You Only Recognize Once: Towards Fast Video Text Spotting

Zhanzhan Cheng[12*], Jing Lu[2*], Yi Niu[2], Shiliang Pu[2], Fei Wu[1+], Shuigeng Zhou[3]
[1]Zhejiang University, Hangzhou, China
chengzhanzhan@hikvision.com,wufei@cs.zju.edu.cn
[2]Hikvision Research Institute, China
lujing6,niuyi,pushiliang@hikvision.com
[3]Fudan University, Shanghai, China
sgzhou@fudan.edu.cn

ABSTRACT

Video text spotting is still an important research topic due to its various real-applications. Previous approaches usually fall into the four-staged pipeline: text detection in individual images, frame-wisely recognizing localized text regions, tracking text streams and generating final results with complicated post-processing skills, which might suffer from the huge computational cost as well as the interferences of low-quality text. In this paper, we propose a fast and robust video text spotting framework by only recognizing the localized text one-time instead of frame-wisely recognition. Specifically, we first obtain text regions in videos with a well-designed spatial-temporal detector. Then we concentrate on developing a novel text recommender for selecting the highest-quality text from text streams and only recognizing the selected ones. Here, the recommender assembles text tracking, quality scoring and recognition into an end-to-end trainable module, which not only avoids the interferences from low-quality text but also dramatically speeds up the video text spotting process. In addition, we collect a larger scale video text dataset (LSVTD) for promoting the video text spotting community, which contains 100 text videos from 22 different real-life scenarios. Extensive experiments on two public benchmarks show that our method greatly speeds up the recognition process averagely by 71 times compared with the frame-wise manner, and also achieves the remarkable state-of-the-art.

CCS CONCEPTS

• **Computing methodologies** → **Object detection**; **Object recognition**; **Tracking**.

KEYWORDS

video text spotting, detection, tracking, quality scoring

---

[*]Both authors contributed equally to this research.
[+]Corresponding author (wufei@cs.zju.edu.cn).

---

1 INTRODUCTION

Video text spotting is still an important research topic due to the large amount of video text reading applications such as port container number identification in industrial monitoring and license plate recognition in intelligent transportation system.

In the past few years, several methods [29, 33, 53] are proposed to read video scene text with a multi-stage pipeline strategy: that is, firstly localizing text regions in individual frames, recognizing localized text regions one-by-one, tracking recognized regions as text streams, and applying post-precessing techniques for generating the final results. Figure 1.(a) shows such process.

However, above methods still suffer from two problems: 1) Excessive computational cost for directly reading text regions one-by-one, which may be impractical especially when working on front-end devices such as surveillance video cameras. 2) Much unstable recognition results due to the overwhelming of *low-quality* (*eg.* blurring, perspective distortion, rotation and poor illumination etc.) text. For example, the low-quality text region in 'stream 3' of Figure 1.(a) leads to the wrong result 'SERLUE'. In practical, it is unnecessary to recognize each text region in text streams, which will result in huge time cost and also bring various interferences of low-quality text.

To circumvent these problems, an idea is to select the *highest-quality* (*eg.* clear and horizontal) text region from each text stream to be recognized once instead of one-by-one, which needs us design a robust text quality scorer to assign a quality score for each candidate text region, as shown in Figure 1.(b). For further speeding up the process of video text spotting, we also attempt to assemble the text tracking, text quality scoring and recognition into an end-to-end trainable module (denoted by **text recommender**). The text recommender will benefit from the complementarity among tracking, scoring and recognition, and then improve the text spotting performance again.

In this paper, we propose a fast and robust video text spotting approach named as YORO by combining a well-designed spatial-temporal video text detector and a novel text recommender into a whole framework. Concretely, the spatial-temporal detector is designed to recall more text by referring to temporal relationship among different frames. The text recommender assembles the text tracking for generating text streams, the text quality scoring for assessing the quality of each text region and the recognition into an end-to-end trainable network, as shown in Figure 2 (b), in which the
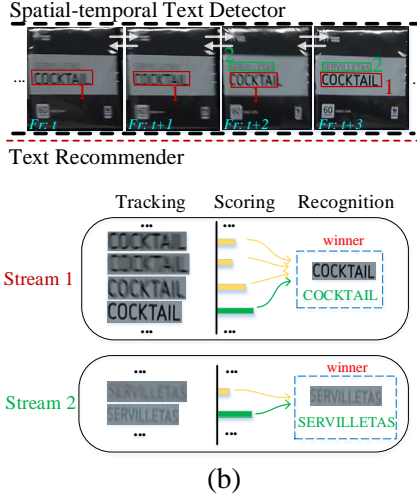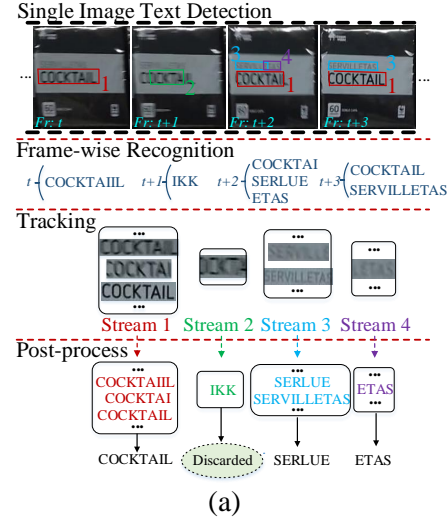
Single Image Text Detection

Frame-wise Recognition

Tracking

Stream 1   Stream 2   Stream 3   Stream 4

Post-process

(a)

Spatial-temporal Text Detector

Text Recommender

Tracking   Scoring   Recognition

Stream 1

Stream 2

(b)

Figure 1: Comparison of existing and proposed pipeline in video text spotting. (a) denotes the traditional framework which contains four steps: text detection in individual frames, frame-wisely recognizing text regions, tracking recognized text regions into streams and post-processing for obtaining final results. (b) denotes our proposed framework which only contains two steps: text detection among consecutive frames and recognizing the highest-quality text from text streams as the final results. *Fr*:(.) means the frame ID, and the numbers next to their corresponding boxes in each frame mean the tracked IDs. The white arrows in (b) represent the temporal relation built between consecutive frames in detection process. [Best viewed in color]

quality scores are learnt in a weakly-supervised manner (detailed in Methodology Section). In this way, we can ignore low-quality text regions and only reserve highest-quality text regions for recognition, which not only improves the recognition performance but also greatly decreases the computational cost.

Last but not least, we also note that the scales of existing video text benchmarks are very limited. For example, the largest video scene text dataset 'Text in Videos' [18] only contains 49 videos from 7 different scenarios, which may limit the video text recognition research progress. Therefore, we collect a **l**arger-**s**cale **v**ideo **t**ext **d**ataset (*abbr*. LSVTD) containing 100 videos from 22 natural scenarios, and hope to help the research of video text understanding.

