arXiv:1903.03299v1 [cs.CV] 8 Mar 2019

Efficient Video Scene Text Spotting: Unifying Detection, Tracking, and Recognition

Zhanzhan Cheng[12*] Jing Lu[2*] Jianwen Xie[2*] Yi Niu[2] Shiliang Pu[2] Fei Wu[1†]
[1]Zhejiang University, China; [2]Hikvision Research Institute, China;
{chengzhanzhan;lujing6;jianwen.xie;niuyi;pushiliang}@hikvision.com; wufei@cs.zju.edu.cn

Abstract

This paper proposes an unified framework for efficiently spotting scene text in videos. The method localizes and tracks text in each frame, and recognizes each tracked text stream one-time. Specifically, we first train a spatial-temporal text detector for localizing text regions in the sequential frames. Secondly, a well-designed text tracker is trained for grouping the localized text regions into corresponding cropped text streams. To efficiently spot video text, we recognize each tracked text stream one-time with a text region quality scoring mechanism instead of identifying the cropped text regions one-by-one. Experiments on two public benchmarks demonstrate that our method achieves impressive performance.

1. Introduction

Natural text recognition is still a hot research topic due to its various real-world applications such as road sign recognition for advanced driver assistant system (ADAS) and license plate recognition for intelligent transportation system (ITS). Thus plenty of works, such as [1, 2, 4, 5, 14, 15, 17, 23, 24, 26, 25, 29, 40, 41, 42, 43, 52, 53, 63], have been proposed to spot text from a single image and achieved promising performance.

However, there are still a large number of text reading applications built on video scenarios (*e.g.* port container number identification in industrial monitoring, license plate recognition system in ITS etc.). Different from text reading from an individual image, reading text from videos has several major challenges: (1) Bad imaging quality due to uncontrollable video recording conditions (*e.g.* changeable illumination, various perspective, uncertain camera shaking etc.), may result in low-quality text such as blurring, perspective distortion, rotation and poor illumination. (2) Complicated matching processes for tracking in continuous video frames would significantly affect the final recognition results. (3) Excessive computational cost for directly reading text regions one-by-one may be impractical when working on front-end devices (*e.g.* surveillance video cameras).

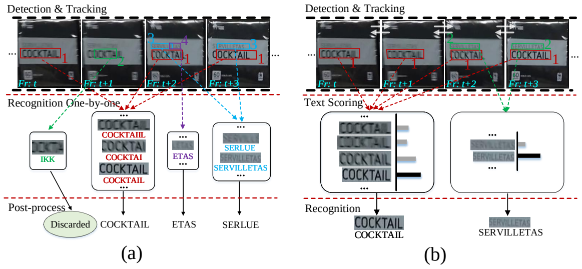


Figure 1. An illustration of pipelines in spotting video text. (a) and (b) separately display the traditional and our proposed framework. The white arrows means the relation built between consecutive frames in detection process. Fr:(.) means the frame ID, and the numbers neighbouring their corresponding boxes in each frame mean the tracked IDs. [Best viewed in color]

In the past few years, only a small number of methods [31, 36, 56] are proposed to read text from scene videos by a naive pipeline strategy. See Figure 1.(a) for a traditional framework for video scene text spotting, where text regions in the frames are localized in a frame-wise setting, sophisticated tracking strategies are adopted for grouping corresponding text regions into text streams, and all text region in a text stream are recognized one-by-one. Nonetheless, the conventional pipeline strategy is not able to cope with the raised challenges, and leads to unreliable performance and excessively time-consuming issue, which may be impractical in the real-world applications. For detection, treating all frames independently and directly learning a detection model from them may lose the important temporal relationship among different frames. Actually, a good detection results in a specific frame can be achieved by referring to the detection process in other frames. For example, the learning process of detecting *bounding box No.1* in the $(t+1)$-th frame can be further improved by considering

[*] Authors contribute equally.
[†] Corresponding author.

the spatial-temporal information in the t-th, $(t+2)$-th and $(t+3)$-th frames, as achieved in Figure 1.(b). For tracking, handcrafted features (*e.g*. SIFT [28] and HOG [6] etc.) are usually un-robust or un-discriminative when facing various interferences, especially when tracking similar text regions. For recognition, it is also unreasonable to recognize each text in the tracked text stream since 1) redundantly reading each text region is unnecessary and time-consuming for obtaining the final recognition results, 2) various interferences among different frames may yield completely unreadable text regions, thus further affect the final recognition results (*e.g*. the recognized 'SERLUE' in Figure 1.(a)).

In order to handle the above problems, we propose an efficient and unified framework named **SVST** for **s**potting **v**ideo **s**cene **t**ext, shown in Figure 1.(b). Firstly, we learn a **s**patial-temporal **v**ideo **t**ext **d**etector (*abbr*. SVTD) to detect text regions among consecutive frames. Secondly, we improve the tracking process of text regions with the discriminative features which are extracted from a well-designed deep **t**ext **r**e-**i**dentification **n**etwork (*abbr*. TRIN). Then, instead of recognizing each text region in a text stream, we develop a **t**ext **s**tream **s**coring **n**etwork (*abbr*. TSSN) for evaluating the quality of each text region, and select the text region with the highest quality score to be recognized. In this way, we can ignore low-quality text regions and only reserve selected text regions for recognition, which not only improves the recognition performance but also decreases the computational cost. Consequently, we adopt a common sequence decoder to output the final character sequences.

