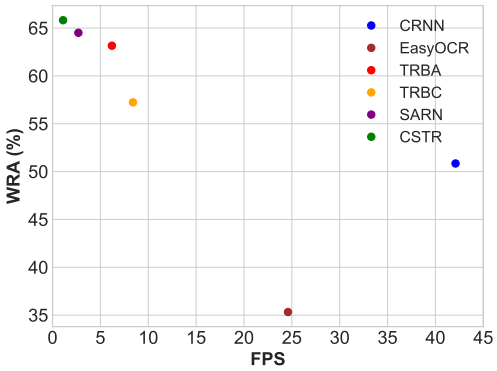
In Figure 5.5 the WRA is plotted relative to the FPS results. The raw results are shown in Table 5.4. As can be seen, EasyOCR is a clear outlier with its low WRA scores. Although it is faster than TRBA, TRBC, SARN, and CSTR, it is noticeably slower than CRNN which in turn also has a higher WRA. CRNN is by far the fastest method with a FPS score of 42. Nevertheless, its WRA is consistently low. Moreover, it seems to perform especially poorly relative to the others in ICDAR2015 and Text in Videos, where it is 6-8% lower than the next best method and 14-16% lower than the best method. This is very likely the effect of not having a rectification step which the other sequence-based methods have. CSTR is the slowest method with an FPS of 1.1. While its WRA is among the highest, it is consistently lower than SARN and also lower than TRBA in the combined results. Since SARN and TRBA also have a higher FPS, they are objectively superior to CSTR. SARN is all in all the method with the highest WRA as can be seen in Figure 5.5d. However, its WRA is only 1% higher than the next best method TRBA. Furthermore, TRBA also has an FPS that is higher by 3.5. TRBC has an FPS that is higher by 2.2 than TRBA. Its WRA however is 6% lower.

The combined results with the TWRA metric in Figure 5.5e show a very similar picture. However, the TWRA score for TRBC is noticeably closer to its better-performing competitors. Moreover, the TWRA score for CSTR is higher than the one for TRBA, even though the WRA score for TRBA was higher. This may indicate that the erroneous predictions for TRBA may be less recoverable than the other ones.

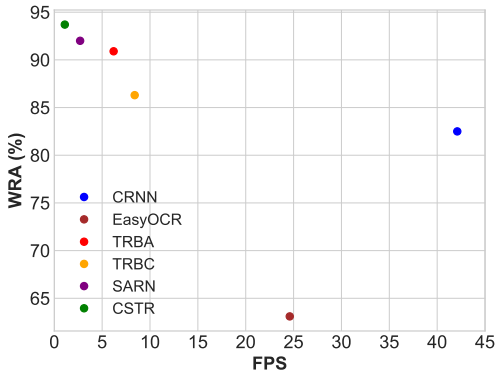
With all of this in mind, CSTR, TRBC, EasyOCR, and CRNN are not going to be candidates for the final text extraction pipeline. CSTR has a high WRA and TWRA, however, SARN has an even higher one and also a higher FPS. EasyOCR and CRNN are very fast, however, their accuracy is relatively low, especially on ICDAR2015 and Text in Videos which closely simulate a real-world scenario. TRBC is a good method with high accuracy and high FPS. Nevertheless, since the final scene text extraction in vitrivr will not happen in real-time, accuracy is the primary concern and in terms of accuracy, it lacks behind SARN and TRBA. Thus, SARN and TRBA will be candidates for the final STE pipeline.



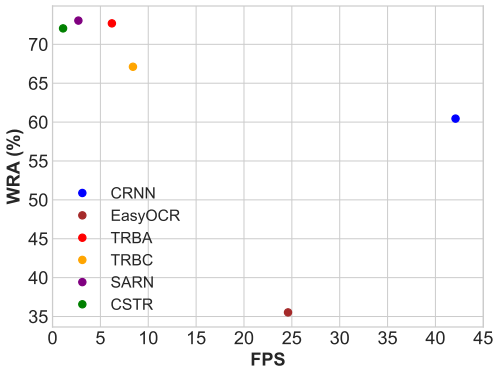
(a) Results on ICDAR2015



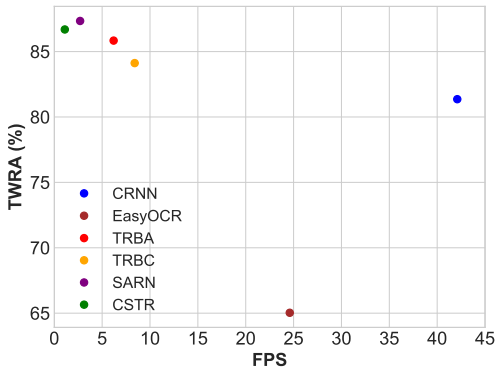
(b) Results on Text in Videos



(c) Results on IIIT5K



(d) Combined results



(e) Combined results (TWRA)

Figure 5.5: WRA, TWRA and FPS results for STR on the datasets: ICDAR2015, IIIT5K and Text in Videos.

6

Scene Text Extraction in Still Images

The goal of STE is to take an unconstrained image containing text and output a target string. Until recently, this task has been seen as being comprised of two individual subproblems. Namely, STD which takes an unconstrained image and produces bounding boxes enveloping the text instance, and STR, which takes the subsequently cropped image with the text instance and generates the final string sequence. Thus, methods explored in Section 4 and 5 can be trained separately and arbitrarily matched to complete the task of STE. Recently, researchers have begun to treat STE as a unified end-to-end trainable network rather than two separately trained ones. This group of methods has multiple advantages such as information sharing, joint optimization, faster inference time, etc., and has thus become a hot topic in the computer vision community.

In this chapter, concrete realizations of STE systems will be analyzed, ranging from separately trained (modular) methods to novel jointly optimized ones. In Section 6.3, the pruned sub-components from Section 4.3 and 5.3 will be matched to form end-to-end (separately trained) STE methods. They will subsequently be evaluated and unsuitable ones pruned.

6.1 Separately Trained Methods

As previously mentioned, separately trained methods view STE as two separate subproblems which can be arbitrarily matched. This architecture thus allows for a wide variety of potential STD and STR combinations to extract scene text from unconstrained images. This section, however, is restricted to the analysis of already realized combinations.

6.1.1 TextBoxes++

TextBoxes++ was initially described in Section 4.2.2 as a STD method. Specifically, it optimizes the one-stage object detector SSD for scene text by taking into account its large aspect ratio and multi-oriented nature. However, in their paper they also pair their introduced STD method with a recognition module, CRNN [Shi et al., 2015]. CRNN is a relatively old but popular CTC-based recognizer that uses the VGG network as a backbone for the feature extraction and BiLSTM for the sequence modeling stage.

Furthermore, they propose to refine the detection by taking into account the recognition score in order to eliminate false positives.

Based on the architecture of the proposed method, one can make several assumptions. The STD module is based on the one-stage object detector SSD, and is thus expected to be relatively fast because it skips the region proposal step in two-stage models. However, in terms of accuracy, it is not expected to outperform two-stage methods such as QRPN [Wang et al., 2020]. CRNN is expected to perform sub-optimally when it comes to irregular text. Vanilla CTC is hardly applicable to two-dimensional prediction problems such as irregular text, and CRNN skips the transformation stage in which they could have rectified the image to improve performance on irregular text.

6.1.2 EasyOCR

EasyOCR [JaidedAI, 2021] is a widely used OCR engine for text in the wild which uses CRAFT [Baek et al., 2019]t o detect text and CRNN [Shi et al., 2015] for STR. CRAFT is a character-level segmentation-based method intended to be invariant to extreme aspect ratios, shape distortion, and character size variety. The sub-components of EasyOCR have already been evaluated in Section 4.3 and 5.3, where both components performed substantially worse than the other methods evaluated.

6.1.3 PaddleOCR

A STE system named PaddleOCR [Du et al., 2020] was introduced in 2020 and uses the newly introduced Differentiable Binarization [Liao et al., 2020] STD method. Differentiable Binarization significantly reduces the complex post-processing binarization required in most segmentation-based methods by integrating the binarization within the segmentation task itself. Thus, this STD procedure is expected to be fast in comparison to other segmentation-based methods. Like in the previously introduced methods, PaddleOCR also uses CRNN for text recognition.

6.2 Jointly Trained Methods

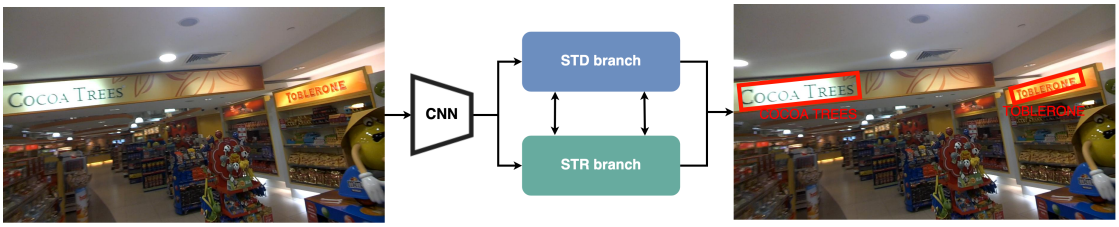


Figure 6.1: Illustration of the jointly trained STE architecture. Image taken from ICDAR2015 [Karatzas et al., 2015].

Even though separately-trained STE pipelines are very popular, they carry several disadvantages in comparison to jointly-trained ones. Firstly, errors can accumulate which may lead to a large number of garbage predictions. Secondly, since each module within the pipeline depends on the output of the previous one, the joint optimization of the end-to-end performance, and fine-tuning the pipeline with new data or adapting it to a novel domain, becomes particularly arduous [Qin et al., 2019]. Thus, jointly trained methods have recently been introduced which mitigate these problems. The idea behind jointly trained STE methods is for the detection and recognition branch to depend on the same feature maps produced by a CNN feature extractor. The STD and STR branch can thus be jointly trained, and at inference time, the pipeline can compute bounding boxes and text predictions in a single forward pass [Qin et al., 2019]. A high-level illustration of such an architecture is presented in Figure 6.1.