Main contributions of this paper are as follows: (1) We design a novel text recommender for selecting the highest-quality text from text streams and then only recognizing the selected text regions once, which significantly speeds up the recognition process, and also improves the video text spotting performance. (2) We integrate a well-designed spatial-temporal text detector and a text recommender into an unified two-stage framework YORO for fast end-to-end video text spotting. (3) In order to promote the progress of video text spotting, we collect and annotate a larger scale video text dataset, which contains 100 videos from 22 different real-life situations. (4) Extensive experiments demonstrate that our method is fast and robust and achieves impressive performance in video scene text reading.

2 RELATED WORK

2.1 Text Reading in Single Images

Traditionally, the scene text reading system contains a text detector for localizing each text region and a text recognizer for generating corresponding character sequences. For text detection, numerous methods are proposed to localize regular and irregular (oriented and curved etc.) text regions, which can be categorized as anchor-based [15, 17, 24, 26, 27, 37] and direct-regression-based [13, 49, 60]. For text recognition, the task is now treated as a sequence recognition problem, in which CTC [9]-based [2, 38, 50] and attention-based [1, 4, 5, 39, 40] methods are designed and have achieved promising results. Recently, there are several methods [2, 12, 23, 25] attempting to spot text end-to-end.

In fact, lots of text reading applications actually work in video scenarios, in which scene text spotting from multiple frames may be more meaningful.

2.2 Text Reading in Videos

In recent years, only a few attention has been drawn to spotting video scene text in contrast to text reading in still images. For more details of text detection, tracking and recognition in video, the readers can refer to a comprehensive survey [56]. In general, reading text from scene videos can be roughly categorized into three major modules: 1) text detection, 2) text tracking, and 3) text recognition.

Text detection in videos. In early years (before 2012), most of methods focus on detecting text in each frame with connected component analysis [55] or sliding window [21] strategy. However, the performance of them is limited due to the low representation of handcrafted features. Though the recent detection techniques (mentioned in *Section.* 2.1) in still images can help improve feature representation, detecting text in scene videos is still challenging because of its complicated temporal characteristics (eg. motion). Therefore, text tracking strategies are introduced for enhancing the detection performance, which are further divided into two

categories [56]: spatial-temporal information based methods [8, 41, 44, 45] for reducing noise and fusion based methods [7, 30, 62] for improving detection accuracy. Recently, Wang et al. [52] employed optical flow based method to refine text locations in the subsequent frames.

Text tracking in videos. The traditional methods such as template matching [6, 31, 41, 44] and particle filtering were popular. But these methods failed to solve the re-initialization problem, especially in scene videos. Then the tracking-by-detection based methods [33–35] were developed to estimate the tracking trajectories and solve this problem.

Recently, Zuo et al. [62] and Tian et al. [46] attempted to fuse multi-tracking strategies (eg. spatial-temporal context learning [57], tracking-by-detection etc.) for text tracking, in which Hungarian [22] algorithm was applied for generating the final text streams. Yang et al. [54] also proposed a motion-based tracking approach in which detected results are directly propagated to the neighboring frames for recovering missing text regions. In fact, the robust feature extractor is the most important component for a text tracker.

Text recognition in videos. With the tracked text streams, there are two strategies for better scene text recognition: selection strategy by selecting the best text regions from streams (popular before 2010), and results fusion strategy by combining corresponding recognized character results. Correspondingly, methods [41, 44, 45] selected the region with the longest horizontal length as the most appropriate region. Then Goto and Tanaka [8] further enhanced the selection algorithm by taking six different features (eg. Fisher's discriminant ratio, text region area etc.) into account. While recent methods [10, 35] directly fused recognized results in text streams for final text prediction by majority voting, CRF or frame-wise comparison, and these approaches assumed that recognition results in most frames are trust-worthy, which may not be true in unconstrained scenarios. In addition, frame-wise text recognition also results in high computation cost.

End-to-end text recognition in videos. There are several works proposed to solve the end-to-end video text spotting problem. Nguyen et al. [33] first proposed an end-to-end video text reading solution by extending Wangs's method [51], in which the frame-wise detection and the tracking with multiple features (eg. the temporal distance, edit distance etc.) are applied. Merino-Gracia and Mirmehdi [29] proposed an end-to-end video scene text reading system by introducing the unscented Kalman filter [48], but mainly focused on large text found in outdoor environments. Recently, Wang et al. [53] firstly utilized an end-to-end deep neural network to detect and recognize text in each frame, and then employed the *tracking-by-detection* strategy to associate text regions, and recovered missed detections with the tracked results, finally obtain the recognition results by voting the most frequently appeared text strings.

Different from frame-wise recognition in [29, 33, 53], in this paper we propose a fast and robust video scene text spotting framework by integrating a spatial-temporal video text detector and a novel text recommender, which dramatically speeds up the text spotting process and also improves the final recognition performance.

3 METHODOLOGY

3.1 Overview

The architecture of YORO is shown in Figure 2, which consists of two modules: the spatial-temporal video text detector for detecting text regions among adjacent frames and the text recommender for recognizing the highest-quality text regions from the tracked text streams.

Concretely, the video text detector is responsible for recalling more text by referring to temporal relationship among different frames. The recommender is developed with three submodules: the essential quality scoring component for directly generating the quality score of each text region by borrowing the 'teacher-student' concept [14], the tracking component for generating text streams with metric learning techniques which can enhance and simplify the tracking process, and the recognition component for generating the final results with an attention-based decoder [39]. Note that, the recommender is implemented as an end-to-end trainable network.

3.2 Spatial-Temporal Detection

The text detection architecture is shown in Figure 2.(a), in which the backbone of EAST [60] is selected as our backbone. Here, we learn relations between consecutive frames with a *spatial-temporal aggregation* strategy for improving video text detection process, which can be divided into three steps: 1) enhancing temporal coherence between frames with a feature warping mechanism [61], 2) spatial matching between frames with a comparing and matching strategy inspired by [3, 43], and 3) temporal aggregation.

Formally, let I_t be the *t-th* frame in a video, the detection results in I_t can be refined with the detections of its consecutive frames $(I_{t-n}, ..., I_{t+n})$ where the size of refining window is $2n+1$.