Main contributions of this paper are as follows: (1) We propose an unified framework for efficient video scene text spotting. (2) We learn spatial-temporal relations between adjacent frames for improving the video text detection stability. In addition, the designed TRIN enhances tracking process for generating text streams. (3) We design a novel text stream scoring network to evaluate the quality of cropped text regions in a text stream, and then the text region with the highest quality score is selected for the final recognition. (4) Extensive experiments show that our SVST 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, 29, 40] and direct-regression-based [14, 52, 63]. For text recognition, the task is now treated as a sequence recognition problem, in which CTC [10]-based [2, 41, 53] and attention-based [1, 4, 5, 42, 43] methods are designed and have achieved promising results. Recently, there are several methods [2, 13, 23, 25] attempting to spot text end-to-end.

In fact, lots of text reading applications actually work in scene 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 spotting in still images. For more details of text detection, tracking and recognition in video, the readers can refer to a comprehensive survey [59]. 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 [58] 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 image can help improve feature representation, detecting text in scene videos is still challenging because of its complicate temporal characteristics (*e.g*. motion). Therefore, text tracking strategies are introduced for enhancing the detection performance, which are further divided into two categories [59]: spatial-temporal information based methods [9, 44, 47, 48] for reducing noise and fusion based methods [8, 33, 65] for improving detection accuracy. Recently, Wang *et al*. [55] employed optical flow based method to refine text locations in the subsequent frames.

Text tracking in videos. The traditional methods such as template matching [7, 34, 44, 47] 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 [36, 37, 38] were developed to estimate the tracking trajectories and solve this problem.

Recently, Zuo *et al*. [65] and Tian *et al*. [49] attempted to fuse multi-tracking strategies (*e.g*. spatial-temporal context learning [60], tracking-by-detection etc.) for text tracking, in which Hungarian [22] algorithm was applied for generating the final text streams. Yang *et al*. [57] 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 [44, 47, 48] selected the region with the longest horizontal length as the most appropriate region. Then Goto and Tanaka [9] further enhanced the selection algorithm by taking six different features (*e.g.* Fisher's discriminant ratio, text region area etc.) into account. While recent methods [11, 38] 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.* [36] first proposed an end-to-end video text reading solution by extending Wangs's method [54], in which the frame-wise detection and the tracking with multiple features (*e.g.* the temporal distance, edit distance etc.) are applied. Merino-Gracia and Mirmehdi [31] proposed an end-to-end video scene text reading system by introducing the unscented Kalman filter [51], but focused on large text found in outdoor environments. Recently, Wang *et al.* [56] proposed an end-to-end deep neural network to detect and recognize text in each video frame, which employed the *tracking-by-detection* strategy to associate text regions, then recovered the missed detections with the tracking results, and finally improved recognition results by voting the most frequently appeared text strings.

Different from frame-wise detection and recognition in [31, 36, 56], in this paper we propose an efficient and unified video scene text spotting framework by integrating a spatial-temporal detector, a discriminative tracker, and a text recognizer which reads each text stream one-time with a scoring strategy.

3. The Framework

The architecture of SVST is shown in Figure 2, which consists of four modules: *The spatial-temporal text detector* (SVTD) for detecting text regions among adjacent frames. *The text tracker* (TRIN) for generating text streams with extracted discriminative features by a text re-identification network. *The quality scorer* (TSSN) for scoring text streams and selecting the highest quality of text region as the best candidate. *The text recognizor* to recognize the selected text region as the final result for each text stream. Note that, the feature extraction network ('ResNet Backbone'+'Conv Blocks') of tracker, scorer and recognizor are sharing parameters, which further decreases computational cost.

3.1. Video Text Detection

The text detection architecture is shown in Figure 2.(a), in which the backbone of EAST [63] 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 [64], 2) spatial matching between frames with a comparing and matching strategy [3, 46], and 3) temporal aggregation.

Spatial-temporal aggregation. Formally, let I_t be the $t\text{-}th$ frame in a video, the detection results in I_t can be refined with the detecting 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

