Contrastive Learning of Semantic and Visual Representations for Text Tracking

Zhuang Li\textsuperscript{1*}, Weijia Wu\textsuperscript{2*}, Mike Zheng Shou\textsuperscript{3}, Jiahong Li\textsuperscript{1†}, Size Li\textsuperscript{1}, Zhongyuan Wang\textsuperscript{1}, Hong Zhou\textsuperscript{2†}

\textsuperscript{1}Kuaishou Technology \quad \textsuperscript{2}Zhejiang University \quad \textsuperscript{3}Show Lab, National University of Singapore

Abstract

Semantic representation is of great benefit to the video text tracking (VTT) task that requires simultaneously classifying, detecting, and tracking texts in the video. Most existing approaches tackle this task by appearance similarity in continuous frames, while ignoring the abundant semantic features. In this paper, we explore to robustly track video text with contrastive learning of semantic and visual representations. Correspondingly, we present an end-to-end video text tracker with Semantic and Visual Representations (SVRep), which detect and track texts by exploiting the visual and semantic relationships between different texts in a video sequence. Besides, with a light-weight architecture, SVRep achieves state-of-the-art performance while maintaining competitive inference speed. Specifically, with a backbone of ResNet-18, SVRep achieves an $\text{ID}_{F1}$ of 65.9\%, running at 16.7 FPS, on the ICDAR2015 (video) dataset with 8.6\% improvement than the previous state-of-the-art methods.

1. Introduction

When trying to track text in a video sequence, it is not only important to locate the texts but more critical to read and comprehend it in the context to the visual scene. Video text tracking (VTT) \cite{43} is the task that requires simultaneously classifying, detecting, and tracking text instances in video. Most mainstream approaches \cite{3, 44} all follow the tracking-by-detection paradigm, which firstly tackles each frame in a video sequence, and then associates the similar text instances in the adjacent frames by various appearance-based matching strategies (i.e., use their IoU, transcription, and feature). IoU-based methods \cite{39, 12} rely on image text detection models \cite{37}, where IoU of detected text boxes from two adjacent frames is higher than the given threshold are associated. Similarly, visual feature-based methods \cite{3, 44} compute visual feature similarity from different text instances in adjacent frames, to join pairs with the same text instance.

However, these methods all ignore the temporal contexts in a video sequence and semantic representation (i.e., feature) between different text instances. In contrast to the visual features of appearance, semantic features are robust cues for matching and tracking text instances in a video sequence. As shown in Figure 1 (a), due to rapid motion blur, out-of-focus, and artifacts issues, the pure visual feature usually causes the ID switch of having already tracked text instances. But it is easy for humans to track the word “FRAICHEC” via recognizing, and comprehending the text, although the low-quality video. Therefore, we try to allow the model to read text, relate it to other texts, and then track.

In this paper, we firstly propose an end-to-end video text tracker with Semantic and Visual representations (SVRep), which detects, and tracks texts by exploiting the visual and semantic relationships between different texts in a video sequence. As shown in Figure 1 (b), given a video clip with texts, SVRep learns the visual and semantic representations.
by detecting and recognizing text, then relating it to other texts in the video clip. With the abundant semantic representations, visual representations, and mutual information, the more stable tracking performance is presented. But there are two main challenges: 1) How to learn efficient visual and semantic representations (i.e., features) for an end-to-end video text tracking model. 2) How to maximize the mutual information of visual and semantic representations between text instances. For the former one, we propose three encoders, i.e., positional encoder, visual encoder, semantic encoder. Positional encoder learns the location information by embedding bounding box coordinates of each text instance. Visual encoder learns visual appearance features (e.g., color, shape, texture) of each text instance by the Masked RoI [38]. Semantic encoder learns semantic features of each text instance by an CTC-based recognition head. For the later one, inspired by contrastive learning [1], with semantic and visual embedding, we maximize agreement between the same texts and maximizing disagreement between different texts by contrasting positive pairs against negative pairs.