Enhancing temporal coherence. We obtain the corresponding sequence of feature maps $F=(F_{t-n}, ..., F_{t+n})$ by propagating frames through the EAST backbone. Given a pair of frame features F_{t+i} and F_t (the reference frame), we enhance their temporal coherence by referring to the estimated flow $flow_{(t+i,t)}$ between I_{t+i} and I_t with a flow-guided warping mechanism

$$F_{t+i}^w = Warp(F_{t+i}, flow_{(t+i,t)}), \tag{1}$$

where $flow_{(t+i,t)}$ is pre-computed with TV-L1 algorithm, $Warp(.)$ is the bilinear warping function applied on each element in the feature maps, and F_{t+i}^w denotes the feature maps warped from frame I_{t+i} to frame I_t. Thus F is further transferred as the warped $F^w = (F_{t-n}^w, ..., F_{t+n}^w)$. Then we generate an enhanced sequence of *confidence maps* $C = (C_{t-n}, ..., C_{t+n})$ by propagating F^w into a classification sub-network, in which each value in C_{t+i} represents the possibility of being a text region.

Comparing and matching. We evaluate the spatial matching degree of two frames with matching weights. The weights are firstly computed with a transform module to produce the feature-aware filter which is represented as

$$F_{t+i}^{trans} = ReLU(BN(WF_{t+i}^w + b)), \tag{2}$$

where W and b are learnable parameters, BN and ReLU represent Batch Normalization and rectified linear unit function, respectively. Given the transformed feature maps, we compute the similarity
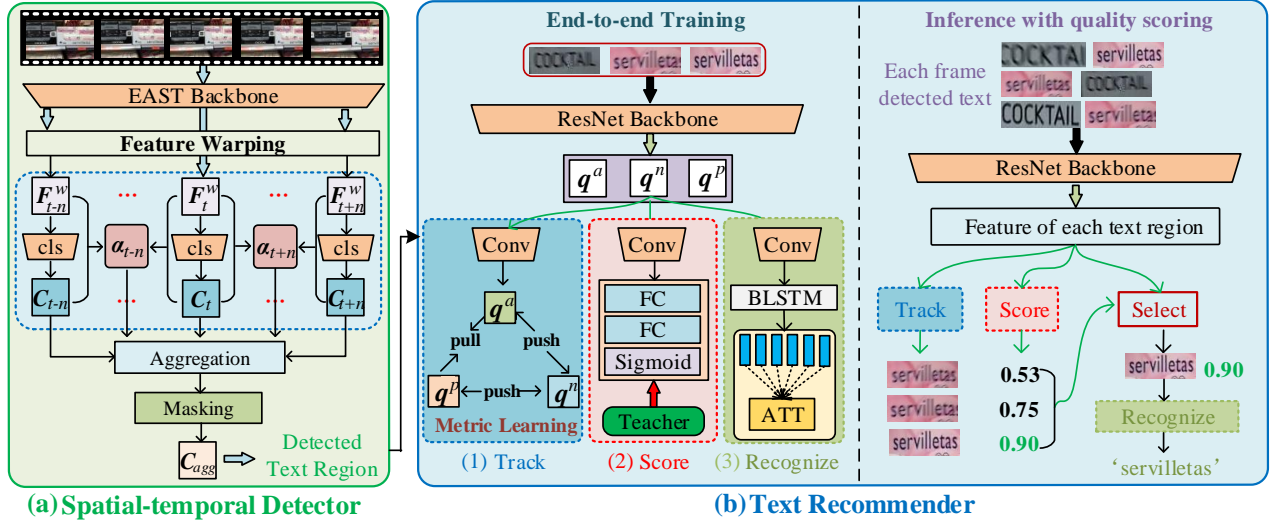
Figure 2: The workflow of YORO, which consists of two submodules: (a) The spatial-temporal text detector for generating text regions; (b) The text recommender assembling the tracking, quality scoring and recognition into a whole end-to-end trainable module, in which the left and right parts separately denote the training and inference processes. Specifically, the tracking, scoring and recognition are simultaneously learnt with the metric learning, the teacher-student strategy and the attention mechanism, respectively.

energy $Sim_{t+i,t} = F_{t+i}^{trans} \odot F_t^{trans}$ of I_{t+i} and I_t as the matching weights, where $\odot$ means the dot product position-wisely.

Temporal aggregation. Then we compute the aggregation weights by

$$a_{t+i} = \frac{exp(Sim_{t+i,t} \odot C_{t+i})}{\sum_{i'=-n}^{n} exp(Sim_{t+i',t} \odot C_{t+i'})}. \quad (3)$$

Here, we multiply $Sim_{t+i,t}$ by C_{t+i} in order to reinforce the aggregation weights of positive detections.

Finally, the temporal aggregation across the consecutive frames is computed by

$$C_{t,agg} = \sum_{i=-n}^{n} a_{t+i} * C_{t+i} \quad (4)$$

where "*" represents element-wise production.

For handling few mis-aggregated situations, we further refine $C_{t,agg}$ as $C_{t,ref} = C_{t,agg} * M_t$ by applying a normalized binary mask M_t to $C_{t,agg}$, where M_t is calculated by normalizing F_t^w as a binary mask with a pre-set threshold (default by 0.5). The detailed training process is the same to that in EAST [60].

3.3 Text Recommender

The text recommender contains three trainable parts: the quality scoring, tracking and recognition, as shown in Figure 2.(b).

3.3.1 Text Quality Scoring. It is almost impossible to manually annotate the quality score for each text region because the judgement of imaging quality is subjective for different annotators, and the annotating cost is also very tremendous. Therefore, we design an off-line scoring strategy (denoted by teacher) for generating the quality score labels in a weakly-supervised fashion, and then train the lightweight quality scoring network (denoted by student) with the teacher's scores.
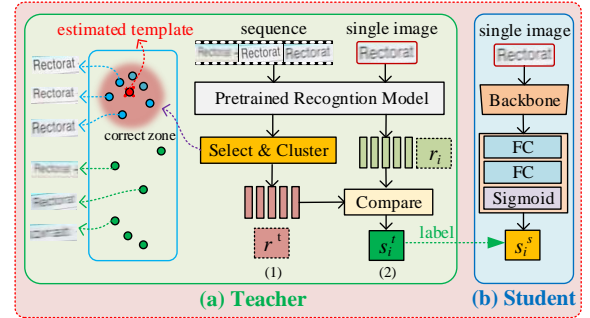
Figure 3: Training mechanism of quality scoring module. (a) Teacher: the off-line scoring mechanism to generate quality score labels by referring the estimate templates. Here, different quality of text regions are located at different positions in feature space, in which those always recognized correctly will fall into a 'correct zone' and its center corresponds to the estimated position of standard template. (b) Student: the quality scoring network.

Teacher. For teaching the student module, the teacher module should generate explicit quality score for each text region. Here, we propose the concept of *standard template*, which means that the template is the feature representation of an ideal text image without any interference. Then each text image is corresponding to a standard template image in the feature space, and we can define the distance between the template and a candidate text region as the corresponding quality score. In this assumption, those images

close to its template are more possible to be correctly recognized, vice versa.