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

where $flow_{(t+i,t)}$ is pre-computed with TV-L1 algorithm, $Warp(.)$ is the bilinear warping function applied on each elements in the feature maps, and F^w_{t+i} denotes the feature maps warped from frame I_{t+i} to frame I_t. Thus F is further transferred as the warped $F^w = (F^w_{t-n}, ..., F^w_{t+n})$. Then we generate an enhanced sequence of *confidence map* $C = (C_{t-n}, ..., C_{t+n})$ 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

$$F^{trans}_{t+i} = ReLU(BN(WF^w_{t+i} + b)), \tag{2}$$

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

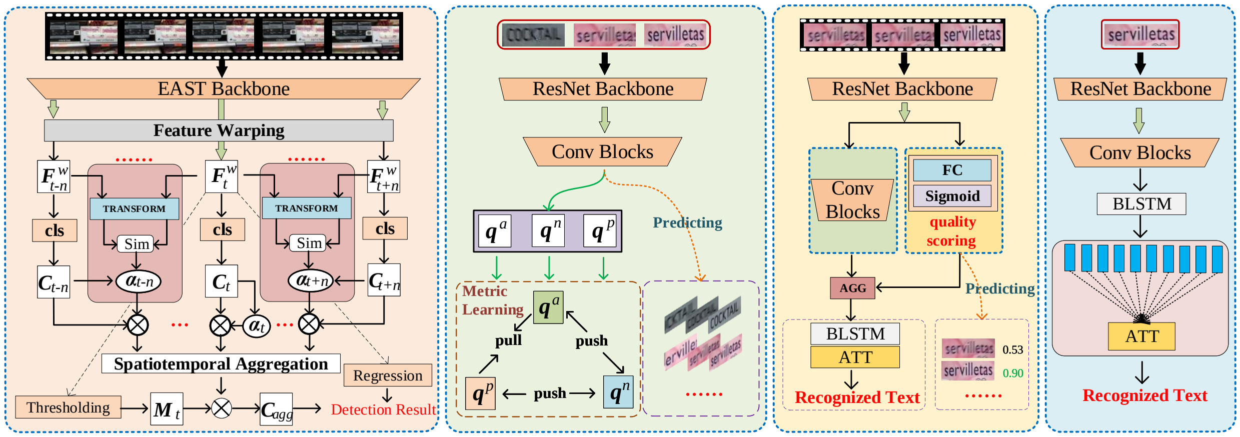
Temporal aggregation. Then we compute the aggregation weights by

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

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

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



(a) **Detection** (b) **Tracking** (c) **Scoring** (d) **Recognition**

Figure 2. The workflow of SVST, which consists of four sub-modules: (a) The spatial-temporal text detector for generating text regions; (b) The text tracker for tracking corresponding text streams; (c) The quality scorer for evaluating the quality of text regions in a text stream; (d) The text recognizor for generating character sequence. The details of neural networks are described in *Experiment* section.

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^w_t as a binary mask with a pre-set threshold (default by 0.5).

The training of SVTD. Similar to [63], the loss function of the detector can be formulated as

$$\mathcal{L}_{detec} = \mathcal{L}_{dice} + \lambda_g \mathcal{L}_{geo}, \tag{5}$$

where $\mathcal{L}_{dice}$ and $\mathcal{L}_{geo}$ mean the losses for the aggregated confidence map $C_{t,ref}$ and the geometry, respectively, and λ_g is a super-parameter (default by 1).

Here, $\mathcal{L}_{dice}$ is implemented with the Dice Loss [32] $\mathcal{L}_{dice} = \frac{2\sum_i^V c_i\hat{c}_i}{\sum_i^V c_i^2+\sum_i^V \hat{c}_i^2}$, where V, c_i and $\hat{c}_i$ separately means the elements number, the predicted confidence and the ground truth. While $\mathcal{L}_{geo}$ is same to that in [63] $\mathcal{L}_{geo} = -(log\frac{B\cap\hat{B}}{B\cup\hat{B}} + \lambda_\theta cos(\theta-\hat{\theta}))$, where B, $\hat{B}$, θ, $\hat{\theta}$ and λ_θ means the predicted geometry, its corresponding ground truth of geometry, the prediction of rotation angle, its corresponding ground truth of rotation angle and the super-parameter (default by 5), respectively.

3.2. Text Region Tracking

The tracking task aims to group corresponding text regions into text streams, shown in Figure 2.(b). Intuitively, the tracker should have the ability to ensure that the features of a text region in one stream should have 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 learning a text re-identification network (TRIN) as used in other tasks (*e.g.* person re-identification [30]). Concretely, we firstly select three regions from localized candidate regions as an image triplet (R^a, R^p, R^n), in which R^aandR^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 TRIN is trained with two metric learning loss: contrastive loss [12] and triplet loss [39]. That is,

$$\mathcal{L}_{trin} = \mathcal{L}_{contra} + \lambda_t \mathcal{L}_{triplet}, \tag{6}$$

where λ_t is default by 1. The contrastive loss $\mathcal{L}_{contra}$=$\frac{1}{N}\sum_{i=0}^{N}||q_i^a$-$q_i^p||$ where $||\cdot||$ denotes the Euclidean distance and N is the number of image triplets. While the triplet loss $\mathcal{L}_{triplet}$=$\frac{1}{N}\sum_{i=0}^{N}[||q_i^a$-$q_i^p||$-$||q_i^a$-$q_i^n||+\alpha]_+$ where $[\cdot]_+$ = max($\cdot$,0) and α is the preset margin.

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

$$MC(R^1, R^2) = \frac{1}{q^1 \circ q^2 + \epsilon}, \tag{7}$$

where $\circ$ denotes the dot product. 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. The details of TRIN is described in *Experiment* section.