To my knowledge, the first jointly trained STE method was created by Bartz et al. [Bartz et al., 2017], who introduced a network called "SEE", which integrates and jointly learns a spatial transformer network, which is responsible for detecting text, and a recognition module. Later, Qin et al. [Qin et al., 2019] develop a method based on a Mask R-CNN detector and a segmentation-free, attention-based STR method. The method has a particular strength in recognizing curved text, and text of arbitrary shapes in general. A key idea is to skip the rectification step, and to feed the cropped masked text instance features directly to the decoder. The model then learns where to start decoding and how to update the attention weights. Another jointly trained method is called TextDragon [Feng et al., 2019], which similar to the previously introduced method also focuses on curved text. The text detector in TextDragon is designed to represent the shape of the text with multiple quadrangles. In order to extract regions containing arbitrary text from the feature maps, a novel operator named RoISlide is introduced. Based on the regions extracted from the RoISlide, a CTC-based recognizer is applied to extract the final string.

While jointly trained methods are certainly promising, they will not be a part of the upcoming evaluation section due to the current unavailability of well-programmed and functioning code.

6.3 Evaluation

In this section, the best methods found in Section 4.3 and 5.3 will be matched to create STE pipelines for still images. Moreover, it will be investigated whether or not the quadrilateral bounding boxes (in contrast to rectangular ones) from the detection methods can be utilized to improve end-to-end performance.

6.3.1 Methods

In Section 4.3, CRAFT and EAST were found to be the best performing methods. In Section 5.3, TRBA and SARN were shown to be the best performing STR methods. Thus, these methods will be combined to create the following end-to-end methods:

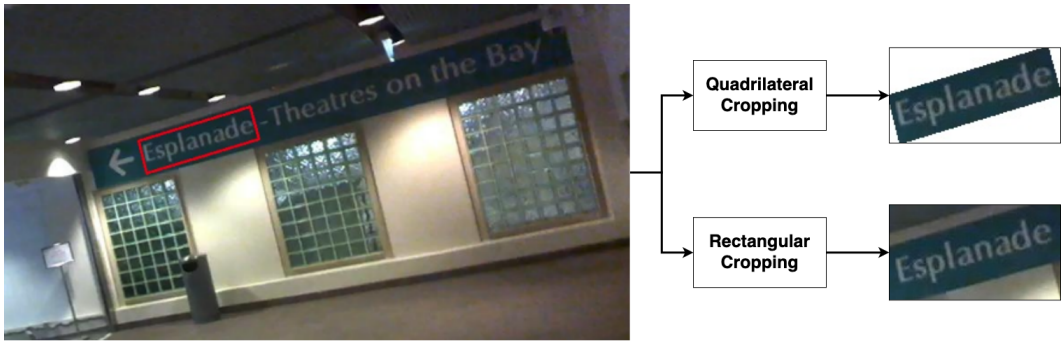


Figure 6.2: Comparison of quadrilateral and rectangular cropping. Image taken from ICDAR2015. Red: detection.

- CRAFT-TRBA
- CRAFT-SARN
- EAST-TRBA
- EAST-SARN

Given the evaluation results of the sub-components, CRAFT-SARN is expected to have high accuracy, but also high inference times. EAST-TRBA is a counterpart to the aforementioned method with low inference time and lower accuracy. The other two methods, namely CRAFT-TRBA and EAST-SARN, are expected to fall somewhere between CRAFT-SARN and EAST-TRBA in respect to accuracy and inference time. EasyOCR will also be evaluated to have a point of reference.

In order to match the detection and recognition modules, an intermediary step is required which takes the detected bounding box and crops the image accordingly. Since CRAFT and EAST produce quadrilateral bounding boxes, there are two different ways of accomplishing this task. For one, a traditional rectangular cropped image could be obtained. This method, however, does not take advantage of the fact that the bounding boxes are quadrilateral. In order to take advantage of the quadrilateral bounding boxes, the area within the quadrilateral is masked and the rest of the image is transformed into a white background. Both procedures are illustrated in Figure 6.2. The idea behind the latter approach is to remove everything from the image that does not relate to the text so that the recognition module can concentrate on the area of interest. This may be especially helpful for multi-oriented and curved text, since a rectangular box may result in a lot of noise. In an effort to find the optimal cropping approach, the STE pipelines will be created and evaluated with both approaches.

6.3.2 Evaluation Metric

To evaluate the STE methods, two metrics will be used. The WRA-based $Hmean$ ($WRA - Hmean$), and the $TWRA - Hmean$.

The $Hmean$ is the same for both and defined as

$$Hmean = 2 \times \frac{Recall \times Precision}{Recall + Precision} \tag{6.1}$$

However, $Recall$ and $Precision$ have differing definitions. $WRA-Recall$ and $WRA-Precision$ are defined as

$$WRA - Recall(G, D) = \frac{\sum_{i=1}^{G} Match_G(G_i)}{G} \tag{6.2}$$

$$WRA - Precision(G, D) = \frac{\sum_{i=1}^{D} Match_D(D_i)}{D} \tag{6.3}$$

where $Match(S_i)$ returns 0 if no string matches S_i and 1 if there is a complete match. Thus, WRA-Hmean is a measure of all the strings that can effectively be retrieved.

$Recall$ and $Precision$ for the TWRA-Hmean on the other hand are defined as

$$TWRA - Recall(G, D) = \frac{\sum_{i=1}^{G} BestMatch_G(G_i)}{G} \tag{6.4}$$

$$TWRA - Precision(G, D) = \frac{\sum_{i=1}^{D} BestMatch_D(D_i)}{D} \tag{6.5}$$

$BestMatch$ refers to the optimal relative similarity matches between two sets of strings. Namely, the Jaro-Winkler Distance is used to compute the normalized string similarity between every string of the two sets. The Hungarian algorithm is then used to obtain the optimal matches. Thus, instead of classifying a prediction as wrong because it does not appear in the ground-truth, the optimal relative match is taken. E.g. if the prediction is "Starbuck", but the ground-truth is "Starbucks", then the prediction will not be classified as simply incorrect, but as a 95% match. This is done to simulate fuzzy searches which are enabled in vitrivr (see Section 2.3).

6.3.3 Datasets

Name	Amount of Images	Amount of Ground-Truth	Blur	Still image
ICDAR2015	500	2081	✓	✓
Text in Videos	538	5275	✓	×

Table 6.1: Overview of datasets used for the STE evaluation.

The STE methods will be evaluated on the already introduced ICDAR2015 [Karatzas et al., 2015] and Text in Videos [Iwamura et al., 2021] datasets. What has not been mentioned yet in this context is the amount of ground-truth bounding boxes relative to the amount of images. ICDAR2015 has an average of 4.2 bounding boxes per image, whereas Text in Videos has more than twice as many with an average of 9.8. This may make the recognition of all ground-truth labels particularly hard due to the density of text which segmentation-based STD methods (such as EAST and CRAFT) have particular troubles with.

6.3.4 Results

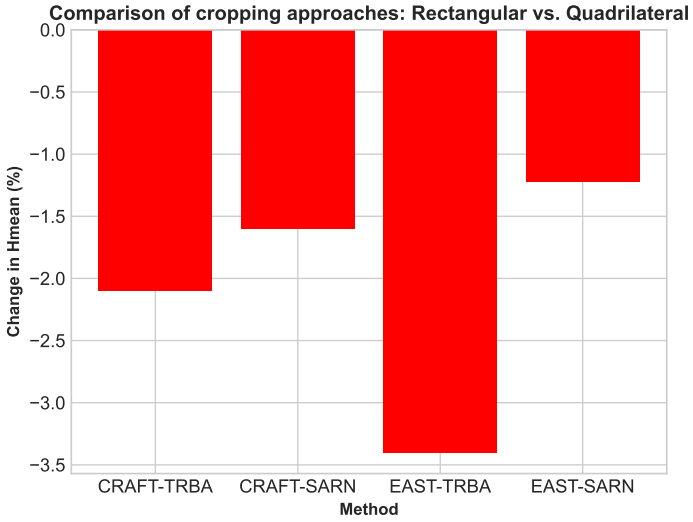
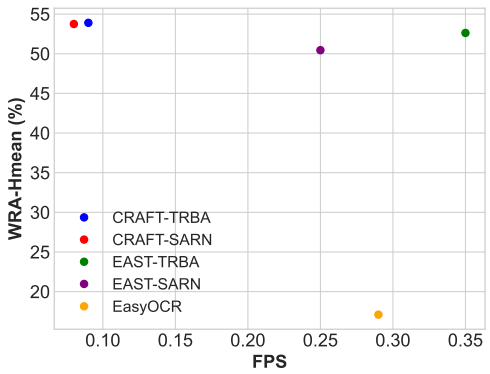


Figure 6.3: Portrayal of how quadrilateral cropping affects the WRA-Hmean relative to rectangular cropping.

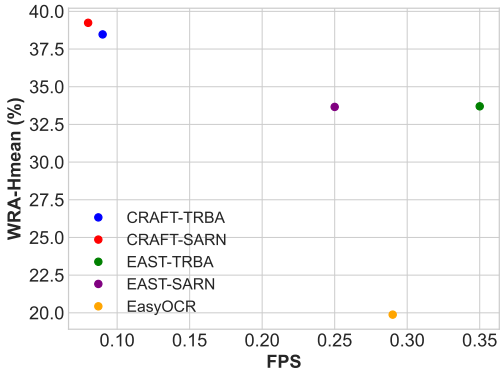
The STE pipelines were created with rectangular and quadrilateral cropping as an intermediary step. Interestingly, quadrilateral cropping did not positively affect the end-to-end performance. In fact, as can be seen in Figure 6.3, quadrilateral cropping actually lead to small decrease in the WRA-Hmean for all evaluated methods. This may be explained by the fact that the STR modules used were not trained on cropped quadrilateral images. Nevertheless, since quadrilateral cropping has shown to slightly negatively affect the accuracy, the rest of this chapter will be an evaluation of pipelines that use rectangular cropping.

As can be seen in Figure 6.4, the WRA-Hmean and TWRA-Hmean results on the dataset ICDAR2015 are very close for all methods except for EasyOCR. However, in terms of inference time EAST-TRBA clearly sticks out with the highest FPS score of 0.35. For the dataset Text in Videos the Hmean results start to diverge with CRAFT-TRBA and CRAFT-SARN taking the lead. Moreover, the Hmean scores for all methods are noticeably lower for the dataset Text in Videos than for ICDAR2015. It is important to mention here that that Text in Videos is a highly irregular dataset, with scene text that is often even unreadable to the human eye. When looking at the combined results, CRAFT-TRBA and CRAFT-SARN take a slight lead in terms of Hmean. Nevertheless, EAST-TRBA is only marginally behind the aforementioned methods.