However, it's unrealistic to directly calculate the distance between text regions in the low-level feature space (*eg.* color and intensity) because it is hard to directly align two text regions due to the difference on color, intensity and scales etc. Then, we attempt to represent each text region with the extracted high-level character features. For example, an image with content 'ABC' will be represented as the concatenation of high-level feature 'A', 'B' and 'C'. Here, the character features can be provided by the recognition model such as the output features of the attention-based decoder.

Specifically, we first feed a text stream into a pre-trained recognition model, and then represent each of them with the concatenated character features from the recognition model. We empirically find that features of correctly recognized text tend to be localized in a compact area while the features of wrongly recognized ones are usually far from this compact region, as shown in Figure 3. Therefore, we conduct the estimation of the template image r^t with the correctly recognized feature sets $\{r_1^{cor}, r_1^{cor}, ..., r_k^{cor}\}$ by applying the K-Means algorithm:

$$r^t = kmeans(r_1^{cor}, r_1^{cor}, ..., r_k^{cor}), \tag{5}$$

Consequentially, we calculate the feature distance between each text region and its estimated template image as the quality score:

$$s_i^t = \frac{r^t \odot r_i}{||r^t|| * ||r_i||}, \tag{6}$$

where $\odot$ means the dot product and $||.||$ means the Euclidean distance.

Student. Supervised by the generated quality scores from teacher, the quality scoring network can directly regress the quality score for each text region, as illustrated in Figure 3.(b). Then the loss function for quality scoring network is as follows:

$$\mathcal{L}_S = \frac{1}{N} \sum_{i=0}^{N} ||s_i^t - s_i^s|| \tag{7}$$

where s^t is the quality score generated by teacher, and s^s is the regressed quality score by student.

3.3.2 Text tracking. The tracking task aims to group corresponding text regions into text streams, shown in the left part of Figure 2.(b). Intuitively, the tracker should have the ability to ensure that the features of a text region in one stream must remain closer distance to those in the same stream than others, which implies: 1) the features must be discriminative enough to tolerate various interferences in unconstrained scenes, and 2) the module may be better if trained with a good distance measure.

Robust feature extraction. Thanks to the studies in deep neural network and metric learning, we extract robust features for the tracker by applying metric learning technique. Concretely, we firstly select three regions from localized candidate regions as an image triplet (R^a, R^p, R^n), in which R^a and R^p are corresponding to the same text instance while R^n is randomly selected from other text instances. Secondly, an image triplet is fed into a deep CNN for generating its L2 Normalized high-level representation (q^a, q^p, q^n). The tracker is then trained with two metric learning loss: contrastive loss [11]

$\mathcal{L}_{contra}$ and triplet loss [36] $\mathcal{L}_{triplet}$, i.e.

$$\mathcal{L}_T = \mathcal{L}_{contra} + \lambda_t \mathcal{L}_{triplet}, \tag{8}$$

where λ_t is default by 1.

Text stream generation. With the trained tracking model, for a pair of candidate text regions (R^1, R^2), we calculate its matching cost by

$$MC(R^1, R^2) = \frac{1}{q^1 \odot q^2 + \epsilon}. \tag{9}$$

For avoiding division by zero error, ϵ is set as 10^{-7}. Then those pairs with MC larger than a threshold are considered as invalid matching pairs and filtered out. Finally, we employ Hungarian algorithm [22] to generate the text streams.

3.3.3 Text Recognition. The text recognition module is not our focus, and we select attention-based method as our decoder just like in [4, 39, 40].

3.3.4 End-to-end training. Since the above three submodules are complementary for extracting discriminative features, we jointly train all the three tasks with same backbone in an end-to-end trainable network, which is supervised by:

$$\mathcal{L} = \lambda_1 \mathcal{L}_T + \lambda_2 \mathcal{L}_S + \lambda_3 \mathcal{L}_R, \tag{10}$$

where $\mathcal{L}_R$ is the recognition part loss and λ_i (i=1,2,3) denotes the loss weight for different tasks. Settings are detailed in *Experiment* section.

4 THE LARGER-SCALE VIDEO TEXT DATASET

In recent years, research in video scene text still remains unpopular in contrast to its promising application prospect. The existing video scene text datasets such as *IC13* or *IC15* (detailed in Table 1) are limited on the scale of video items and scenarios, which may restrain research of video scene text spotting. Therefore, we collect and annotate a **l**arger-**s**cale **v**ideo **t**ext **d**ataset (LSVTD), which contains 100 scene videos acquired from 22 typical real-life scenarios.

Datasets	#scenarios	#videos	#frames	#instances	quality?
Merino [28]	4	–	–	–	
Minetto [30]	–	5	3599	8706	
IC13 [19]	7	28	15277	93934	✓
YVT [33]	–	30	13500	–	
IC15 [18]	7	49	27824	–	✓
LSVTD	**22**	**100**	**66700**	**569300**	✓

Table 1: Comparison with existing video text datasets. 'qual-ity?' means if each text region is labeled with quality score.

Dataset characteristics. LSVTD is detailed in Table 1, and mainly characterized by 1) Much larger scale, which is more than twice the scale of IC15. 2) More diversified scenarios. LSVTD covers a wide range of 13 indoor (eg. bookstore, shopping mall) and 9 outdoor (eg. highway, city road) scenarios, which is more than 3 times the diversity of IC15. The variety of scenarios challenges text spotting algorithms to achieve robust performance. 3) Multilingual text instances. LSVTD contains text with more than three kinds of languages (English, Spanish and Chinese etc.), which are further divided into 2 major categories: Latin and Non-Latin. We additionally label this attribute for the convenience of evaluation

for different algorithms. More details of LSVTD are shown in the supplementary materials[1].

Annotation details. Following the annotation details adopted by IC15, we annotate the following attributes for each text: 1) *Polygon coordinate* represents text location; 2) *ID* means the unique identification for each text among consecutive frames; That is, the same text in consecutive frames shares the same ID; 3) *Language* is categorized as Latin and Non-Latin as mentioned above; 4) *Quality* indicates the image quality of each text region, which is labelled as 'high','moderate' or 'low'. 5) *Transcripts* means text string for each text region.

These videos are collected by our 6 colleagues and range from 5 seconds to 1 minute. It took more than 3500 man-hours for annotating these frames. Here, we hope that the availability[1] of LSVTD may spur more interests in corresponding areas.

5 EXPERIMENTS

5.1 Evaluation Protocols

The evaluation protocols for text detection, tracking and recognition in videos have been declared in [56]. We only select several mainstream evaluation metrics in this paper.