To achieve high efficiency, PAN++ [38] as a lightweight architecture for scene text detection and recognition is adopted to combine the proposed contrastive learning of semantic and visual representations. On the ICDAR2015(video) [14] dataset, the ID$_F$:ID$_V$ of “Ours+ResNet18” reaches 65.9%, with up to 10% improvements than that of Free [3] and TransVTSpotter [41], while its inference speed (i.e., 16.7 FPS) is faster. The main contributions of this work are listed below:

- We propose three encoders, i.e., positional encoder, visual encoder, and semantic encoder, which learn location representation (i.e., feature), visual representation, and semantic representation, respectively. With the three encoders, the proposed model is the first one, like human, to track text by recognizing it.
- With contrastive learning, we firstly propose to maximize agreement between the same texts (i.e., positive pair) and maximizing disagreement between different texts (i.e., negative pairs) both in visual embedding space and semantic bedding space.
- SVRep achieves the state-of-the-art performance on five public datasets with faster speed. Especially, SVRep achieves 65.9% ID$_F$ and 16.7 FPS for video text tracking task on ICDAR2015 [14], with up to 10% improvements than the previous SOTA methods in performance.

2. Related Work

2.1. Text Detection and Recognition

Recent methods [37, 29, 38] based on deep learning have been made tremendous progress for image-level text detection and recognition. For text detection, CTPN [32] developed vertical anchors and constructed a CNN-RNN joint model to detect horizontal text lines. PSENet [37] proposed the post-processing of progressive scale expansion for improving the detection accuracy. For text recognition, CRNN [29] proposed one of the first scene text recognition models that are based on a convolutional feature extractor and a recurrent neural network. CharNet [23] proposed a neural network that includes a spatial transformer network which is used to put focus on the features of single characters. For end-to-end text spotting, [21] proposed the first end-to-end scene text spotting method, which uses an RoI Pooling [27] to joint detection and recognition features. Its improved version [36] further significantly improves the performance. Based on the kernel representation, PAN++ [38] proposed an end-to-end text spotting framework, which detects and recognizes a text with a feature extractor (Masked RoI), and a lightweight attention-based recognition head. However, these image-based methods can not obtain temporal information (i.e., tracking id) in the video, which is essential for other video-and-language tasks such as video understanding.

2.2. Video Text Tracking

The detailed survey [43] summarizes and compares the existing text detection, tracking, and recognition methods in a video before 2016. [35] captures spatial-temporal information by exploiting the cues of the background regions of the text. Wang et al. [39] links texts in the current frame and several previous frames to obtain the tracking results by hand-crafted post-processing, such as IoU-based associations. [44] tracked texts by using ConvLSTM to capture spatial structure information and motion memory, and an appearance-geometry descriptor is proposed to learn the visual representation of text instances. ASGD [5] introduced a new character center segmentation branch, and try to extract semantic features, which encode the category and position of characters. But these features still belong to visual features for learning from classification and detection tasks. The above methods track text by appearance similarity, ignoring the abundant semantic features. In this work, we try to allow the model to recognize text, relate it to other texts with semantic and visual features, and then track.

2.3. Contrastive Learning

Dating back to [10], these approaches learn visual representations by measuring the similarities of sample pairs in representation space. And the contrastive loss is at the core of several recent works on unsupervised learning [1, 11] by contrasting positive pairs against negative pairs. Except for unsupervised learning, there are relations between generative adversarial networks [6] and contrastive losses (i.e., noise-contrastive estimation [9]). The contrastive loss [6], as
a widely successful technique for unsupervised data generation, measuring the difference between probability distributions. In this work, we firstly adopt the contrastive losses (i.e., noise-contrastive estimation [9]) to model the relationship between different text pairs both in visual representation and semantic representation spaces.