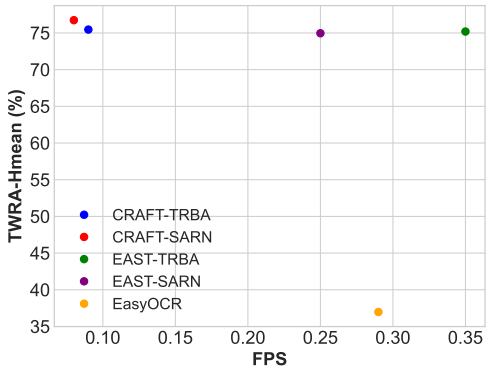
With these results in mind, a decision can be made. EAST-SARN has a similar WRA- and TWRA-Hmean to EAST-TRBA, however, EAST-TRBA is noticeably faster. Thus, EAST-SARN will not be part of Section 7.3. CRAFT-SARN and CRAFT-TRBA are very close in terms of Hmean and FPS. While they are both slower than the methods



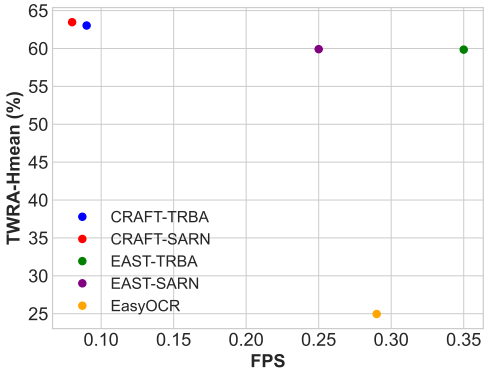
(a) WRA results on ICDAR2015



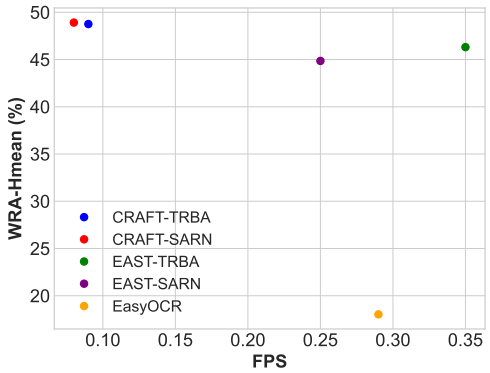
(b) WRA results on Text in Videos



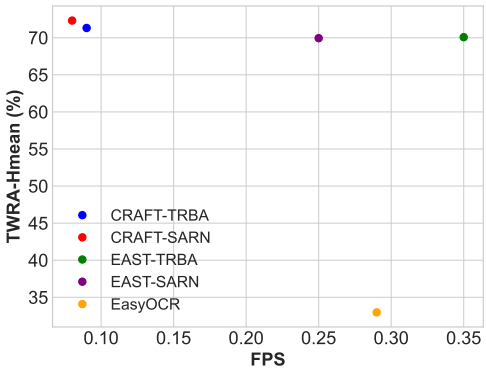
(c) TWRA results on ICDAR2015



(d) TWRA results on Text in Videos



(e) WRA combined results



(f) TWRA combined results

Figure 6.4: WRA-Hmean, TWRA-Hmean and FPS results for STE methods on the datasets: ICDAR2015, and Text in Videos.

Name	FPS	FPS rank	WRA	WRA rank	TWRA	TWRA rank
EAST-TRBA	0.35	**1**	46.3%	3	70.1%	3
EasyOCR	0.29	2	18.0%	5	33.0%	5
EAST-SARN	0.25	3	44.9%	4	69.9%	4
CRAFT-TRBA	0.09	4	48.8%	2	71.3%	2
CRAFT-SARN	0.08	5	48.9%	**1**	72.3%	**1**

Table 6.2: Overview of STE results in still images.

based on the EAST text detector, they have higher accuracy especially for the dataset Text in Videos. Moreover, the slow inference time can be addressed with various measures when it comes to video optimization. Due to this, two methods will remain to be evaluated for video optimization: CRAFT-SARN and EAST-TRBA. These are two distinct methods with their own advantages and disadvantages, where CRAFT-SARN is a relatively slow, but highly accurate method, and EAST-TRBA a faster, but less accurate one.

7

Scene Text Extraction in Videos

Research so far has heavily focused on STE in still images rather than on its video-based counterpart. Nevertheless, the STD and STR modules developed in the process are necessary components for STE in video and are thus inherently useful. However, STE in videos is not merely STE performed on multiple frames, since spotted text in adjacent frames may describe the same text instance. Thus, STE in videos introduces another component, multi-object tracking, to match equivalent text instances in multiple frames. Last but not least, once equivalent text instances have been assembled, the question arises which recognition result to choose in the case that the recognitions are not equal.

In order to shed a light on these newly introduced components and difficulties, Section 7.1 will introduce several approaches for STE in videos. In Section 7.2, my novel method called HyText will be introduced and explained in detail. Last but not least, the aforementioned method will be evaluated in Section 7.3.

7.1 Survey

Text tracking methods can be divided into three subgroups: Bayesian framework, template matching, and tracking-by-detection. Bayesian framework-based methods tend to use particle filtering and the Kalmar filter to track text. Template matching seeks the most similar region as the template image [Cheng et al., 2019]. However, the aforementioned methods fail to solve the reinitialization problem. Namely, if scene text were to appear after the tracking was initialized, it would not be able to be detected and thus be lost. Tracking-by-detection, on the other hand, is not susceptible to the reinitialization problem by detecting text in every frame of the video. Equivalent text instances in adjacent frames are then matched via advanced feature extractors and similarity functions.

A method introduced by Wang et al. [Wang et al., 2017], approached STE in video by first detecting and recognizing text in each frame via a jointly-trained end-to-end method whose network is based on the VGG16 Faster R-CNN. After extracting text in each individual frame, the method uses an association formula comprised of the position of the text instance, the recognized text as well as the frame offset to match equivalent text instances. The method then uses majority voting to determine the most likely text for the text stream. The paper by Wang et al., unfortunately, does not mention what would happen if there is no singular majority string.

The aforementioned method has been criticized by Cheng et al [Cheng et al., 2019]. They argue that reading text in every frame is excessively computationally costly and thus operationally unsuitable. Moreover, recognizing text in every frame also introduces erroneous results due to the abundance of "low quality" (e.g. blurring, rotation, perspective distortion, poor illumination, etc.) text regions. To circumvent these problems, Cheng et al. introduce a method called You Only Recognize Once (YORO), which, as the name already suggests, only recognizes text in one region from a text stream by selecting the most "high quality" one. YORO, like the previously introduced method, follows the tracking-by-detection framework by detecting text in every frame. The detection module is based on the EAST text detector, and its score output map is temporally enhanced via a spatial-temporal aggregation strategy that optimizes the output in the current frame by referring to the output in adjacent frames. Then, a module they call Text Recommender is responsible for three tasks: text quality scoring, text tracking, and text recognition. All of these tasks depend on the same feature map produced by a ResNet-based feature extractor. The text quality scoring is performed on each detected text region and calculates scores between 0 and 1, where 1 denotes the highest quality achievable. The text tracking is achieved by comparing the L2 normalized feature maps, and optimal matches are determined via the use of the Hungarian Algorithm. At this point, text streams have been generated, and each text region within the stream has a quality score. The region with the highest quality is selected and an attention-based recognition module produces the final string.

7.2 HyText

Similar to Cheng et al. [Cheng et al., 2019], I argue that recognizing text in each frame is too computationally expensive and might introduce erroneous recognitions. However, in contrast to Cheng et al., I argue that the computational cost of STE in video is mainly driven by the detection module. For example, the CRAFT detection module has an FPS rate of 0.1 (see Section 4.3.4), whereas TRBA has one of 6 (see Section 5.3.4). Newer and more accurate detection methods such as TextFuseNet [Ye et al., 2020] have an even lower FPS. Thus, unless the video has an unusually high amount of text instances per frame, the detection module is clearly the main driver in computational cost. Moreover, recognized text is a very salient component of a text stream, and can be used to validate the internal consistency of a text stream, and associations to other streams. To take advantage of this observation, I introduce the **Hy**brid Bidirectional **Text Ext**ractor (HyText), which can reliably extract scene text from videos by only detecting text in a subset of frames. Moreover, it can utilize recognized text to validate streams and for inter-frame association, without having to rely on per-frame recognition. In order to explain the HyText method in sufficient detail, its description is divided into three sequentially ordered parts.