3.3. Text Stream Scoring

We focus on learning quality score for each candidate text region in the corresponding stream, which means mapping an image set $\mathcal{R}$={$\mathcal{R}_1, \mathcal{R}_2, ..., \mathcal{R}_N$} to the corresponding score set S={$s_1, s_2, ..., s_N$}. This problem is similar to the quality aware network (QAN) [27] for set-to-set recognition in person re-identification study, in which QAN first generates quality scores for images, and then uses these quality scores to weight images' representations and sums them up to produce the final set's representation.

The scoring network. In this module, we build our text stream scoring network (TSSN) by referring to QAN, as shown in Figure 2.(c). Concretely, we denote Rep_i as the representation (the output of 'Conv Blocks') of $\mathcal{R}_i$. The representation of $\mathcal{R}$ can be aggregated as

$$AGG = \frac{\sum_{i=1}^{N} s_i Rep_i}{\sum_{i=1}^{N} s_i}. \tag{8}$$

And each score s_i is computed by s_i=$Q(I_i)$ where $Q(.)$=$Sigmoid(FC(.))$ is a score generation operation. So the the representation of $\mathcal{R}$ is the fusion of each image's features, which is further decoded as corresponding text transcripts by a recognition module (See in next subsection).

The training of TSSN. Based on the notion that the higher quality text regions are more likely to be predicted correctly, leading to smaller loss values in recognition module, vice versa, the TSSN is weakly supervised with only the text transcripts, and its loss function is represented as

$$\mathcal{L}_{tssn} = -\sum_{t} lnP(\hat{y}_t|\mathcal{R}_i, \theta), \tag{9}$$

where $\hat{y}_t$ is the ground truth of the t-th character and θ is a vector that combines all the network parameters. Note that, the learning of TSSN **does not** need the quality annotation of each text region, and the network tends to enhance the quality score of high-quality text regions and lower the quality score of low-quality text regions.

The scoring phase. In testing stage, the quality score $Q(\mathcal{R}_i)$ of each text region in $\mathcal{R}$ is calculated. And a text region with the highest quality score $max(Q(\mathcal{R}_1), Q(\mathcal{R}_2), ..., Q(\mathcal{R}_N))$ is treated as the winner.

3.4. Text Recognition

In this paper, the text recognition module is not our focus, and we just select attention-based method as our decoder just like used in previous methods [4, 42, 43]. Formally, given the selected image $\mathcal{R}_i$, we encode it into a sequence of feature vectors $H = (h_1, h_2, ..., h_M)$ with the encoder which is same to the 'ResNet Backbone' + 'Conv Blocks' used in tracker and scorer. And the attention decoder is applied for sequently mapping H to target sequence $Y = (y_1, y_2, ..., y_T)$. Specificity, when generating the t-th character, the decoder is briefly described as:

$$\begin{aligned} y_t &= softmax(v\bar{s}_t), \\ \bar{s}_t &= LSTM(y_{t-1}, \bar{c}_t, \bar{s}_{t-1}), \end{aligned} \tag{10}$$

where $\bar{s}_t$ and $\bar{c}_t$ separately represent the LSTM hidden state and the weighted sum of H, that is, $\bar{c}_t$= $\sum_{k=1}^{M} \alpha_{t,k} h_k$, and v is the trainable parameters. Here, $\alpha_{t,k}$=$Attend(\bar{s}_{t-1}, h_k)$ is computed with the attending function $Attend(.)$ [42]. The loss function of the module is same to Equation. 9.

3.5. Jointly Learning TRIN and TSSN

Actually, it will be better if TRIN and TSSN share the same neural network (See Figure 3) and are trained simultaneously, because 1) metric learning can help extract high discriminative features for text stream scoring, 2) the text stream scoring task driven by text transcripts can help enhance features' discrimination of different text, and 3) the parameter sharing further decreases the computational cost.

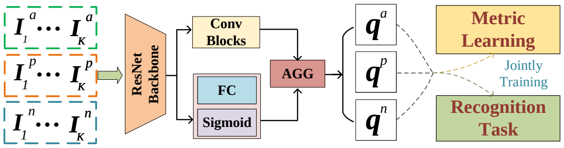


Figure 3. The jointly training of TRIN and TSSN.

Given a triplet of text streams, we can simultaneously obtain the robust features for tracking and scoring by conducting such co-training task, that is

$$\mathcal{L}_{joint} = \lambda_1 \mathcal{L}_{tssn} + \lambda_2 \mathcal{L}_{contra} + \lambda_3 \mathcal{L}_{triplet}, \tag{11}$$

where λ_i (i=1,2,3) denotes the loss weight for different tasks and is detailed in *Experiment* section.

4. Experiments

We evaluate our method on two existing video scene text datasets: IC13 [19] and IC15 [18]. All of our work is built on the CAFFE framework.

4.1. Implementation Details

Detection Network. The EAST backbone is pre-trained on the 'Incidental Scene Text' dataset [18] and 'COCO-Text' dataset [50] by following [25], and then the model is fine-tuned on corresponding video training set (*e.g*. 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}*, a 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 *thresholding=0.2*.