3. Method

The architecture of the proposed method is shown in Figure 2, PAN++ [38] as the base network is adopted, including backbone, FPN, Up-sample, and detection head. And the CTC-based recognition head [29] is proposed to replace the attention-based recognition head [38]. There are two benefits for using CTC-based: 1) much shorter inference time using parallel decoding. 2) more discrete semantic representation. More detailed analysis and experiments for the two points can be found in the supplementary material. For semantic and visual representations learning, we propose three encoders, i.e., positional encoder, visual encoder and semantic encoder. To track text with visual and semantic relationship of text, SVRep maximizes agreement between the same text example of different frames via a contrastive learning. Given a set \( \{x_n^k\} \) (e.g., ‘Text’, ‘Mall’ in Figure 2) of text instances in a video sequences, including a positive pair (i.e., the same text in different frame) of examples \( x_i \) and \( x_{i-1} \), where \( t \) means \( t \)-th frame and \( i \) means \( i \)-th text instance in \( t \)-th frame. The contrastive learning aims to identify \( x_i \) in \( \{x_n^k|k \neq j \cup n \neq t-1\} \) for a given \( x_{i-1} \), both in semantic and visual embedding spaces. Both in visual and semantic embedding spaces, we argue that the same text in a continuous video sequence should tend to close and different text with different identification shows a greater distance.

3.1. Semantic and Visual Representation Learning

Human usually watches videos by frequently reading, tracking, and comprehending visual and semantic information. To follow the human mechanism, we design an recognition head and three encoders (i.e., positional encoder, visual encoder, and semantic encoder) for simultaneously learning the visual and semantic representations.

Recognition Head. With Masked RoI and the rotated bounding box from detection head, the fixed-size visual feature patches \((h \times w)\) are extracted for each text instances. Similar to PAN++ [38], in all experiments, \( h \) and \( w \) are set to 8 and 32 pixels, respectively. Then one-layer Bidirectional LSTM (BiLSTM) [8], as the sequence model, extracts a semantic sequence \( \hat{H} = \text{Seq}_V(V) \) from the visual feature patches.

As shown in figure 2, the transcription module [29] produces the final output (a sequence of characters) from the input semantic sequence \( H = h_1, \ldots, h_T \), where \( T \) is the sequence length. Then Connectionist Temporal Classification (CTC) [7] trains the network to optimize the summation of probabilities over all paths:

\[
p(Y | H) = \sum_{\pi: \mathcal{M}(\pi) = Y} p(\pi | H),
\]

\[
\mathcal{L}_{rec} = -\log p(Y | H),
\]

where \( \mathcal{M} \) defines the operation of mapping all possible paths \( \pi \) to the target label. For example, it maps the path “ttt – e – xxx – t – ” into “text”.

Semantic Encoder. In this encoder, the semantic sequence feature \( H = h_1, \ldots, h_T \) from BiLSTM is fed into semantic encoder for learning a high-dimensional semantic representation (i.e., feature) \( R_s \), whose size is \( 128 \times 1 \times 1 \). To reduce the computation overhead, we only employ two
1 × 1 convolutions with 1 stride, two batch normalization layers [13], and max-pooling layer to extract the feature. The detailed architecture for the encoder is provided in the supplementary material.

**Visual Encoder.** Corresponding to the semantic sequence feature \( H \) from the recognition head, the fixed-size visual feature patches \( V \) of each text instance from Masked RoI is used to extract the high-dimensional visual representation \( R_v \), whose size is \( 128 \times 1 \times 1 \). The use of \( V \) from Masked RoI has three benefits: 1) The visual feature patches \( V \) of each text instance contains abundant visual appearance representations, e.g., color, shape, texture. 2) The binary mask of the rotated bounding box can eliminate the noise features caused by the background or other text lines, so as to accurately extract the visual features. 3) The reusing with recognition head reduces the time cost of feature extraction. Similar to semantic encoder, two \( 3 \times 3 \) convolutions with 1 stride, two batch normalization layers [13], and max-pooling layer are used to extract the high-dimensional feature. The detailed architecture for the encoder is provided in supplementary material.