7.2.1 Text Stream Formation

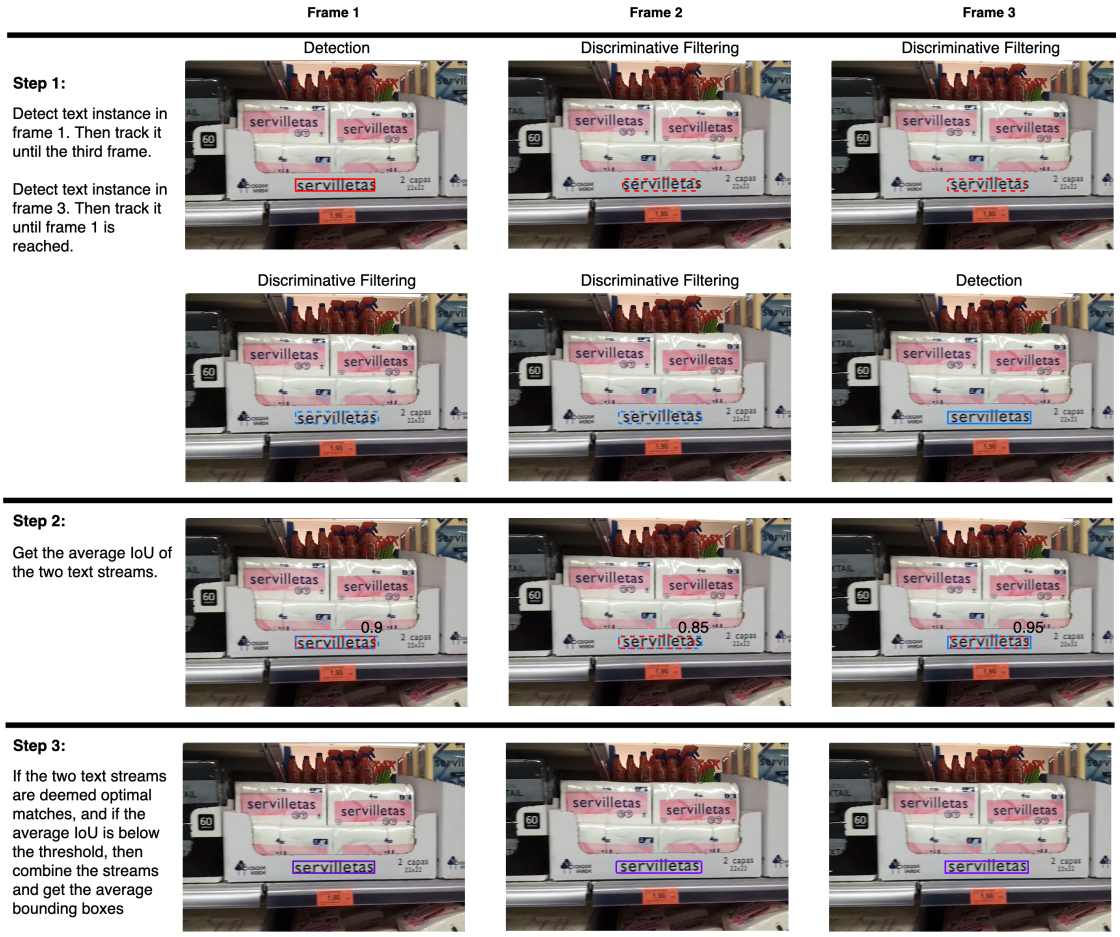


Figure 7.1: Example of the text formation phase of HyText for a particular text instance. Connected line: Detection. Dotted line: Discriminative filtering. Image taken from Text in Videos dataset [Iwamura et al., 2021].

While tracking text with the Bayesian framework or template matching cannot handle reinitialization, it has a few important advantages over tracking-by-detection. For one, since the relative region of the text in the previous frame is known, these methods seem to be particularly faster than applying object detection. Moreover, they can handle occlusion a lot better since they are able to estimate the probable region of the text. Thus, HyText aims to take advantage of the speed and superior occlusion treatment, while still being able to take into account newly appearing text instances. HyText accomplishes this by allowing the applicant of HyText to define a rate at which the detections may occur. For example, if the user of the method specifies the rate to be two, then a detection will be done at every second frame. In order to not miss the text occurrences between the frames at which detections were done, HyText uses the

discriminative correlation filter method named CSR-DCF [Lukezic et al., 2016] to track the text instance in the frames at which no detection was performed.

To more concretely show how text streams are initially formed, Figure 7.1 provides an illustrative example. Let us say that the rate is specified as two. At first, the bounding boxes are obtained in the first frame via the detection module. Then, CSR-DCF tracks the bounding boxes until it reaches the third frame. In the third frame, a detection is also done, but here the bounding boxes are tracked three frames back (to the first frame). Then, the text streams that originate from the detections in the first frame are associated with the text streams originating from the third frame. The association here, contrary to most tracking algorithms, is an intra-frame rather than an inter-frame association. Since the temporal aspect of association is thus removed, the association becomes radically simpler. Namely, the association between text streams can simply be captured by the average intersection over union of the text streams. I use the Hungarian algorithm to find the optimal one-to-one match between the streams. If the average intersection over union is below 0.4, the association is regarded as insufficient and the two text streams are deemed unreliable and are removed. If it is above the threshold of 0.4, the text streams are combined and regarded as one. Here, the question arises which bounding box to choose, since they both describe the same text instance within the same frame. Since both a too small and too big bounding box can negatively affect the later recognition as mentioned in Section 4.3.2, the average of the two bounding boxes is taken. This procedure is then continued with text streams originating in the third frame and in the fifth frame, then in the fifth frame and in the seventh frame, etc. until the end of the video is reached.

7.2.2 Text Stream Recognition

Figure 7.2: Example of the disappearing text phenomenon where quality scoring and unfiltered majority polling produce incorrect results. Image taken from Text in Videos dataset.

Once the procedure in the previous section is done, we will have a list of text streams describing individual text instances. The next step is to extract the correct string from the streams. In order to do this, I introduce a hybrid approach that does not entirely depend on majority voting, nor on text selection such as in YORO [Cheng et al., 2019]. Namely, I argue that unfiltered majority voting may introduce flawed text recognitions which can be easily filtered out. Moreover, the text scoring system in YORO may reliably select the highest quality bounding box for recognition. However, it neglects the

most radical deteriorating factor: disappearing text. Cut text is a particularly abundant characteristic in video, since a text instance may be in the center of the image in one frame, but then, as the camera moves, progressively disappear. In the process, the text gets cut more and more, and if that text is used for recognition, the recognition will necessarily be incorrect. This phenomenon is exemplified in Figure 7.2.

In order to address this, we calculate the aspect ratio of each bounding box within a stream and prune those whose aspect ratio is below the fiftieth percentile. With this procedure, I effectively mitigate the problem of disappearing text and simultaneously reduce computational cost. For the remaining bounding boxes, the recognition is performed, and the final string is obtained via majority polling. However, there may be cases in which two or more strings appear equally often in the stream. As a means to address this, I prune streams that cannot agree on one or two strings, as they may be indicative of unreliability. If two strings remain after the majority polling, I apply the global pairwise sequence alignment algorithm called "Needleman-Wunsch" [Needleman and Wunsch, 1970]. The Needleman-Wunsch algorithm is typically used in bioinformatics to align protein or nucleotide sequences. In my case, I use the algorithm to extract a common substring from the two recognitions. While this may not result in a completely accurate final recognition, it is still of value since the default scene text search in vitrivr will not look for complete matches, but for relative matches using the edit distance (see Section 2.3.2).

7.2.3 Text Stream Aggregation

With the aforementioned steps completed, the STE task for video could be regarded as completed. However, the procedure so far can still be improved to handle text instances that disappear and reappear again. Namely, if a text instance disappears at a time at which a detection is performed, and then reappears later, the text instance will be regarded as two separate ones, and thus results in unwanted duplicity. This can also occur in the case that the detector module simply does not detect a text instance in a certain frame, e.g. due to high motion blur.

As a means to overcome this problem, I introduce an additional step that is heavily inspired by the method introduced by Wang et al. [Wang et al., 2017]. I start with text streams that have the minimal distance between them, which in all cases would be the rate that the user-specified. The distance between two text streams is regarded as the difference between the tail and the head of a stream. For example, if a stream starts at frame 0 and stops at frame 62, and another stream starts at frame 64 and ends at 80, then the distance between the two would be 2. To check whether or not they actually describe the same text instance, the following formula is applied:

$$S_{ab} = k_1 \times S_{dis}^{spatial} + k_2 \times S_{op}^{area} + k_3 \times S_{winkler}^{jaro} + k_4 \times S_{offset}^{frame} \tag{7.1}$$

Where k_1, k_2, k_3, k_4, are the respective weights set at 1, 1, 10, and 0.2.

$S_{dis}^{spatial}$ refers to the spatial distance between the bounding boxes and is calculated like the following:

$$S_{dis}^{spatial} = \frac{mean(x_i^a - x_i^b)}{max(W_a, W_b)} + \frac{mean(y_i^a - y_i^b)}{max(H_a, H_b)} \tag{7.2}$$

Where $(x_i, y_i), i \in \{1, 2, 3, 4\}$ are the coordinates of the quadrilateral bounding box. S_{op}^{area} refers to the inverse intersection over union to capture the overlap between the two boxes:

$$S_{op}^{area} = 1 - \frac{Area(a \cap b)}{Area(a \cup b} \tag{7.3}$$

$S_{winkler}^{jaro}$ is the inverse Jaro Winkler Distance between the two strings. This metric is particularly informative, hence it is given the highest weight of 10.

$$S_{winkler}^{jaro} = 1 - JaroWinkler(str_a, str_b) \tag{7.4}$$

Last but not least, the distance between the two frames, or in other words the frame offset, is also an important characteristic to consider. To include it, I simply take the absolute difference between the head and the tail of the two streams.

$$S_{offset}^{frame} = abs(id_a - id_b) \tag{7.5}$$

The optimal match between the streams is calculated via the Hungarian Algorithm. If the final score S_{ab} is below the threshold of 8, then the streams will be combined and are regarded as one. The recognition of the stream which spans the most frames, will be regarded as the recognition for the new combined stream. This procedure will be continued to be done for $distance \in \{2{\times}rate, 3{\times}rate, ..., n{\times}rate\}$, where $(n{+}1){\times}rate >= \frac{threshold}{k_4}$. At last, HyText prunes text streams that appeared for fewer than 10 consecutive frames. This is done to remove unreliable text streams, and to prune those which appeared for such a small amount of time that the user would not be able to have memorized them.

7.3 Evaluation

The pruned sub-components from Section 6.3.4 will be used to materialize the HyText method. The method will be evaluated on five distinct videos using different rates. The insights here are threefold: Can HyText extract text in a sufficiently accurate way, can the rate be increased without major losses in accuracy, and can computationally extensive text detectors be used with high rates without majorly impeding on its accuracy?

7.3.1 Methods

The pruned sub-components from the evaluation results on still images from Section 6.3.4 will be used for the HyText evaluation. Specifically, this would be CRAFT and EAST for the detection and TRBA for the recognition. EAST is a very fast detector, and thus also prominently used in video text extractors such as YORO [Cheng et al., 2019]. Thus, this

method will be evaluated using low rates (1, 2, and 3). The CRAFT detector symbolizes a strong contrast to EAST. It is a lot slower, but also more accurate. Text detectors such as CRAFT are not conventional candidates for video-based STE due to their high inference times. However, since HyText can speed up inference times by selecting a high rate, CRAFT may still be a viable candidate. Moreover, we can tolerate high inference times because the extraction in the retrieval engine Cineast will only happen offline (see Section 2.3.2). Thus, the CRAFT detector will be evaluated using high rates (7, 8, and 9). If high accuracy can be achieved in spite of the high rates, than HyText would be deemed suitable for computationally expensive text detectors.