Tracking and Quality Network. The 'ResNet Backbone'+'Conv Blocks' used in tracking, scoring and recognizing is adopted from an image encoder used in [4], and the

'BLSTM'+'ATT' module in scoring and recognizing is an attention decoder used in [4, 42]. The joint network is pretrained on the 8-million synthetic data [16] using 'Adadelta' by following [42], 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. 11 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. In scoring task, image features of $\mathcal{R}$ are extracted from the third convolution blocks in the encoder [4].

Recognition Network. The encoder and decoder are same as TSSN but without quality scoring branch. The training strategy is same to the learning of TSSN, but the parameters of backbone are fixed with *batch-size=24*.

4.2. Evaluation Protocols

The evaluation protocols for text detection, tracking and recognition in videos have been declared in [59]. We only select several mainstream evaluation metrics in this paper.

Detection metrics. Following detection methods [17, 62, 63], *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 $QSHR$=$\sum_{i=0}^{N}\frac{\bar{q}_i}{N}$, 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=$\sum_{i=0}^{N}\frac{\bar{a}_i}{N}$, 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 evaluates performance in word recognition level. That is, a predict word is considered 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 to 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 stream-level recall (REC_s=$\frac{N_r}{N_g}$) and precision (PRE_s=$\frac{N_r}{N_d}$) by *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 stream-level F-score (F-$Score$) is denoted as

$$F\text{-}Score = \frac{2 * PRE_s * REC_s}{PRE_s + REC_s} \tag{12}$$

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 penalizes 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.

4.3. Performance Evaluation of Different Modules

Effectiveness of SVTD. We only evaluate the SVTD performance on IC13 because there is no results reported on IC15. We select frame-wise video text detection as our baseline named D-BASE.

From Table. 1, we find that the D-BASE already outperforms existing approaches by a large margin thanking to the robust EAST, but still suffers from the low recall due to the complicated motion scenarios. As expected, the SVTD can significantly improves recall by 4% *REC* and 1.3% *F-measure*, but with a 4.3% drop on *PRE*. Actually, boosting recall performance is more important when facing a 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.* [61]	47.02	46.30	46.65
Shivakumara [45]	53.71	51.15	50.67
Yin *et al.* [58]	54.73	48.62	51.56
Wang *et al.* [55]	51.74	58.34	54.45
D-BASE	56.21	**85.76**	67.91
SVTD	**60.23**	81.45	**69.25**

Table 1. Detection performance evaluation of SVTD on IC13.

Effectiveness of TRIN. Here, we directly extract the robust features from the recognizer's output of 'Conv Blocks' for tracking (used in Equation. 7), which is treated as our baseline named T-BASE. Table 2 shows the comparing results.

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 $MOTA_D$. TRIN-based method can further separately improve the performance by 0.13 and 0.03 on ATA_D and $MOTA_D$, and maintains the $MOTP_D$ performance.

Evaluation on IC15. T-BASE also achieves a comparable results with previous methods. Comparing to the best reported results, TRIN-based method significantly improves the ATA_D and $MOTA_D$ by 0.04 and 0.03, but falls behind [57] on $MOTP_D$. However, [57] points out that ATA_D is the most important metric in IC15 because ATA_D measures the tracking performance over all the text.

Effects of jointly training. Comparing the separated training of TRIN and TSSN, the jointly learning strategy helps improve the tracking results on two datasets.

Dataset	Methods	ATA_D	$MOTP_D$	$MOTA_D$
IC13	IC13's base [19]	0.00	0.63	-0.09
	TextSpotter [35]	0.12	0.67	0.27
	Nguyen *et al.* [36]	0.15	-	-
	T-BASE	0.50	0.75	0.64
	TRIN	0.62	0.75	0.65
	TRIN + TSSN	**0.63**	**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.* [56]	0.56	0.70	0.57
	Yang *et al.* [57]	0.61	**0.79**	0.66
	T-BASE	0.53	0.76	0.65
	TRIN	0.64	0.76	0.68
	TRIN + TSSN	**0.65**	0.76	**0.69**

Table 2. Tracking performance evaluation of TRIN on IC13 and IC15. The suffix 'D' means tracking is applied for detection.

Effectiveness of TSSN. 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 of TSSN.

To evaluate the proposed scoring mechanism, we compare our method with two commonly used scoring-and-selection 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 [56].

Methods	$QSHR$ (IC13/IC15)	RCR (IC13/IC15)	$\#Frames$ (IC13/IC15)
PCW	74.55/75.83	66.06/66.32	all
HFP	75.32/76.34	68.30/68.56	all
TSSN	81.49/82.51	69.15/69.40	**1**
TRIN+TSSN	**82.00/82.77**	**69.66/69.92**	**1**

Table 3. Effects of TSSN on IC13 and IC15 compared with other frame selection methods. The '#Frame' denotes the number of regions needs to be recognized in a stream. The 'all' means each region is recognized.