**Positional Encoder.** Except for the attribute (i.e., visual and semantic features) of text instance, the positional information (location) is equally important for tracking. Inspired by the previous works [31, 33], we extract the high-dimensional positional representation \( R_p \) by embedding bounding box coordinates of each text instance (i.e., positional encoder). Each Masked RoI of text is characterized by a 4-d vector, as \((x_{LT}, y_{LT}, x_{RB}, y_{RB})\), where \((x_{LT}, y_{LT})\) and \((x_{RB}, y_{RB})\) denote the coordinate of the top-left and bottom-right corner respectively, and \(W, H\) are the width and height of the input image. Then the 4-d vector is embedded into a positional feature embedding (of \( 4 \times 128 \) in paper) by computing sine and cosine functions of different wavelengths. But the relative positional feature embedding is still a low-level feature, which cannot be extended to the high-dimensional representation space. Therefor, we use two \( 1 \times 1 \) convolutions with 1 stride, two batch normalization layers [13], and max-pooling layer to further convert the positional feature embedding to the final high-dimensional positional representation \( R_p \), whose size is \( 128 \times 1 \times 1 \), similar to the visual and semantic representations.

### 3.2. Contrastive Learning for Text Instance

The idea behind contrastive learning is to learn an embedding that separates (contrasts) samples from two different distributions. Given a text instance set \( \{x^k_n\} \) (e.g., ‘Text’, ‘Mall’ in figure 2) in a video sequences, where \( n \) means \( n \)-th frame and \( k \) means \( k \)-th text instance in \( n \)-th frame. We consider the same text in adjacent frames, e.g., ‘Text’ \( x^k_t \) in \( t \)-th frame and ‘Text’ \( x^k_{t-1} \) in \( (t-1) \)-th frame, which we call positive pair, different text instances with different semantic and visual information (e.g., ‘Text’ and ‘Mall’), which we call negative pair.

A “critic” (a discriminating function) \( F_\theta(\cdot) \) is trained to maximize agreement for positive pairs and maximizing disagreement for negative pairs in semantic and visual embedding space. And for each text instance \( x^k_n \), the corresponding positional representation \( R_p \), semantic representation \( R_s \), and visual representation \( R_v \) have already obtained from three encoders, and require further integration for learning the discriminating function \( F_\theta(\cdot) \). The three features (\( R_p, R_v, R_s \)) are concatenated together directly, and fed into two \( 1 \times 1 \) convolutions to output the final high-dimensional representation \( R^k_n \), whose size is \( b \times c \) \((c \) is set to 128 in paper).

For convenience, we flatten text examples in \( T \) frames, define \( R^k_n \) as \( z_p \), where \( p = k + d \) \((d \) means the sum of text examples before \( n \)-th frame). And we randomly sample a minibatch of \( T \) frames, \( N \) text examples and define the contrastive prediction task on positive pairs of the same text in different frame, resulting in \( N/2 \) data points. Similar to SimCLR [1], we do not sample negative examples explicitly. Instead, given a positive pair (i.e., \( z_p, z_q \)), we treat the other text pairs within a minibatch as negative examples. Let \( \text{sim}(u, v) = u^T v / ||u|| ||v|| \) denote the cosine similarity between two vectors \( u \) and \( v \). Then the loss function for a positive pair of examples \( z_p \) and \( z_q \) is defined as

\[
L_{p,q} = -\log \frac{\exp(\text{sim}(z_p, z_q)/\tau)}{\sum_{K=1}^{N} I_{[K \neq p]} \exp(\text{sim}(z_p, z_K)/\tau)},
\]

where \( I_{[K \neq q]} \in \{0, 1\} \) is an indicator function, and \( \tau \) denotes a temperature parameter. The final NT-Xent loss [1] (contrastive loss) is computed across all positive pairs, both \((p, q)\) and \((p, q)\) in a mini-batch:

\[
L_{\text{con}} = \frac{1}{N} \sum_{(z_p, z_q) \in \{z_1, z_2, \ldots, z_N\}} L_{p,q},
\]

where \( z_p, z_q \in \{z_1, z_2, \ldots, z_N\} \) is the set of the final text representations from the discriminating function \( F_\theta(\cdot) \).