7.3.2 Evaluation Metric

The same evaluation metric used for STE in still images, described in Section 6.3.2, will also be used here. Namely, WRA to capture absolute matches, and TWRA to capture relative matches. To keep track of relative matches is especially important for this use case as vitrivr can recover strings that are not absolute matches. Moreover, the FPS rate will be recorded to put the results in perspective.

7.3.3 Datasets

(a) Urban scenario 1

(b) Urban scenario 2

(c) Urban scenario 3

(d) Car driving scenario

(e) Inside store scenario

Figure 7.3: Five videos from Text in Videos [Iwamura et al., 2021] chosen for HyText evaluation.

To evaluate HyText, a carefully chosen subset of the Text in Videos [Iwamura et al., 2021] dataset is used. The five videos comprise three distinct scenarios in which STE may be

of particular importance for retrieval. As mentioned in Section 1.1, it is very challenging to differentiate different stores merely by visual features. In this case, scene text retrieval does a much better job. Thus, three videos were chosen which depict such a scenario (see Figure 7.3a, 7.3b and 7.3c). Moreover, videos inside stores are also highly visually similar. However, labels on packages are very prominent and eye-catching, and thus may be a suitable candidate for scene text retrieval. For this reason, a video was chosen which was done inside a store (see Figure 7.3e). Last but not least, videos from roads are especially similar, with very little visual variety. However, street signs often appear on those roads, and even if the user does not remember what was written on the signs, he or she might remember where the car was heading and thus still be able to retrieve the video via scene text search. Thus, the video depicted in Figure 7.3d was selected for the evaluation.

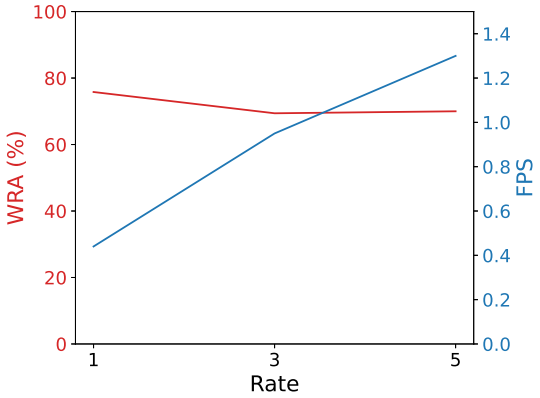
7.3.4 Results

Detector and rate	FPS	FPS rank	WRA	WRA rank	TWRA	TWRA rank
EAST and rate 1	0.44	4	75.8%	3	90.7%	**1**
EAST and rate 3	0.95	2	69.4%	6	87.0%	4
EAST and rate 5	1.3	**1**	70.0%	5	86.72%	5
CRAFT and rate 5	0.31	6	79.4%	**1**	89.75%	2
CRAFT and rate 7	0.4	5	72.8%	4	82.22%	6
CRAFT and rate 9	0.46	3	76.9%	2	87.4%	3

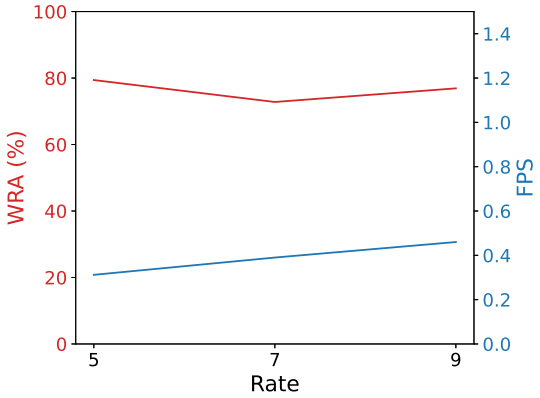
Table 7.1: Overview of HyText results using the EAST and CRAFT detector respectively.

As can be seen in all results illustrated in Figure 7.4, changing the rate can substantially affect inference time. When the rate is set as 1, the EAST detector has an FPS rate of 0.44. When the rate of 5 is used, the FPS rate increases to 1.3. Changing the rate for the CRAFT text detector has similar results. CRAFT alone has an FPS rate of 0.1 as shown in Section 4.3.4. The whole HyText pipeline using CRAFT with a rate of 5 reaches an FPS rate of 0.31 and for 9 one of 0.46. This is a significant increase in the FPS rate, and will have an especially noticeable effect when a large amount of multimedia content is processed.

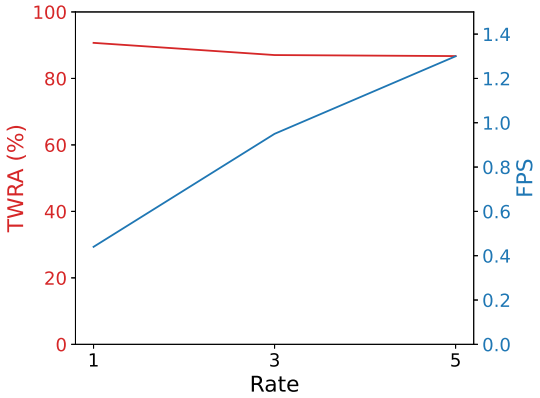
As expected, the accuracy for HyText decreases when a higher rate is used. However, the decrease is arguably negligible, especially compared to the speed up. HyText with EAST and a rate of 1 reaches a very high WRA-Hmean of 75.8%. When the rate of 5 is used, the WRA-Hmean is 70%. The TWRA-Hmean scores for EAST are even more impressive. For the rate of 1 a score of 90.7% is achieved, and one of 86.72% for the rate of 5. This means that when the user does not look for one-to-one matches, almost everything could be reliably retrieved without issues. The WRA-Hmean results for CRAFT are, as expected, even higher than the one for EAST. CRAFT got a WRA-Hmean score of 79.5% for the rate of 5 and a score of 76.9% with the rate of 9. Surprisingly, the



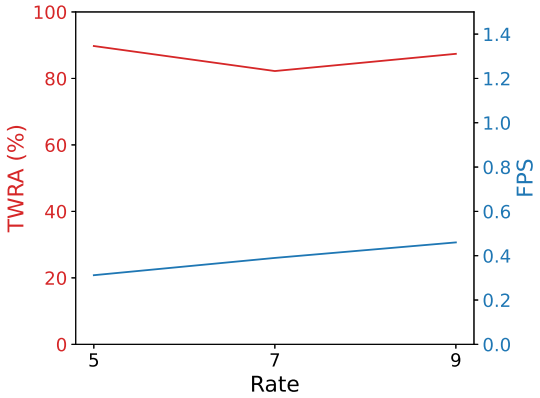
(a) WRA results with the EAST detector



(b) WRA results with the CRAFT detector



(c) TWRA results with the EAST detector



(d) TWRA with the CRAFT detector

Figure 7.4: WRA-Hmean, TWRA-Hmean and FPS results for HyText using the detectors EAST and CRAFT respectively.

WRA-Hmean score for the rate of 7 is 4.1% lower than for the rate of 9. This is a pattern echoed through all the videos evaluated, with the exception of the video showing the inside of a store (see Figure 7.3e). Another interesting observation is that while the WRA-Hmean score is higher for HyText using CRAFT in comparison to EAST, the TWRA-Hmean is actually minusculely lower. For a complete overview of the results, please have a look at Table 7.1.

All in all, HyText achieves state-of-the-art performance, and allows for substantially better inference times whilst only minusculely affecting accuracy. Moreover, HyText has shown that computationally costly detection methods such as CRAFT can be used for STE in video by increasing the rate.

8

Implementation

The STE methods for extracting text in still images, as well as the newly introduced HyText method, have shown to be of adequate accuracy and speed to be candidates for vitrivr. For this reason, this section deals with the implementation of these methods in vitrivr, and more specifically in the retrieval engine Cineast.

Since the original methods were written in Python, which is different from the language used in Cineast which is Java, Section 8.1 will deal with the implementation details regarding this transformation. Moreover, HyText will be evaluated again in Section 8.1, to see whether or not the sub-components (STD and STR), could be ported to Java without losses in accuracy and inference time.

8.1 Details

State-of-the-art STD and STR methods are almost exclusively written in Python, with a heavy bias towards the Pytorch package. However, some methods are also implemented in Tensorflow such as the EAST text detector [argman, 2021]. The recognition module, TRBA [Media-Smart, 2021], and the STD method CRAFT [ClovaAI, 2021a] are however written in Pytorch, which is a package not available for Java. Since HyText using EAST is particularly fast and has shown high accuracy, especially in terms of TWRA, I choose to implement this version of HyText in Cineast.

8.1.1 STD

The post-processing step of the EAST text detector from the original Github repository [argman, 2021], which was also used for our previous evaluation, is highly reliant on the Numpy package, which is unfortunately not available for Java. For this reason, I rewrote the post-processing step by using OpenCV. Since OpenCV can read Tensorflow-based weight files, I did not have to transform the weight file to another type. However, the weight file was originally used for training, and thus contains a lot of metadata that is superfluous for production. Thus, I froze the file to prune redundant metadata, and to nicely package everything within one single file. Since OpenCV comes with a lot of helpful methods which facilitate post-processing, the post-processing could be neatly accomplished within a manageable amount of lines of code. Lastly, in order to speed up the detection module, I enabled batch processing.

8.1.2 STR

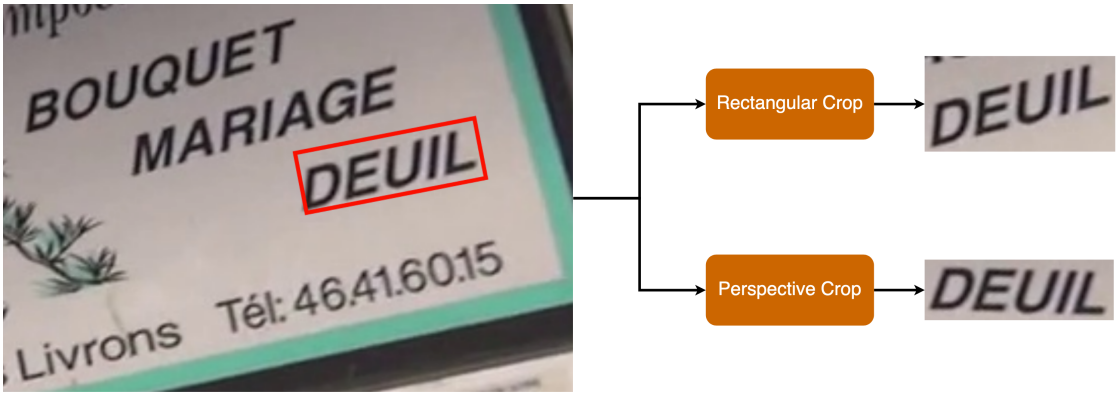


Figure 8.1: Difference between the rectangular cropping mechanism and the one using perspective transformation. Image taken from [Iwamura et al., 2021].