Detection metrics. Following detection methods [17, 59, 60], *precision* (denoted by PRE), *recall* (denoted by REC) and *F-measure* are selected as the evaluation metrics.

Tracking metrics. The tracking metrics should maximize the sum of overlap between the tracking results and ground truth. In general, multiple object tracking precision (*abbr*. MOTP), multiple object tracking accuracy (*abbr*. MOTA), and the average tracking accuracy (*abbr*. ATA) are used to evaluate the performance of tracking.

Quality scoring metrics. Naturally, the better quality frames are selected, the higher recognition accuracy we will get. To evaluate the performance of the quality scoring mechanism, we first define the **q**uality **s**election **h**itting **r**ate(*abbr*. QSHR) for evaluating the selection accuracy

$$QSHR = \sum_{i=0}^{N} \frac{\bar{q}_i}{N}, \quad (11)$$

where N denotes the number of text streams, and $\bar{q}_i \in \{0, 1\}$. In the i-th text stream, $\bar{q}_i = 1$ means the region annotated with "high" is hit, 0 otherwise.

Based on the selection mechanism, we further define the **r**ate of **c**orrectly **r**ecognizing selected text regions (*abbr*. RCR) for evaluating sequence-level recognition accuracy

$$RCR = \sum_{i=0}^{N} \frac{\bar{a}_i}{N}, \quad (12)$$

where $\bar{a}_i \in \{0, 1\}$. In the i-th text stream, $\bar{a}_i = 1$ means the selected text region is correctly recognized, 0 otherwise.

End-to-end metrics. In previous methods, MOTP, MOTA and ATA are generally used in end-to-end evaluation, which evaluate performance in word-level recognition. That is, a predict word is considered as a true positive if its IoU over ground truth is larger than 0.5 and the word recognition is correct.

However, in our task, we just score and recognize a tracked text stream one-time. According to the selection-and-recognition strategy, we redefine the end-to-end metrics by considering two constrains: 1) The recognized result of selected region should match the corresponding text transcription. 2) The temporal location (frame ID) of selected region should fall into the interval between the annotated starting and ending frame. In addition, the selected candidate should have a spatial overlap ratio (default by over 0.5) with the annotated bounding box. Thus we define the sequence-level recall REC_s and precision (PRE_s)

$$REC_s = \frac{N_r}{N_g}, \; PRE_s = \frac{N_r}{N_d}, \quad (13)$$

for *constrain 1* and *constrain 2*, in which N_r, N_g and N_d separately denote the number of valid recalled streams, the number of total ground truth streams and the number of detected text streams. Correspondingly, the sequence-level F-score (*F-Score*) is denoted as

$$F\text{-}Score = \frac{2 * PRE_s * REC_s}{PRE_s + REC_s} \quad (14)$$

by simultaneously considering PRE_s and REC_s.

It's worth to note that we only match a given ground truth stream once, which also penalize the stream fragmentation problem occurred in text tracking. In all, the evaluation protocol measures the accuracy and efficiency to extract useful text information from videos.

5.2 Implementation Details

All of our work is built on the CAFFE framework.

Detection Network. The EAST backbone is pre-trained on the 'Incidental Scene Text' dataset [18] and 'COCO-Text' dataset [47] by following [25], and then the model is fine-tuned on corresponding video training set such as IC13 or IC15. Images are randomly cropped and resized to 512×512 and then fed into the network. In training stage, we set *batch-size=4* and learn the network by adopting 'Adam' with *learning rate=10^{-4}, decay rate=0.94* for every 10^4 iterations, in which text regions with short side less than 10 pixels are ignored during training. While in testing stage, we only conduct the single-scale testing. In the post-processing stage, we adopt NMS on predicted geometric shapes with *threshold=0.2*.

Text Recommender Network. The 'ResNet Backbone'+'Conv Blocks' used in tracking, scoring and recognition is adopted from an image encoder used in [4], and the 'BLSTM'+'ATT' module in scoring and recognition is an attention decoder used in [4, 39]. The network is pre-trained on the 8-million synthetic data [16] using 'Adadelta' by following [39], and further fine-tuned on IC13 or IC15 using SGD with the fixed learning rate of 10^{-4}. The loss weights λ_1, λ_2, λ_3 in Equation. 10 are all set to 1, and the margin α used in triplet loss is set to 0.8. In text tracking process, the threshold MC for filtering out invalid text pairs is set to 0.92.

5.3 Ablation Study

Effects of text recommender. In IC13 and IC15, text regions are annotated as 3 quality levels ('low', 'moderate' and 'high'). Those streams containing at least two types of quality annotations are treated as our testing dataset.

To evaluate the proposed text recommender (*abbr.* TR), we compare our method with two commonly used selection-and-recognition strategies: 1) Using the predicted confidence (the average probability of generating characters) of a word as the quality score (denoted by PCW). 2) Selecting the text region with the highest frequency of predicted results as the voted best one (denoted by HFP), which is similar to the *majority voting* strategy used in [53].

Methods	QSHR (IC13/IC15)	RCR (IC13/IC15)	FPS
PCW	74.55/75.83	66.06/66.32	4.52
HFP	75.32/76.34	68.30/68.56	
TR ($\mathcal{L}_S$)	77.89/79.69	68.89/69.41	
TR ($\mathcal{L}_S$+$\mathcal{L}_T$)	78.64/80.36	69.12/69.82	324.58
TR ($\mathcal{L}_S$+$\mathcal{L}_R$)	81.23/83.03	69.92/70.69	
TR ($\mathcal{L}$)	**81.74/83.29**	**70.18/70.95**	

Table 2: Effects of TR on IC13 and IC15 compared with other frame selection methods. (.) refers to the loss applied in TR.

Table. 2 shows the results. Compared to HFP, TR ($\mathcal{L}$) significantly improves the *QSHR* performance by 6.4% on IC13 and 7% on IC15, while improves the *RCR* performance by 1.9% on IC13 and 2.4% on IC15. Compared to the results of other settings (TR ($\mathcal{L}_S$), TR ($\mathcal{L}_S$+$\mathcal{L}_R$) and TR ($\mathcal{L}_S$+$\mathcal{L}_T$)), the end-to-end training strategy can further boosting the scoring performance. For simplified representation, we use TR as TR ($\mathcal{L}$) in the sequel.

Moreover, TR only needs recognize a text stream one-time, which can greatly decrease the computational cost. As shown in Table. 2, TR speeds up the recognition process averagely by 71 times compared with the frame-wise manner on two datasets.