### 3.3. Loss Function of Multi-Task Learning

The proposed pipeline contains three loss, i.e., detection loss, recognition loss, and contrastive loss, which belong to three different tasks. To improve learning efficiency and prediction accuracy, Multi-task learning [16] is adopted in our method to learn multiple objectives from a shared representation.

\[
L = \frac{1}{\sigma_1} L_{\text{det}} + \frac{1}{\sigma_2} L_{\text{rec}} + \frac{1}{\sigma_3} L_{\text{con}} + \log \sigma_1 \sigma_2 \sigma_3,
\]

where \( \sigma_1, \sigma_2, \sigma_3 \) are three learnable parameters, and the \( \log \sigma_1 \sigma_2 \sigma_3 \) is the a regulariser for the noise terms.
3.4. Inference

In the inference phase, similar to many previous works [39, 5], SVRep obtains the final tracking result (i.e., tracking trajectory) by the cosine similarity matching for each text pair \((z_q, z_p)\) in adjacent frame and the Kuhn-Munkres(KM) algorithm [20].

4. Experiments

ICDAR2013 Video [15] is proposed in the ICDAR 2013 Robust Reading Competition, which contains 13 videos for training and 15 videos for testing. These videos are harvested from indoors and outdoors scenarios, and each text is labeled as a quadrangle with 4 vertexes in word-level. **ICDAR2015 Video** [14] is the expanded version of ICDAR2013(video), which consists of a training set of 25 videos (13,450 frames) and a test set of 24 video (14,374 frames). **Minetto** [24] contains 5 videos in outdoor scenes. The resolution is fixed as 640×480. Each text is labeled in the form of axis-aligned bounding box. As all videos are for testing, we use the model trained on ICDAR2015 Video to evaluate this dataset directly.

**YouTube Video Text (YVT)** [25] dataset is harvested from YouTube, contains 30 videos, where 15 videos for training and 15 videos for testing. Different from the above datasets, it contains web videos except for scene videos.

**BOVText** [41] is a bilingual, open world dataset, including 2,021 videos with 1,757,598 frames. The data is collected from the worldwide user of YouTube and KuaiShou, cover various daily scenarios without region limitation and virtual scenes.

4.1. Implementation Details

All the experiments are conducted on PyTorch with Tesla V100 GPUs. We use the PAN++ [38] as our basic network. Following the common practices [37, 38], we ignore the blurred text regions labeled as “DO NOT CARE” during training, and apply random scale, random horizontal flip, random rotation, and random crop on training images. The model is firstly pretrained on the COCO-Text [34], and then finetune on other video datasets. COCO-Text is a largest scene text detection dataset with 63,686 images, which reuses the images from MS-COCO dataset [22]. For the static images from COCO-Text, we apply the random shift [46] to generate video clips with pseudo tracks. All models are optimized by using ADAM [18] optimizer with a batch size of 48 on 8 GPUs. The initial learning rate is set to \(1 \times 10^{-3}\). In the testing phase, we resize the input image to different scales and report the performance on text detection and video text tracking tasks. All results are tested with a batch size of 1 on a V100 GPU and a 2.20GHz CPU in a single thread. In the metric, Mostly Tracked (\(M\)-*Tracking*) denotes the number of objects tracked for at least 80 percent of lifespan, Mostly Lost (\(M\)-*Lost*) denotes number of objects tracked less than 20 percent of lifespan.

### Table 1 – Effect of Semantic, Visual and Positional Encoder on ICDAR2015 video

| Semantic | Visual | Positional | ID\(F_1\) | MOTA | MOTP |
|----------|--------|------------|---------|------|-----|
| ✓        | ✓      | ✓          | 64.8    | 48.2 | 73.9 |
| ✓        | ✓      |            | 65.0    | 48.5 | 73.6 |
| ✓        | ✓      | ✓          | 65.9    | 49.1 | 73.8 |

### Table 2 – Comparison of different text instances association methods on ICDAR2015 video

| IoU-based | Contrastive-based | ID\(F_1\) | MOTA | MOTP |
|-----------|-------------------|---------|------|-----|
| ✓         | ✓                 | 57.5    | 41.2 | 74.3 |
| ✓         | ✓                 | 65.8    | 48.9 | 73.5 |
| ✓         | ✓                 | 65.9    | 49.1 | 73.8 |