Since the recognition module TRBA was orginially written in Pytorch, which is unavailable for Java, I intend to write the method using OpenCV as I did with the STD module. However, as of the time of this writing OpenCV cannot read Pytorch-based weight files. For this reason, I need to convert the weight file to the ONNX type, which OpenCV can read. Unfortunately, the rectification step (TPS), and the attention mechanism used in TRBA contain loops, which ONNX does not support. Thus, I had to remove the rectification step and replace the attention mechanism with CTC-decoding. This change will have a few ramifications. For one, as discussed in Section 5.2.1, the attention-mechanism is slower, but more accurate than CTC. Moreover, the rectification step is there to align the text in such a way that it facilitates the recognition. Subsequently, the new method is expected to be faster, but also less accurate than TRBA.

Nevertheless, there is a way to mitigate the alignment problem which the rectification step previously solved. Instead of using rectangular cropping, perspective transformation can be applied to align the text horizontally and exclude superfluous surrounding regions. As can be seen in Figure 8.1, perspective cropping can align the non-horizontal text instance “DEUIL” horizontally, and simultaneously avoid including redundant neighboring regions, which in a worst-case scenario can also include other text instances.

Lastly, it is important to mention that the recognition module implemented in Cineast is configured in a way that it can easily be extended for any other CTC-based recognition module. For this reason, the weight file for the CRNN recognition method was also added to Cineast in case that the STE has to be particularly fast. The recognition module works with CRNN without any changes to the underlying code.

8.1.3 Tracker

In the evaluation of HyText, the tracker CSR-DCF was used. OpenCV does provide this tracker, and the OpenCV version of CSR-DCF was also used for the evaluation.

However, this tracker is not yet available for Java. Instead, the tracker provided by the package BoofCV called "Circulant" is used. The method was created by Henriques et al. [Henriques et al., 2012] and uses the theory of Discrete Fourier Transform (DCF), Circulant matrices, and linear classifiers to reliably track an object and learn its changes in appearance.

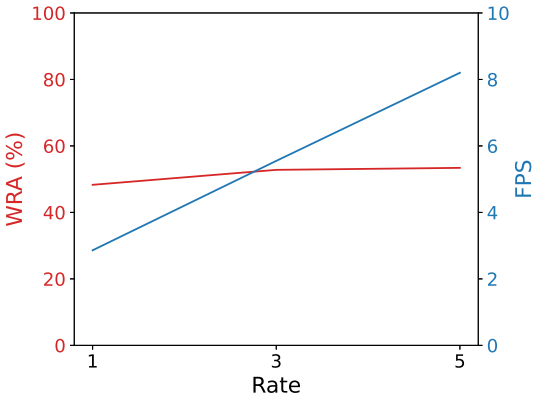
8.2 Evaluation

Detector and rate	FPS	FPS rank	WRA	WRA rank	TWRA	TWRA rank
Python version and rate 1	0.44	6	75.8%	**1**	90.7%	**1**
Python version and rate 3	0.95	5	69.4%	3	87.0%	2
Python version and rate 5	1.3	4	70.0%	2	86.72%	3
Vitrivr version and rate 1	2.86	3	48.4%	6	79.0%	6
Vitrivr version and rate 3	5.55	2	52.8%	5	82.1%	5
Vitrivr version and rate 5	8.2	**1**	53.6%	4	83.3%	4

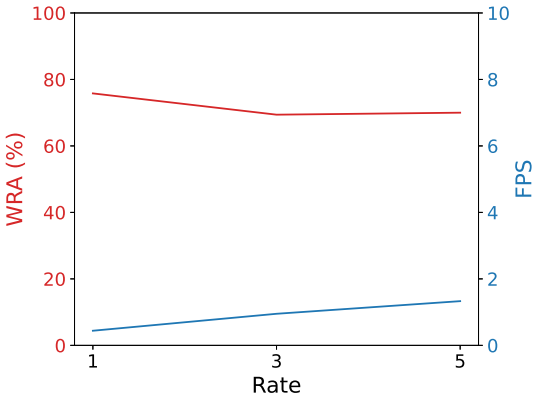
Table 8.1: Overview of HyText results using the EAST and CRAFT detector respectively.

In order to evaluate the HyText version implemented in vitrivr, the STE will be analyzed exactly the same way as HyText was in Section 7.3. However, to more closely resemble the usage in vitrivr, the extraction of the videos was not performed in a sequential manner. Instead, all videos were provided to Cineast at once to take advantage of the concurrency that Cineast enables. Moreover, the batch size was set to 16 frames.

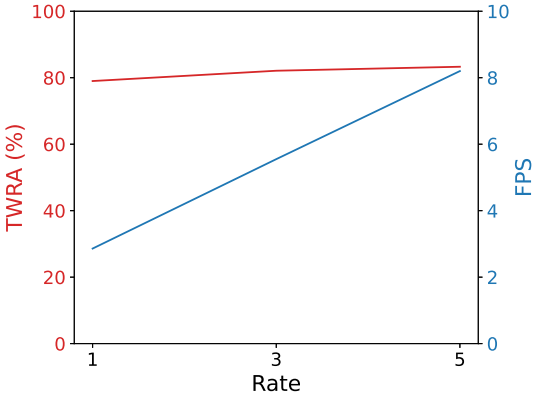
As can be seen in Table 8.1 and in Figure 8.2, the vitrivr-based version is significantly faster than its Python-based counterpart. The vitrivr-based version reaches an FPS score of 8.2 with a rate of 5, whereas the Python-based version has one of 1.3 for the same rate. This may be the result of the faster recognition module used, and of the concurrency provided by Cineast. Unfortunately, the WRA scores are noticeably below the Python-based version. However, the TWRA scores, which measure the similarity between the predicted and target string, are relatively close. This may indicate that the detection, tracking, and association are done as well as the Python-based version, but that the recognition of the text instances is impaired. There are several potential reasons for this impairment. The most obvious one is that the recognition module used in vitrivr is less accurate than TRBA, as already stated in Section 8.1.2. However, it could also be that the detection module and/or the Circulant tracker do not tightly envelop the target, which can result in cut text or the inclusion of neighboring text. Another interesting observation is that increasing the rate did not lead to a decline in WRA or TWRA, but in a noticeable increase. Given that an increase in the rate did not negatively affect accuracy, but in fact, increased it, may show that the recognition is not impaired by the tracker, but by the detection module. Since the same weight file was used for the Python-based detector and for the vitrivr-based one, the difference in accuracy can only be explained by the post-processing.



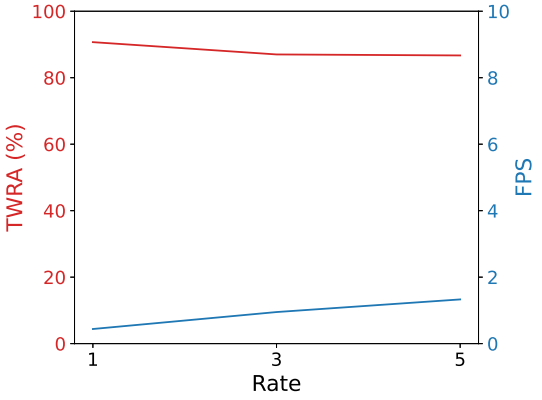
(a) WRA results with the vitrivr-based HyText version.



(b) WRA results with the Python-based HyText version.



(c) TWRA results with the vitrivr-based HyText version.



(d) TWRA results with Python-based HyText version.

Figure 8.2: WRA-Hmean, TWRA-Hmean and FPS results comparison between the Python-based HyText version and the one implemented in vitrivr.

9

Conclusions

9.1 Summary

As multimedia collections grow progressively larger, prior annotation becomes increasingly unfeasible, and raises the need for content-based rather than tag-based retrieval. Vitrivr is a content-based retrieval stack, and provides searches such as QbE and QbS to retrieve objects based on their visual content [Rossetto et al., 2016]. However, these queries are unsuitable for certain retrieval scenarios. These scenarios can typically be classified as cases in which two or more meaningfully different objects are visually close to indistinguishable. As mentioned in Section 1.1, a prime example of that is the retrieval of shops of a particular kind. A coffee shop, and a tea shop are for example hardly distinguishable without referring to storefront signs. However, providing access to the textual content on the signs allows for explicit and superior retrieval. For this reason, this thesis extended the retrieval engine Cineast with the ability to extract and retrieve scene text.

Since the final STE pipeline is a separately trained one rather than a jointly trained one, the sub-components, STD and STR, were first explored and evaluated separately. STD is a very distinguished type of detection task. In comparison to generic objects, scene text can have a wide range of aspect ratios due to the variety of text lengths and its multi-oriented appearance in natural images. For this reason, object-detection based STD methods had to design anchors of diverse aspect ratios, scales, and orientations to better capture text. Segmentation-based methods are more suitable for the detection of text with arbitrary shapes, however, they have difficulties distinguishing closely adjacent text instances, and require complex post-processing. Lastly, it is important to mention that text detections are done for the sake of later recognition, and thus it is particularly important to tightly capture text. For this reason, the TIoU metric [Liu et al., 2019] was used which takes both compactness and completeness into consideration. From the evaluation, the CRAFT [Baek et al., 2019] and the EAST [Zhou et al., 2017] detector were found to be the most promising, with CRAFT having a particular aptitude for accuracy and EAST for speed. In the investigation and evaluation of STR methods, it was found that segmentation-free methods have dominated the field of STR due to their ability to capture inter-contextual dependencies. A particular difficulty of STR is to recognize multi-oriented text. In order to overcome this, novel STR methods have begun

using rectification networks such as TPS [Shi et al., 2019], while others have focused on changing the one-dimensional decoding system to a two-dimensional one such as Wan et al. [Wan et al., 2019] and Lee et al. [Lee et al., 2019]. In the evaluation, the STR method TRBA and TRBC were found to be the most suitable. The sub-components were matched in Section 6.3, and CRAFT-TRBA and EAST-TRBA were found to be superior to the others in regards to accuracy and inference time.