Table. 3 shows that *HFP* performs better than *PCW* in both *QSHR* and *RCR* by the voting process. Compared to HFP, *TSSN* further achieves a 2.8% improvement in *QSHR* and 0.8% increasing in *RCR*. More than that, TSSN only needs recognize a text stream one-time, which can greatly decrease the computational cost (compared in the #Frame column). Besides, the jointly learning strategy further helps improve the scoring performance.

It is worth noticing that TSSN 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 TSSN 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 4 gives the results and demonstrates that TSSN 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
TSSN	**56.69/59.05**	**72.82/73.22**

Table 4. Extreme testing of TSSN on IC13 and IC15 compared with other frame selection methods.

4.4. End-to-end Evaluation

To analyze the contributions of above components, we conduct the ablation study on the popular IC15 dataset. Table. 5 shows that 1) Comparing to the baseline (D-BASE), SVTD can steadily help the end-to-end recognition. 2) The TRIN greatly improves the end-to-end performance of PRE_s, REC_s and F-$score$ by 2.5%, 6% and 4% respectively. 3) As expected, the jointly training of TRIN and TSSN achieves the best performance, and improves the *D-BASE+T-BASE+TSSN* by 6.5%.

D-BASE	✓	✓	✓			
SVTD				✓	✓	✓
T-BASE	✓			✓		
TRIN		✓			✓	
TSSN	✓	✓		✓	✓	
TRIN+TSSN			✓			✓
PRE_s	69.49	71.24	72.51	61.71	64.17	66.66
REC_s	54.85	60.29	60.88	62.81	68.70	68.74
F-$score$	61.30	65.30	66.18	62.26	66.38	**67.68**

Table 5. The ablation evaluation of our framework on IC15.

Conventionally, we also place the frame-wise recognition results on IC15 by referring to the previous works, in which we test the results with the *SVTD+TRIN* setting. Table. 6 gives the results and 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.* [56]	0.70	0.69	0.60
Ours	**0.76**	**0.70**	**0.63**

Table 6. The traditional end-to-end evaluation on IC15. The suffix 'R' mean tracking is applied for measuring recognition.

5. Conclusion

In this paper, we propose an unified framework for efficiently spotting video scene text. Firstly, we learn a spatial-temporal video text detector for robustly localizing text regions in scene videos. Secondly, we design a text re-identification network to learn the discriminative features for text tracking. Finally, with a learnt text stream scoring model, we select the best text region from a text stream for the final text recognition. In future, we'll further explore the spatial-temporal text localization and recognition in an end-to-end trainable way.

References

[1] F. Bai, Z. Cheng, Y. Niu, S. Pu, and S. Zhou. Edit Probability for Scene Text Recognition. In *CVPR*, pages 1508–1516, 2018. 1, 2

[2] F. Borisyuk, A. Gordo, and V. Sivakumar. Rosetta: Large Scale System for Text Detection and Recognition in Images. In *SIGKDD*, pages 71–79, 2018. 1, 2

[3] D. Chen, H. Li, T. Xiao, S. Yi, and X. Wang. Video Person Re-Identification With Competitive Snippet-Similarity Aggregation and Co-Attentive Snippet Embedding. In *CVPR*, pages 1169–1178, 2018. 3

[4] Z. Cheng, F. Bai, Y. Xu, G. Zheng, S. Pu, and S. Zhou. Focusing attention: Towards accurate text recognition in natural images. In *ICCV*, pages 5086–5094, 2017. 1, 2, 5, 6

[5] Z. Cheng, Y. Xu, F. Bai, Y. Niu, S. Pu, and S. Zhou. AON: Towards Arbitrarily-Oriented Text Recognition. In *CVPR*, pages 5571–5579, 2018. 1, 2

[6] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, volume 1, pages 886–893, 2005. 2

[7] V. Fragoso, S. Gauglitz, S. Zamora, J. Kleban, and M. Turk. TranslatAR: A mobile augmented reality translator. In *WACV*, pages 497–502, 2011. 2

[8] L. Gómez and D. Karatzas. MSER-based real-time text detection and tracking. In *ICPR*, pages 3110–3115, 2014. 2

[9] H. Goto and M. Tanaka. Text-tracking wearable camera system for the blind. In *ICDAR*, pages 141–145, 2009. 2, 3

[10] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber. Connectionist Temporal Classification : Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In *ICML*, pages 369–376, 2006. 2

[11] J. Greenhalgh and M. Mirmehdi. Recognizing Text-Based Traffic Signs. *IEEE TITS*, 16(3):1360–1369, 2015. 3

[12] R. Hadsell, S. Chopra, and Y. LeCun. Dimensionality reduction by learning an invariant mapping. In *CVPR*, pages 1735–1742, 2006. 4

[13] T. He, Z. Tian, W. Huang, C. Shen, Y. Qiao, and C. Sun. An End-to-End TextSpotter With Explicit Alignment and Attention. In *CVPR*, pages 5020–5029, 2018. 2

[14] W. He, X.-Y. Zhang, F. Yin, and C.-L. Liu. Deep Direct Regression for Multi-Oriented Scene Text Detection. In *ICCV*, pages 745–753, 2017. 1, 2