### Table 3 – Effect of loss functions on ICDAR2015 video

| Name          | \(\tau\) or \(m\) | ID\(F_1\) | MOTA | MOTP |
|---------------|-------------------|---------|------|-----|
| Margin Triplet| 0.05              | 52.6    | 48.0 | 73.4 |
|               | 0.5               | 62.2    | 48.2 | 73.4 |
| NT-Xent       | 0.05              | 64.2    | 48.2 | 73.7 |
|               | 0.5               | 63.4    | 48.5 | 73.8 |

### Table 4 – Effect of backbone and input image shorter side on ICDAR2015 video

| Backbone     | Shorter Side | ID\(F_1\) | MOTA | MOTP | FPS |
|--------------|--------------|---------|------|-----|-----|
| ResNet18     | 512          | 59.0    | 43.1 | 73.4 | 23.3 |
| ResNet18     | 640          | 64.1    | 47.5 | 74.2 | 18.6 |
| ResNet18     | 720          | 65.9    | 49.1 | 73.8 | 16.7 |
| ResNet50     | 512          | 59.4    | 43.6 | 74.0 | 19.7 |
| ResNet50     | 640          | 64.3    | 47.9 | 73.6 | 15.4 |
| ResNet50     | 720          | 66.1    | 49.5 | 73.9 | 13.4 |
4.2. Ablation Study

Here, we conduct four groups of ablation experiments to study the core factors of SVRep.

Semantic, Visual and Positional Encoder. As shown in Table 1, we compare the three encoders on ICDAR2015(video). To focus on the effect of different encoders, we implement all of them in the same backbone (i.e., ResNet) and loss function (i.e., NT-Xent loss). When we only evaluate one encoder (i.e., visual encoder), other two embedding features (i.e., $R_p$, $R_s$) from corresponding encoder would not be used in the discriminating function $F_θ(·)$ for contrastive learning as described in Sec. 3.2. As such, the comparison is solely on the three encoder. The model presents a base performance (i.e., 64.8% ID$_{F1}$ and 48.2% MOTA) with visual encoder. With positional and semantic feature embedding, the model obtains the ID$_{F1}$ of 65.9% and MOTA of 49.1%, with 1.1% and 0.9% improvements, respectively. Especially for the semantic feature embedding, the abundant semantic representations bring obvious improvements (0.9% ID$_{F1}$) by recognizing each text, and relating them.

Loss for Contrastive Learning. We study the effect of the contrastive loss, and comparing the NT-Xent loss against other commonly used contrastive loss functions, such as margin loss [28]. As shown in Table 3, NT-Xent loss shows a better performance, around 4% improvement than the counterpart of Margin Triplet loss. $τ$ is the temperature parameter for NT-Xent loss in Equ. 3, and we set three values to evaluate the effect from it. With the change of $τ$, ID$_{F1}$ as the tracking stability metric, shows a higher volatility than MOTA and MOTP, since $τ$ affect the weight of positive text pair and negative text pairs. In this paper, we set the $τ$ to 0.1 for the competitive performance.

Speed Analysis. Table 4 presents the time cost of SVRep with different backbones and input image shorter side. We evaluate all testing images and calculate the average speed. These results are tested with 1 batch size on one V100 GPU and one 2.20GHz CPU in a single thread. With ResNet18, 512 pixels of image shorter side, model presents the faster speed with 23.3 fps. On the contrary, model shows the best performance (i.e., 66.1% ID$_{F1}$), while the inference speed is slow with 13.4 fps. Besides, we compare with other methods in Figure. 3. Moreover, “Ours+ResNet18” reaches 16.7 FPS, which is 7.7 FPS faster than that of Free [3] and TransVTSpotter [41], while its performance achieves a great improvement with ID$_{F1}$ of 8.6%.