Last but not least, a novel STE method for videos called HyText was developed. Previous methods have relied on detecting and recognizing text in each individual frame. However, as pointed out by Cheng et al. [Cheng et al., 2019], performing STE in each frame is too computationally expensive. In response, they recognize text only in a subset of bounding boxes. However, in contrast to Cheng et al., I argue that the main driver in computational cost is not the recognition, but the detection module. Moreover, the method neglects the main deteriorating factor of recognizing text in videos: Progressively disappearing text. For this reason, HyText was created which reduces computational cost substantially by only detecting text in a subset of frames, and increases accuracy by pruning bounding boxes which cut text. The evaluation of HyText showed state-of-the-art performance, however, the method could not be ported to Cineast without some losses in accuracy.

9.2 Future Work

The thesis presented has successfully extended the existing vitrivr with the ability to extract and retrieve scene text from visual multimedia. However, STE as it relates to content-based retrieval, and in particular to vitrivr, is far from a finished topic. Thus, this section aims at illuminating three domains that may serve as inspiration for future research.

9.2.1 STE Performance Enhancement

As already mentioned in Section 8, the STE method implemented in vitrivr did not achieve the same accuracy as the Python-based one. A contributor to this loss in accuracy is most likely the recognition module, which does not include a rectification step, nor does it apply the attention mechanism, which are both known to increase accuracy under the presence of multi-oriented text. While both steps cannot currently be used in vitrivr due to the use of loops which the ONNX type does not support, one could potentially implement the method developed by Wan et al. [Wan et al., 2019], which uses two-dimensional CTC to overcome the difficulty of recognizing irregular text. Moreover, I have postulated that the EAST detector in vitrivr might be inferior to the Python-based one due to differences in the post-processing. While the Python-based EAST version is highly reliant on the Numpy package, which is not available for Java, the Deep Java Library package [Amazon, 2021] could be used instead which provides many well-performing matrix operations.

9.2.2 Prominence-Enhanced Similarity Search

In the current version, the similarity between the textual query and the target is computed with the help of the normalized edit distance. However, while string similarity is necessarily needed to retrieve relevant media based on scene text, it does not take into account the prominence of the text. For example, let us say that a user wants to retrieve a video segment of an Adidas advertisement by providing the query string "Adidas". The STE might have been successful and extracted the string "Adidas" from the advertisement. However, the STE method might also have extracted the text from a completely different video where a person happens to have Adidas shoes on. Vitrivr would regard these two video segments as equally similar to the query string, even though the query is significantly more descriptive of the Adidas advertisement than for the other video. Thus, the similarity between the query string and the extracted text could be extended by taking into account the length at which a string appeared in the video and how big the bounding box is relative to the size of the frame.

9.2.3 Visuo-Textual Interplay

(a) Target image (b) QbS extended with text

Figure 9.1: Illustration of target image and query, where QbS has been enhanced with the ability to add text to capture the visuo-textual interplay.

Scene text queries and visual queries are currently regarded as separate query terms. However, textual content is embedded in visual scenes, and thus there naturally is a dynamic between the viusal and textual domain. However, this interplay is currently not captured. QbS could for example be extended with the ability to add text, and to subsequently position and color it. Thus, instead of the user providing a sketch and a text query separately, the two tasks can be combined to model the interplay between scene text and the scene itself and thus capture features such as text positioning and color which would otherwise be impossible to include.

References

[Amazon, 2021] Amazon (2021). Deep java library. *https://djl.ai/*.

[Apache, 2021] Apache (2021). Apache lucene. *https://lucene.apache.org/*.

[argman, 2021] argman (2021). East. *https://github.com/argman/EAST*.

[Baek et al., 2019] Baek, J., Kim, G., Lee, J., Park, S., Han, D., Yun, S., Oh, S. J., and Lee, H. (2019). What Is Wrong With Scene Text Recognition Model Comparisons? Dataset and Model Analysis. *International Conference on Computer Vision (ICCV)*, pages 715–4723.

[Baek et al., 2019] Baek, Y., Lee, B., Han, D., Yun, S., and Lee, H. (2019). Character region awareness for text detection. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9357–9366.

[Bahdanau et al., 2014] Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural Machine Translation by Jointly Learning to Align and Translate. *In Proceedings of ICLR*.

[Bartz et al., 2017] Bartz, C., Yang, H., and Meinel, C. (2017). See: Towards semi-supervised end-to-end scene text recognition.

[Borisyuk et al., 2018] Borisyuk, F., Gordo, A., and Sivakumar, V. (2018). Rosetta: Large scale system for text detection and recognition in images. *the 24th ACM SIGKDD International Conference*.

[Busta et al., 2017] Busta, M., Neumann, L., and Matas, J. (2017). Deep textspotter: An end-to-end trainable scene text localization and recognition framework. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2223–2231.

[Cai et al., 2021] Cai, H., Sun, J., and Xiong, Y. (2021). Revisiting classification perspective on scene text recognition.

[Chen et al., 2020] Chen, X., Jin, L., Zhu, Y., Luo, C., and Wang, T. (2020). Text recognition in the wild: A survey. *ArXiv*.

[Chen et al., 2004] Chen, X., Yang, J., Zhang, J., and Waibel, A. (2004). Automatic detection and recognition of signs from natural scenes. *IEEE Transactions on Image Processing*, 13(1):87–99.

[Cheng et al., 2019] Cheng, Z., Lu, J., Xie, J., Niu, Y., Pu, S., and Wu, F. (2019). Efficient video scene text spotting: Unifying detection, tracking, and recognition. *CoRR*, abs/1903.03299.

[Cheng et al., 2018] Cheng, Z., Xu, Y., Bai, F., Niu, Y., Pu, S., and Zhou, S. (2018). Aon: Towards arbitrarily-oriented text recognition. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5571–5579.

[ClovaAI, 2021a] ClovaAI (2021a). Craft. *https://github.com/clovaai/CRAFT-pytorch*.

[ClovaAI, 2021b] ClovaAI (2021b). What is wrong with scene text recognition model comparisons? dataset and model analysis. *https://github.com/clovaai/deep-text-recognition-benchmark*.

[Deng et al., 2018] Deng, D., Liu, H., Li, X., and Cai, D. (2018). Pixellink: Detecting scene text via instance segmentation. *CoRR*.

[Deng et al., 2019] Deng, L., Gong, Y., Lu, X., Lin, Y., Ma, Z., and Xie, M. (2019). Stela: A real-time scene text detector with learned anchor. *IEEE Access*.

[Du et al., 2020] Du, Y., Li, C., Guo, R., Yin, X., Liu, W., Zhou, J., Bai, Y., Yu, Z., Yang, Y., Dang, Q., and Wang, H. (2020). Pp-ocr: A practical ultra lightweight ocr system. *ArXiv*.

[Epshtein et al., 2010] Epshtein, B., Wexler, Y., and Ofek, E. (2010). Stroke width transform. *In Proceedings of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, page 2963–2970.

[Etsy, 2021] Etsy (2021). Personalized street sign, street sign, street signs. *https://www.etsy.com/listing/717873351/personalized-street-sign-street-sign*.

[Everingham et al., 2014] Everingham, M., Eslami, S., Gool, L., Williams, C. K., Winn, J., and Zisserman, A. (2014). The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111:98–136.

[Feng et al., 2019] Feng, W., He, W., Yin, F., Zhang, X.-Y., and Liu, C.-L. (2019). Textdragon: An end-to-end framework for arbitrary shaped text spotting. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9075–9084.

[Foursquare-City-Guide, 2021] Foursquare-City-Guide (2021). Tea shop montcada. *https://de.foursquare.com/v/tea-shop-montcada/4dc050ad6a2374fbdd929e39/photos*.

[Galerie, 2021] Galerie, M. (2021). Fresh tea: Life is too short for bad tea! *https://www.muellergalerie.de/shoppen-geniessen/the-fresh-tea-shop*.

[Gasser et al., 2019] Gasser, R., Rossetto, L., and Schuldt, H. (2019). Towards an all-purpose content-based multimedia information retrieval system. *CoRR*, abs/1902.03878.

[Gómez et al., 2018] Gómez, L., Mafla, A., Rusiñol, M., and Karatzas, D. (2018). Single shot scene text retrieval. *CoRR*, abs/1808.09044.

[Graves et al., 2006] Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. (2006). Connectionist temporal classifica- tion: labelling unsegmented sequence data with recurrent neural networks. *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006)*.

[Graves et al., 2009] Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., and Schmidhuber, J. (2009). A novel connectionist system for unconstrained handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):855–868.

[Gupta et al., 2016] Gupta, A., Vedaldi, A., and Zisserman, A. (2016). Synthetic data for text localisation in natural images. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2315–2324.

[He et al., 2015] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep Residual Learning for Image Recognition. *International Conference on Computer Vision (ICCV)*.

[Henriques et al., 2012] Henriques, J. F., Caseiro, R., Martins, P., and Batista, J. (2012). Exploiting the circulant structure of tracking-by-detection with kernels. In Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., and Schmid, C., editors, *Computer Vision – ECCV 2012*, pages 702–715, Berlin, Heidelberg. Springer Berlin Heidelberg.

[Iwamura et al., 2021] Iwamura, M., Gomez i Bigorda, L., and Karatzas, D. (2021). Text in videos. *https://rrc.cvc.uab.es/*.

[Jaderberg et al., 2015] Jaderberg, M., Simonyan, K., Zisserman, A., and Kavukcuoglu, K. (2015). Spatial Transformer Networks. *Proceedings of the 28th International Conference on Neural Information Processing Systems*, 2:2017–2025.

[JaidedAI, 2021] JaidedAI (2021). Easyocr.

[Jiang et al., 2017] Jiang, Y., Zhu, X., Wang, X., Yang, S., Li, W., Wang, H., Fu, P., and Luo, Z. (2017). R2cnn: Rotational region cnn for orientation robust scene text detection.

[Jiao et al., 2019] Jiao, L., Zhang, F., Liu, F., Yang, S., Li, L., Feng, Z., and Qu, R. (2019). A survey of deep learning-based object detection. *IEEE Access*, 7:128837–128868.