Extreme testing for text recommender. It is worth noticing that TR can still select the best one when handling text streams with a large proportion of low-quality text regions, while the voting strategy becomes useless. It implies that TR is more robust in complex and heavily distorted video scenarios. Therefore, we conduct extreme testing on a constituted *low-quality text stream set* by discarding all streams containing more than 40% highest quality text regions on IC13 and IC15. We calculate the *QSHR* and *RCR* on this set by checking whether the highest quality of text is hit and whether the selected text is correctly recognized. Table 3 gives the results and demonstrates that TR is more robust in complex and low-quality video scenarios.

Methods	QSHR (IC13/IC15)	RCR (IC13/IC15)
PCW	41.73/45.66	59.78/60.62
HFP	39.37/41.73	58.96/60.06
TR	**51.18/54.33**	**66.14/67.71**

Table 3: Extreme testing of TR on IC13 and IC15 compared with other frame selection methods.

End-to-end evaluation. To analyze the contributions of above components, we conduct the end-to-end evaluation on the popular IC15 dataset, as shown in Table. 4.

For fair comparison, we here declare some baselines in our framework, i.e. For detection, we treat the video text detection without our spatial-temporal strategy as our detection baseline (denoted by D-BASE), and denote the spatial-temporal detection model as D-ST; While for text recommender, compared with the trainable TR, we treat TR ($\mathcal{L}_S$) as our baseline.

From Table. 4, we find that

- Comparing to the detection baseline, the spatial-temporal strategy can steadily help the end-to-end recognition.
- In the D-ST case, our end-to-end trainable TR greatly improves the end-to-end performance of PRE_s, REC_s and *F-score* by 3.4%, 5.7% and 4.6%, respectively.
- The proposed method (D-ST + TR) significantly outperform the D-BASE+TR ($\mathcal{L}_S$) by 6.6%.

Above results demonstrate the effectiveness of text recommender as well as the whole framework.

D-BASE	✓	✓		
D-ST			✓	✓
TR ($\mathcal{L}_S$)	✓		✓	
TR		✓		✓
PRE_s	69.91	72.84	64.88	68.28
REC_s	54.34	61.73	61.54	67.21
F-score	61.15	66.83	63.17	**67.74**

Table 4: The end-to-end evaluation on IC15.

5.4 Comparison with State-of-The-Arts

Comparison on detection. We only evaluate the detection results on IC13 because there are no results reported on IC15. From Table. 5, we find that the D-BASE already outperforms existing approaches by a large margin thanking to the robust EAST, but still suffers from low recall due to the complicated motion scenarios. As expected, the spatial-temporal strategy can significantly improves *REC* by 4% and *F-measure* by 1.3%, but with a 4.3% drop on *PRE*. Actually, boosting recall performance is more important when facing very low recall results, which generally results in the precision decreasing.

Methods	REC	PRE	F-measure
Khare et al. [20]	41.40	47.60	44.30
Zhao et al. [58]	47.02	46.30	46.65
Shivakumara [42]	53.71	51.15	50.67
Yin et al. [55]	54.73	48.62	51.56
Wang et al. [52]	51.74	58.34	54.45
D-BASE	56.21	**85.76**	67.91
D-ST	**60.23**	81.45	**69.25**

Table 5: Detection results on IC13.

Comparison on tracking. Table 6 shows the comparing results on IC13 and IC15.

Evaluation on IC13. T-BASE outperforms the reported results by a large margin 0.35 on ATA_D, 0.08 on $MOTP_D$ and 0.37 on $MOTA_D$.

TR can further separately improve the performance by 0.14 and 0.03 on ATA_D and $MOTA_D$, and maintain the $MOTP_D$ performance.

Evaluation on IC15. T-BASE also achieves comparable results with previous methods. Comparing to the best reported result, TR significantly improves the ATA_D and $MOTA_D$ by 0.12 and 0.03, but falls behind [54] on $MOTP_D$. However, [54] points out that ATA_D is the most important metric in IC15 because ATA_D measures the tracking performance over all the text.

Dataset	Methods	ATA_D	$MOTP_D$	$MOTA_D$
IC13	IC13's base [19]	0.00	0.63	-0.09
	TextSpotter [32]	0.12	0.67	0.27
	Nguyen et al. [33]	0.15	-	-
	T-BASE	0.50	0.75	0.64
	TR	**0.64**	**0.75**	**0.67**
IC15	Stradvision-1 [18]	0.32	0.71	0.48
	Deep2Text-I [18]	0.45	0.71	0.41
	Wang et al. [53]	0.56	0.70	0.57
	Yang et al. [54]	0.61	**0.79**	0.66
	T-BASE	0.53	0.76	0.65
	TR	**0.65**	0.76	**0.68**

Table 6: Tracking performance evaluation on IC13 and IC15. The suffix 'D' means tracking is applied for detection.

Comparison on end-to-end evaluation. Conventionally, we also place the frame-wise recognition results on IC15 by referring to the previous works, in which we just treat the recognition output of the selected text region as the final recognition result in each text stream. Table. 7 shows that this setting also achieves the state of the art.

Method	$MOTP_R$	$MOTA_R$	ATA_R
Stradvision [18]	0.69	0.57	0.29
Deep2Text [18]	0.62	0.35	0.19
Wang et al. [53]	0.70	0.69	0.60
Ours	**0.76**	**0.69**	**0.63**

Table 7: The traditional end-to-end evaluation on IC15. The suffix 'R' means tracking is applied for measuring recognition.

5.5 Challenges on LSVTD

We also evaluate the proposed framework on the collected LSVTD. Figure 4 shows the *F-score* performance on 22 different scenarios. In brief, we find that video text spotting in *books opening* and *metro station* is easier comparing to that in *bookstore* or *shopping mall*. While video text spotting in *city road*, *camera surveillance* and *train watching* is still very challenging due to extremely complex background and fierce shaking, which needs more efforts to boost method's performance. Overall, spotting text in outdoor is more challenging than that in indoor. More statistics and visualization results are shown in supplementary materials.
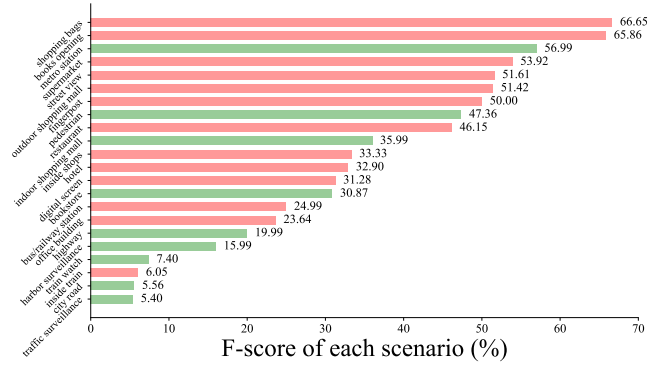

Figure 4: Illustration of *F-score* performance on 22 scenarios. Indoor/Outdoor scenarios are highlighted in red/green.