[15] H. Hu, C. Zhang, Y. Luo, Y. Wang, J. Han, and E. Ding. WordSup: Exploiting Word Annotations for Character Based Text Detection. In *ICCV*, pages 4940–4949, 2017. 1, 2

[16] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman. Synthetic Data and Artificial Neural Networks for Natural Scene Text Recognition. *arXiv preprint arXiv:1406.2227*, 2014. 6

[17] Y. Jiang, X. Zhu, X. Wang, S. Yang, W. Li, H. Wang, P. Fu, and Z. Luo. R2CNN: Rotational Region CNN for Orientation Robust Scene Text Detection. *CoRR*, abs/1706.09579, 2017. 1, 2, 6

[18] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, S. Ghosh, A. Bagdanov, M. Iwamura, J. Matas, L. Neumann, V. R. Chandrasekhar, S. Lu, et al. ICDAR 2015 competition on robust reading. In *ICDAR*, pages 1156–1160, 2015. 5, 7, 8

[19] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L. G. i Bigorda, S. R. Mestre, J. Mas, D. F. Mota, J. A. Almazan, and L. P. De Las Heras. ICDAR 2013 robust reading competition. In *ICDAR*, pages 1484–1493, 2013. 5, 7

[20] V. Khare, P. Shivakumara, and P. Raveendran. A new Histogram Oriented Moments descriptor for multi-oriented moving text detection in video. *Expert Systems with Applications*, 42(21):7627–7640, 2015. 6

[21] K. I. Kim, K. Jung, and J. H. Kim. Texture-based approach for text detection in images using support vector machines and continuously adaptive mean shift algorithm. *IEEE TPAMI*, 25(12):1631–1639, 2003. 2

[22] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955. 2, 4

[23] H. Li, P. Wang, and C. Shen. Towards End-To-End Text Spotting With Convolutional Recurrent Neural Networks. In *ICCV*, pages 5238–5246, 2017. 1, 2

[24] M. Liao, Z. Zhu, B. Shi, G.-s. Xia, and X. Bai. Rotation-Sensitive Regression for Oriented Scene Text Detection. In *CVPR*, pages 5909–5918, 2018. 1, 2

[25] X. Liu, D. Liang, S. Yan, D. Chen, Y. Qiao, and J. Yan. FOTS: Fast Oriented Text Spotting with a Unified Network. In *CVPR*, pages 5676–5685, 2018. 1, 2, 5

[26] Y. Liu and L. Jin. Deep Matching Prior Network: Toward Tighter Multi-oriented Text Detection. In *CVPR*, pages 3454–3461, 2017. 1, 2

[27] Y. Liu, J. Yan, and W. Ouyang. Quality Aware Network for Set to Set Recognition. In *CVPR*, pages 4694–4703, 2017. 5

[28] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 2

[29] J. Ma, W. Shao, H. Ye, L. Wang, H. Wang, Y. Zheng, and X. Xue. Arbitrary-oriented scene text detection via rotation proposals. *IEEE Transactions on Multimedia*, 2018. 1, 2

[30] N. McLaughlin, J. Martinez del Rincon, and P. Miller. Recurrent Convolutional Network for Video-based Person Re-Identification. In *CVPR*, pages 1325–1334, 2016. 4

[31] C. Merino-Gracia and M. Mirmehdi. Real-time text tracking in natural scenes. *IET Computer Vision*, 8(6):670–681, 2014. 1, 3

[32] F. Milletari, N. Navab, and S.-A. Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *3DV*, pages 565–571, 2016. 4

[33] R. Minetto, N. Thome, M. Cord, N. J. Leite, and J. Stolfi. Snoopertrack: Text detection and tracking for outdoor videos. In *ICIP*, pages 505–508, 2011. 2

[34] Y. Na and D. Wen. An effective video text tracking algorithm based on sift feature and geometric constraint. In *PRCM*, pages 392–403, 2010. 2

[35] L. Neumann and J. Matas. On combining multiple segmentations in scene text recognition. In *ICDAR*, pages 523–527, 2013. 7

[36] P. X. Nguyen, K. Wang, and S. Belongie. Video text detection and recognition: Dataset and benchmark. In *WACV*, pages 776–783, 2014. 1, 2, 3, 7

[37] M. Petter, V. Fragoso, M. Turk, and C. Baur. Automatic text detection for mobile augmented reality translation. In *Workshop on ICCV*, pages 48–55, 2011. 2

[38] X. Rong, C. Yi, X. Yang, and Y. Tian. Scene text recognition in multiple frames based on text tracking. In *ICME*, pages 1–6, 2014. 2, 3

[39] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *CVPR*, pages 815–823, 2015. 4

[40] B. Shi, X. Bai, and S. Belongie. Detecting Oriented Text in Natural Images by Linking Segments. In *CVPR*, pages 2550–2558, 2017. 1, 2

[41] B. Shi, X. Bai, and C. Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE TPAMI*, 39(11):2298–2304, 2017. 1, 2