[Karatzas et al., 2015] Karatzas, D., Gomez-Bigorda, L., Nicolaou, A., Ghosh, S., Bagdanov, A., Iwamura, M., Matas, J., Neumann, L., Chandrasekhar, V. R., Lu, S.,

Shafait, F., Uchida, S., and Valveny, E. (2015). Icdar 2015 competition on robust reading. *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 1156–1160.

[Lee and Osindero, 2016] Lee, C.-Y. and Osindero, S. (2016). Recursive Recurrent Nets with Attention Modeling for OCR in the Wild. *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2231–2239.

[Lee et al., 2019] Lee, J., Park, S., Baek, J., Oh, S. J., Kim, S., and Lee, H. (2019). On recognizing texts of arbitrary shapes with 2d self-attention. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 2326–2335.

[Li et al., 2019] Li, H., Wang, P., Shen, C., and Zhang, G. (2019). Show, attend and read: A simple and strong baseline for irregular text recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:8610–8617.

[Liao et al., 2018] Liao, M., Shi, B., and Bai, X. (2018). Textboxes++: A single-shot oriented scene text detector. *IEEE Transactions on Image Processing*, 27(8):3676–3690.

[Liao et al., 2016] Liao, M., Shi, B., Bai, X., Wang, X., and Liu, W. (2016). Textboxes: A fast text detector with a single deep neural network.

[Liao et al., 2020] Liao, M., Wan, Y., Yao, C., Chen, K., and Bai, X. (2020). Real-time scene text detection with differentiable binarization. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34:11474–11481.

[Liao et al., 2019] Liao, M., Zhang, J., Wan, Z., Xie, F., Liang, J., Lyu, P., Yao, C., and Bai, X. (2019). Scene Text Recognition from Two-Dimensional Perspective. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33:8714–8721.

[Lin et al., 2020] Lin, H., Yang, P., and Zhang, F. (2020). Review of scene text detection and recognition. *Archives of Computational Methods in Engineering*, 27.

[Lin et al., 2014] Lin, T., Maire, M., Belongie, S. J., Bourdev, L. D., Girshick, R. B., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. (2014). Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312.

[Litman et al., 2020] Litman, R., Anschel, O., Tsiper, S., Litman, R., Mazor, S., and Manmatha, R. (2020). SCATTER: Selective Context Attentional Scene Text Recognizer. *Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Liu et al., 2018] Liu, H., Jin, S., and Zhang, C. (2018). Connectionist Temporal Classification with Maximum Entropy Regularization. *In Proceedings of NeurIPS*, 31.

[Liu et al., 2016a] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., and Berg, A. (2016a). Ssd: Single shot multibox detector. *European Conference on Computer Vision*, pages 21–37.

[Liu et al., 2016b] Liu, W., Chen, C., Wong, K.-Y., Su, Z., and Han, J. (2016b). Star-net: A spatial attention residue network for scene text recognition.

[Liu and Jin, 2017] Liu, Y. and Jin, L. (2017). Deep matching prior network: Toward tighter multi-oriented text detection. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3454–3461.

[Liu et al., 2019] Liu, Y., Jin, L., Xie, Z., Luo, C., Zhang, S., and Xie, L. (2019). Tightness-aware evaluation protocol for scene text detection. *CoRR*.

[Lukezic et al., 2016] Lukezic, A., Vojír, T., Cehovin, L., Matas, J., and Kristan, M. (2016). Discriminative correlation filter with channel and spatial reliability. *CoRR*, abs/1611.08461.

[Luo et al., 2021] Luo, C., Lin, Q., Liu, Y., Jin, L., and Shen, C. (2021). Separating content from style using adversarial learning for recognizing text in the wild. *nternational Journal of Computer Vision*.

[Media-Smart, 2021] Media-Smart (2021). Vedastr. *https://github.com/Media-Smart/vedastr*.

[Mishra et al., 2012] Mishra, A., Alahari, K., and Jawahar, C. V. (2012). Scene text recognition using higher order language priors. In *BMVC*.

[Movshovitz-Attias et al., 2015] Movshovitz-Attias, Y., Yu, Q., Stumpe, M. C., Shet, V., Arnoud, S., and Yatziv, L. (2015). Ontological supervision for fine grained classification of street view storefronts. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1693–1702.

[Natura-Coffee and Tea, 2021] Natura-Coffee and Tea (2021). How you can open a coffee shop cafe. *https://naturacoffeeandtea.com/how-you-can-open-a-coffee-shop-cafe/*.

[Needleman and Wunsch, 1970] Needleman, S. B. and Wunsch, C. D. (1970). A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453.

[Nguyen et al., 2014] Nguyen, P., Wang, K., and Belongie, S. (2014). Video text detection and recognition: Dataset and benchmark. pages 776–783.

[Pan et al., 2011] Pan, Y.-F., Hou, X., and Liu, C.-L. (2011). A hybrid approach to detect and localize texts in natural scene images. *IEEE Transactions on Image Processing*, 20(3):800–813.

[Qin et al., 2019] Qin, S., Bissaco, A., Raptis, M., Fujii, Y., and Xiao, Y. (2019). Towards unconstrained end-to-end text spotting. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4703–4713.

[Redmon et al., 2016] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788.

[Ren et al., 2015] Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39.

[Rossetto et al., 2016] Rossetto, L., Giangreco, I., Tanase, C., and Schuldt, H. (2016). vitrivr: A flexible retrieval stack supporting multiple query modes for searching in multimedia collections. pages 1183–1186.

[Shi et al., 2017] Shi, B., Bai, X., and Belongie, S. (2017). Detecting oriented text in natural images by linking segments. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3482–3490.

[Shi et al., 2015] Shi, B., Bai, X., and Yao, C. (2015). An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

[Shi et al., 2016] Shi, B., Wang, X., Lyu, P., Yao, C., and Bai, X. (2016). Robust scene text recognition with automatic rectification. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4168–4176.

[Shi et al., 2019] Shi, B., Yang, M., Wang, X., Lyu, P., Yao, C., and Bai, X. (2019). ASTER: An Attentional Scene Text Recognizerwith Flexible Rectification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41:2035–2048.

[Simonyan and Zisserman, 2014] Simonyan, K. and Zisserman, A. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition. *Conference on Computer Vision and Pattern Recognition (CVPR)*.

[Sun et al., 2018] Sun, Y., Dawut, A., and Hamdulla, A. (2018). A review: Text detection in natural scene image. *2018 3rd International Conference on Smart City and Systems Engineering (ICSCSE)*, pages 826–829.

[Tripadvisor, 2021] Tripadvisor (2021). The ferry coffee shop. *https://www.tripadvisor.ch/Restaurant_Review-g528815-d14246260-Reviews-The_Ferry_Coffee_Shop-Shepperton_Surrey_England.html*.

[Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *ArXiv*.

[Wan et al., 2019] Wan, Z., Xie, F., Liu, Y., Bai, X., and Yao, C. (2019). 2D-CTC for Scene Text Recognition.

[Wang et al., 2021] Wang, H., Bai, X., Yang, M., Zhu, S., Wang, J., and Liu, W. (2021). Scene text retrieval via joint text detection and similarity learning. *CoRR*, abs/2104.01552.

[Wang et al., 2011] Wang, K., Babenko, B., and Belongie, S. (2011). End-to-end scene text recognition. *IEEE International Conference on Computer Vision*, pages 1457–1464.

[Wang et al., 2020] Wang, S., Liu, Y., He, Z., Wang, Y., and Tang, Z. (2020). A quadrilateral scene text detector with two-stage network architecture. *Pattern Recognition*, 102:107230.

[Wang et al., 2019] Wang, W., Xie, E., Li, X., Hou, W., Lu, T., Yu, G., and Shao, S. (2019). Shape robust text detection with progressive scale expansion network.

[Wang et al., 2017] Wang, X., Jiang, Y., Yang, S., Zhu, X., Li, W., Fu, P., Wang, H., and Luo, Z. (2017). End-to-end scene text recognition in videos based on multi frame tracking. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, volume 01, pages 1255–1260.

[WHO, 2021] WHO (2021). Eye care, vision care, vision impairment and blindness. *https:// www.who.int/ health-topics/ blindness-and-vision-loss*.

[Xie et al., 2018] Xie, L., Ahmad, T., Jin, L., Liu, Y., and Zhang, S. (2018). A new cnn-based method for multi-directional car license plate detection. *IEEE Transactions on Intelligent Transportation Systems*, 19(2):507–517.

[Yang et al., 2019] Yang, M., Guan, Y., Liao, M., He, X., Bian, K., Bai, S., Yao, C., and Bai, X. (2019). Symmetry-Constrained Rectification Network for Scene Text Recognition. *International Conference on Computer Vision (ICCV)*, pages 9147–9156.

[Yang et al., 2019] Yang, M., Guan, Y., Liao, M., He, X., Bian, K., Bai, S., Yao, C., and Bai, X. (2019). Symmetry-constrained rectification network for scene text recognition. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9146–9155.

[Yang et al., 2020] Yang, P., Yang, G., Gong, X., Wu, P., Han, X., Wu, J., and Chen, C. (2020). Instance segmentation network with self-distillation for scene text detection. *IEEE Access*, 8:45825–45836.

[Yang et al., 2017] Yang, X., He, D., Zhou, Z., Kifer, D., and Giles, C. (2017). Learning to read irregular text with attention mechanisms. *Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 3280–3286.

[Ye et al., 2020] Ye, J., Chen, Z., Liu, J., and Du, B. (2020). Textfusenet: Scene text detection with richer fused features. pages 516–522.

[Yu et al., 2020] Yu, D., Li, X., Zhang, C., Han, J., Liu, J., and Ding, E. (2020). Towards accurate scene text recognition with semantic reasoning networks. *In Proceedings of CVPR*.

[Zhang et al., 2016] Zhang, Z., Chengquan, Z., Shen, W., and Yao, C. (2016). Multi-oriented text detection with fully convolutional networks. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4159–4167.

[Zhou et al., 2017] Zhou, X., Yao, C., Wen, H., Wang, Y., Zhou, S., He, W., and Liang, J. (2017). East: An efficient and accurate scene text detector. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2642–2651.

[ZJULearning, 2021] ZJULearning (2021). Pixellink. *https://github.com/ZJULearning/pixel_link*.

List of Figures

List of Tables