6 CONCLUSION

In this paper, we propose a fast and robust video text spotting framework named as YORO by integrating a well-designed spatial-temporal video text detector and a novel text recommender. The video text detector is responsible for recalling more text by referring the relation between different frames. Moreover, we concentrate on developing the text recommender for selecting the highest-quality text from tracked text streams and then only recognizing the selected text region once, which not only significantly speeds up the recognition process, and also improves the video text spotting performance. In addition, we release a larger-scale video scene text dataset for better evaluating video text spotting algorithms. In future, we'll further explore the spatial-temporal video text spotting in an end-to-end trainable way.

REFERENCES

[1] Fan Bai, Zhanzhan Cheng, Yi Niu, Shiliang Pu, and Shuigeng Zhou. 2018. Edit Probability for Scene Text Recognition. In *CVPR*. 1508–1516.
[2] Fedor Borisyuk, Albert Gordo, and Viswanath Sivakumar. 2018. Rosetta: Large Scale System for Text Detection and Recognition in Images. In *SIGKDD*. 71–79.
[3] Dapeng Chen, Hongsheng Li, Tong Xiao, Shuai Yi, and Xiaogang Wang. 2018. Video Person Re-Identification With Competitive Snippet-Similarity Aggregation and Co-Attentive Snippet Embedding. In *CVPR*. 1169–1178.
[4] Zhanzhan Cheng, Fan Bai, Yunlu Xu, Gang Zheng, Shiliang Pu, and Shuigeng Zhou. 2017. Focusing attention: Towards accurate text recognition in natural images. In *ICCV*. 5086–5094.
[5] Zhanzhan Cheng, Yangliu Xu, Fan Bai, Yi Niu, Shiliang Pu, and Shuigeng Zhou. 2018. AON: Towards Arbitrarily-Oriented Text Recognition. In *CVPR*. 5571–5579.
[6] Victor Fragoso, Steffen Gauglitz, Shane Zamora, Jim Kleban, and Matthew Turk. 2011. TranslatAR: A mobile augmented reality translator. In *WACV*. 497–502.
[7] Lluís Gómez and Dimosthenis Karatzas. 2014. MSER-based real-time text detection and tracking. In *ICPR*. 3110–3115.
[8] Hideaki Goto and Makoto Tanaka. 2009. Text-tracking wearable camera system for the blind. In *ICDAR*. 141–145.
[9] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist Temporal Classification : Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In *ICML*. 369–376.
[10] Jack Greenhalgh and Majid Mirmehdi. 2015. Recognizing Text-Based Traffic Signs. *IEEE TITS* 16, 3 (2015), 1360–1369.
[11] Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *CVPR*. 1735–1742.
[12] Tong He, Zhi Tian, Weilin Huang, Chunhua Shen, Yu Qiao, and Changming Sun. 2018. An End-to-End TextSpotter With Explicit Alignment and Attention. In *CVPR*. 5020–5029.
[13] Wenhao He, Xu-Yao Zhang, Fei Yin, and Cheng-Lin Liu. 2017. Deep Direct Regression for Multi-Oriented Scene Text Detection. In *ICCV*. 745–753.
[14] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network. (2015). arXiv:arXiv:1806.00578

[15] Han Hu, Chengquan Zhang, Yuxuan Luo, Yuzhuo Wang, Junyu Han, and Errui Ding. 2017. WordSup: Exploiting Word Annotations for Character Based Text Detection. In *ICCV*. 4940–4949.

[16] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Synthetic Data and Artificial Neural Networks for Natural Scene Text Recognition. (2014). arXiv:arXiv:1406.2227

[17] Yingying Jiang, Xiangyu Zhu, Xiaobing Wang, Shuli Yang, Wei Li, Hua Wang, Pei Fu, and Zhenbo Luo. 2017. R2CNN: Rotational Region CNN for Orientation Robust Scene Text Detection. (2017). arXiv:arXiv:1706.09579

[18] Dimosthenis Karatzas, Lluis Gomez-Bigorda, Anguelos Nicolaou, Suman Ghosh, Andrew Bagdanov, Masakazu Iwamura, Jiri Matas, Lukas Neumann, Vijay Ramaseshan Chandrasekhar, Shijian Lu, et al. 2015. ICDAR 2015 competition on robust reading. In *ICDAR*. 1156–1160.

[19] Dimosthenis Karatzas, Faisal Shafait, Seiichi Uchida, Masakazu Iwamura, Lluis Gomez i Bigorda, Sergi Robles Mestre, Joan Mas, David Fernandez Mota, Jon Almazan Almazan, and Lluis Pere De Las Heras. 2013. ICDAR 2013 robust reading competition. In *ICDAR*. 1484–1493.

[20] Vijeta Khare, Palaiahnakote Shivakumara, and Paramesran Raveendran. 2015. A new Histogram Oriented Moments descriptor for multi-oriented moving text detection in video. *Expert Systems with Applications* 42, 21 (2015), 7627–7640.

[21] Kwang In Kim, Keechul Jung, and Jin Hyung Kim. 2003. Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm. *IEEE TPAMI* 25, 12 (2003), 1631–1639.

[22] Harold W Kuhn. 1955. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly* 2, 1-2 (1955), 83–97.

[23] Hui Li, Peng Wang, and Chunhua Shen. 2017. Towards End-To-End Text Spotting With Convolutional Recurrent Neural Networks. In *ICCV*. 5238–5246.

[24] Minghui Liao, Zhen Zhu, Baoguang Shi, Gui-song Xia, and Xiang Bai. 2018. Rotation-Sensitive Regression for Oriented Scene Text Detection. In *CVPR*. 5909–5918.

[25] Xuebo Liu, Ding Liang, Shi Yan, Dagui Chen, Yu Qiao, and Junjie Yan. 2018. FOTS: Fast Oriented Text Spotting with a Unified Network. In *CVPR*. 5676–5685.

[26] Yuliang Liu and Lianwen Jin. 2017. Deep Matching Prior Network: Toward Tighter Multi-oriented Text Detection. In *CVPR*. 3454–3461.

[27] Jianqi Ma, Weiyuan Shao, Hao Ye, Li Wang, Hong Wang, Yingbin Zheng, and Xiangyang Xue. 2018. Arbitrary-oriented scene text detection via rotation proposals. *IEEE Transactions on Multimedia* (2018).

[28] Carlos Merino and Majid Mirmehdi. 2007. A framework towards realtime detection and tracking of text. In *CBDAR*. 10–17.

[29] Carlos Merino-Gracia and Majid Mirmehdi. 2014. Real-time text tracking in natural scenes. *IET Computer Vision* 8, 6 (2014), 670–681.