[42] B. Shi, X. Wang, P. Lyu, C. Yao, and X. Bai. Robust Scene Text Recognition with Automatic Rectification. In *CVPR*, pages 4168–4176, 2016. 1, 2, 5, 6

[43] B. Shi, M. Yang, X. Wang, P. Lyu, C. Yao, and X. Bai. Aster: An attentional scene text recognizer with flexible rectification. *IEEE TPAMI*, pages 1–1, 2018. 1, 2, 5

[44] H. Shiratori, H. Goto, and H. Kobayashi. An efficient text capture method for moving robots using dct feature and text tracking. In *ICPR*, volume 2, pages 1050–1053, 2006. 2

[45] P. Shivakumara, R. P. Sreedhar, T. Q. Phan, S. Lu, and C. L. Tan. Multioriented video scene text detection through bayesian classification and boundary growing. *IEEE TCSVT*, 22(8):1227–1235, 2012. 6

[46] J. Si, H. Zhang, C.-G. Li, J. Kuen, X. Kong, A. C. Kot, and G. Wang. Dual Attention Matching Network for Context-Aware Feature Sequence based Person Re-Identification. In *CVPR*, pages 5363–5372, 2018. 3

[47] M. Tanaka and H. Goto. Autonomous text capturing robot using improved DCT feature and text tracking. In *ICDAR*, volume 2, pages 1178–1182, 2007. 2

[48] M. Tanaka and H. Goto. Text-tracking wearable camera system for visually-impaired people. In *ICPR*, pages 1–4, 2008. 2

[49] S. Tian, W.-Y. Pei, Z.-Y. Zuo, and X.-C. Yin. Scene Text Detection in Video by Learning Locally and Globally. In *IJCAI*, pages 2647–2653, 2016. 2

[50] A. Veit, T. Matera, L. Neumann, J. Matas, and S. Belongie. COCO-Text: Dataset and Benchmark for Text Detection and Recognition in Natural Images. *CoRR*, abs/1601.07140, 2016. 5

[51] E. A. Wan and R. Van Der Merwe. The unscented Kalman filter for nonlinear estimation. In *AS-SPCC*, pages 153–158, 2000. 3

[52] F. Wang, L. Zhao, X. Li, X. Wang, and D. Tao. Geometry-Aware Scene Text Detection With Instance Transformation Network. In *CVPR*, pages 1381–1389, 2018. 1, 2

[53] J. Wang and X. Hu. Gated recurrent convolution neural network for OCR. In *NIPS*, pages 335–344, 2017. 1, 2

[54] K. Wang, B. Babenko, and S. Belongie. End-to-end scene text recognition. In *ICCV*, pages 1457–1464. IEEE, 2011. 3

[55] L. Wang, Y. Wang, S. Shan, and F. Su. Scene Text Detection and Tracking in Video with Background Cues. In *ICMR*, pages 160–168, 2018. 2, 6

[56] X. Wang, Y. Jiang, S. Yang, X. Zhu, W. Li, P. Fu, H. Wang, and Z. Luo. End-to-End Scene Text Recognition in Videos Based on Multi Frame Tracking. In *ICDAR*, volume 1, pages 1255–1260, 2017. 1, 3, 7, 8

[57] X.-H. Yang, W. He, F. Yin, and C.-L. Liu. A Unified Video Text Detection Method with Network Flow. In *ICDAR*, volume 1, pages 331–336, 2017. 2, 7

[58] X.-C. Yin, X. Yin, K. Huang, and H.-W. Hao. Robust text detection in natural scene images. *IEEE TPAMI*, 36(5):970–983, 2014. 2, 6

[59] X.-C. Yin, Z.-Y. Zuo, S. Tian, and C.-L. Liu. Text detection, tracking and recognition in video: a comprehensive survey. *IEEE TIP*, 25(6):2752–2773, 2016. 2, 6

[60] K. Zhang, L. Zhang, Q. Liu, D. Zhang, and M.-H. Yang. Fast visual tracking via dense spatio-temporal context learning. In *ECCV*, pages 127–141, 2014. 2

[61] X. Zhao, K.-H. Lin, Y. Fu, Y. Hu, Y. Liu, and T. S. Huang. Text from corners: a novel approach to detect text and caption in videos. *IEEE TIP*, 20(3):790–799, 2011. 6

[62] Z. Zhong, L. Sun, and Q. Huo. An Anchor-Free Region Proposal Network for Faster R-CNN based Text Detection Approaches. *CoRR*, abs/1804.09003, 2018. 6

[63] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, and J. Liang. EAST: An Efficient and Accurate Scene Text Detector. In *CVPR*, pages 5551–5560, 2017. 1, 2, 3, 4, 6

[64] X. Zhu, Y. Wang, J. Dai, L. Yuan, and Y. Wei. Flow-Guided Feature Aggregation for Video Object Detection. In *ICCV*, pages 408–417, 2017. 3

[65] Z.-Y. Zuo, S. Tian, W.-y. Pei, and X.-C. Yin. Multi-strategy tracking based text detection in scene videos. In *ICDAR*, pages 66–70, 2015. 2