[30] Rodrigo Minetto, Nicolas Thome, Matthieu Cord, Neucimar J Leite, and Jorge Stolfi. 2011. Snoopertrack: Text detection and tracking for outdoor videos. In *ICIP*. 505–508.

[31] Yinan Na and Di Wen. 2010. An effective video text tracking algorithm based on sift feature and geometric constraint. In *PRCM*. 392–403.

[32] Luka Neumann and Jiri Matas. 2013. On combining multiple segmentations in scene text recognition. In *ICDAR*. 523–527.

[33] Phuc Xuan Nguyen, Kai Wang, and Serge Belongie. 2014. Video text detection and recognition: Dataset and benchmark. In *WACV*. 776–783.

[34] Marc Petter, Victor Fragoso, Matthew Turk, and Charles Baur. 2011. Automatic text detection for mobile augmented reality translation. In *Workshop on ICCV*. 48–55.

[35] Xuejian Rong, Chucai Yi, Xiaodong Yang, and Yingli Tian. 2014. Scene text recognition in multiple frames based on text tracking. In *ICME*. 1–6.

[36] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *CVPR*. 815–823.

[37] Baoguang Shi, Xiang Bai, and Serge Belongie. 2017. Detecting Oriented Text in Natural Images by Linking Segments. In *CVPR*. 2550–2558.

[38] Baoguang Shi, Xiang Bai, and Cong Yao. 2017. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE TPAMI* 39, 11 (2017), 2298–2304.

[39] B. Shi, X. Wang, P. Lyu, C. Yao, and X. Bai. 2016. Robust Scene Text Recognition with Automatic Rectification. In *CVPR*. 4168–4176.

[40] B. Shi, M. Yang, X. Wang, P. Lyu, C. Yao, and X. Bai. 2018. ASTER: An Attentional Scene Text Recognizer with Flexible Rectification. *IEEE TPAMI* (2018), 1–1.

[41] Hiroki Shiratori, Hideaki Goto, and Hiroaki Kobayashi. 2006. An efficient text capture method for moving robots using DCT feature and text tracking. In *ICPR*, Vol. 2. 1050–1053.

[42] Palaiahnakote Shivakumara, Rushi Padhuman Sreedhar, Trung Quy Phan, Shijian Lu, and Chew Lim Tan. 2012. Multioriented video scene text detection through Bayesian classification and boundary growing. *IEEE TCSVT* 22, 8 (2012), 1227–1235.

[43] Jianlou Si, Honggang Zhang, Chun-Guang Li, Jason Kuen, Xiangfei Kong, Alex C Kot, and Gang Wang. 2018. Dual Attention Matching Network for Context-Aware Feature Sequence based Person Re-Identification. In *CVPR*. 5363–5372.

[44] Makoto Tanaka and Hideaki Goto. 2007. Autonomous text capturing robot using improved DCT feature and text tracking. In *ICDAR*, Vol. 2. 1178–1182.

[45] Makoto Tanaka and Hideaki Goto. 2008. Text-tracking wearable camera system for visually-impaired people. In *ICPR*. 1–4.

[46] Shu Tian, Wei-Yi Pei, Ze-Yu Zuo, and Xu-Cheng Yin. 2016. Scene Text Detection in Video by Learning Locally and Globally. In *IJCAI*. 2647–2653.

[47] Andreas Veit, Tomas Matera, Lukas Neumann, Jiri Matas, and Serge Belongie. 2016. COCO-Text: Dataset and Benchmark for Text Detection and Recognition in Natural Images. (2016). arXiv:arXiv:1601.07140

[48] Eric A Wan and Rudolph Van Der Merwe. 2000. The unscented Kalman filter for nonlinear estimation. In *AS-SPCC*. 153–158.

[49] Fangfang Wang, Liming Zhao, Xi Li, Xinchao Wang, and Dacheng Tao. 2018. Geometry-Aware Scene Text Detection With Instance Transformation Network. In *CVPR*. 1381–1389.

[50] Jianfeng Wang and Xiaolin Hu. 2017. Gated recurrent convolution neural network for OCR. In *NIPS*. 335–344.

[51] Kai Wang, Boris Babenko, and Serge Belongie. 2011. End-to-end scene text recognition. In *ICCV*. IEEE, 1457–1464.

[52] Lan Wang, Yang Wang, Susu Shan, and Feng Su. 2018. Scene Text Detection and Tracking in Video with Background Cues. In *ICMR*. 160–168.

[53] Xiaobing Wang, Yingying Jiang, Shuli Yang, Xiangyu Zhu, Wei Li, Pei Fu, Hua Wang, and Zhenbo Luo. 2017. End-to-End Scene Text Recognition in Videos Based on Multi Frame Tracking. In *ICDAR*, Vol. 1. 1255–1260.

[54] Xue-Hang Yang, Wenhao He, Fei Yin, and Cheng-Lin Liu. 2017. A Unified Video Text Detection Method with Network Flow. In *ICDAR*, Vol. 1. 331–336.

[55] Xu-Cheng Yin, Xuwang Yin, Kaizhu Huang, and Hong-Wei Hao. 2014. Robust text detection in natural scene images. *IEEE TPAMI* 36, 5 (2014), 970–983.

[56] Xu-Cheng Yin, Ze-Yu Zuo, Shu Tian, and Cheng-Lin Liu. 2016. Text detection, tracking and recognition in video: a comprehensive survey. *IEEE TIP* 25, 6 (2016), 2752–2773.

[57] Kaihua Zhang, Lei Zhang, Qingshan Liu, David Zhang, and Ming-Hsuan Yang. 2014. Fast visual tracking via dense spatio-temporal context learning. In *ECCV*. 127–141.

[58] Xu Zhao, Kai-Hsiang Lin, Yun Fu, Yuxiao Hu, Yuncai Liu, and Thomas S Huang. 2011. Text from corners: a novel approach to detect text and caption in videos. *IEEE TIP* 20, 3 (2011), 790–799.

[59] Zhuoyao Zhong, Lei Sun, and Qiang Huo. 2018. An Anchor-Free Region Proposal Network for Faster R-CNN based Text Detection Approaches. (2018). arXiv:arXiv:1804.09003

[60] Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. 2017. EAST: An Efficient and Accurate Scene Text Detector. In *CVPR*. 5551–5560.

[61] Xizhou Zhu, Yujie Wang, Jifeng Dai, Lu Yuan, and Yichen Wei. 2017. Flow-Guided Feature Aggregation for Video Object Detection. In *ICCV*. 408–417.

[62] Ze-Yu Zuo, Shu Tian, Wei-yi Pei, and Xu-Cheng Yin. 2015. Multi-strategy tracking based text detection in scene videos. In *ICDAR*. 66–70.