Contrastive Learning vs. IoU-based Matching. Contrastive Learning, as the core of the paper, is the main different with the previous works. As shown in figure. 2, IoU-based
bounding box match is used to compare with our feature-based cosine similarity with contrastive learning. With the same condition, contrastive-based feature cosine similarity match shows a better performance than IoU-based, with at least 8% IDF1 improvement.

4.3. Comparison with State-of-the-arts

We compare SVRep against state-of-the-art methods for video text tracking task on five public benchmarks. Besides, more experiments of video text tasks are provided in the supplementary material.

ICDAR2015(video) and ICDAR2013(video), as the two most popular public benchmarks are used to evaluate our method. As shown in figure 5, our SVRep shows a powerful performance with 65.9% IDF1 and 49.1% MOTA on ICDAR2015(video), achieving 8.6% and 5.0% improvements than the previous SOTA method, respectively. Besides, when the input image shorter side is 720 pixels, with ResNet18, the inference speed of our method reaches 16.7 FPS, which is faster than previous methods, while the F-measure is still very competitive (65.9%).

Minetto, as one small dataset to evaluate the robustness of SVRep. Following the previous works [5, 41], we train on ICDAR2015(video), and evaluating the model on the Minetto dataset directly. Figure 5 presents that SVRep with ResNet18 achieves a better performance (83.9% v.s. 74.7% for IDF1) than TransVTSpotter [41] with a faster inference speech (19.5 fps).

BOVText is a bilingual and large-scale dataset with more than 2 million video frames, which collected from the worldwide user of YouTube and KuaiShou. With ResNet18, our method achieves 75.4% IDF1 for video text tracking task, at least 10% improvements than the previous works. The great performance on the practice benchmark further shows the remarkable robustness and generalization of our SVRep. For the long caption text challenge (text average width-height ratio more than 6.8) on BOVText, the obvious improvement proves the robustness of SVRep for long text tracking.

YVT. YouTube Video Text dataset mainly includes overlay text and scene text (e.g., street signs, business signs, words on shirt). Similar to other datasets, SVRep achieves the state-of-the-art performance with 69.1% IDF1, while maintaining high inference (i.e., 16.2 fps).

5. Conclusion

Unlike the previous works track text by visual feature, we firstly propose an end-to-end video text tracker with semantic and visual representations, which tracking text by exploiting the visual and semantic relationships between texts with contrastive learning in high-dimensional embedding space. Without bells and whistles, SVRep achieves the best result with up to 10% improvement and the highest speed among methods using a single model on the ICDAR2015(video).

| Method                  | Video Text Detection/% | Precision | Recall | F-Score | FPS |
|-------------------------|------------------------|-----------|--------|---------|-----|
| SWT [4]                 | 39.8                   | 32.5      | 35.9   | -       | -   |
| OpticalFlow[45]         | 47.0                   | 46.3      | 46.7   | -       | -   |
| RTD[42]                 | 48.6                   | 54.7      | 51.6   | -       | -   |
| AOMTD[17]               | 57.9                   | 55.9      | 51.7   | -       | -   |
| E2EVTTR[35]             | 58.3                   | 51.7      | 54.5   | -       | -   |
| FRA[30]                 | 61.0                   | 57.0      | 59.0   | -       | -   |
| MOSTD[40]               | 63.0                   | 68.0      | 65.0   | -       | -   |
| OlineTrack[44]          | **82.4**               | 56.4      | 66.9   | -       | -   |
| ASGD[5]                 | 75.5                   | 64.1      | 69.3   | 9.6     | -   |
| Free [3]                | 79.7                   | 68.4      | 73.6   | 8.8     | -   |
| SVRep+ResNet18+S(720)   | 81.2                   | 68.3      | 74.2   | **13.5**| -   |
| SVRep+ResNet50+S(720)   | 80.7                   | **68.6**  | **74.2**| 8.7     | -   |

Table 6 – Video frame text detection performance on ICDAR2013(video). ‘S’: means the shorter side of input image. Our method achieves high inference speed while maintaining competitive accuracy.

| Recognition Head       | IDF1/% | Time(ms) | params($\times 10^5$) |
|------------------------|--------|----------|------------------------|
| CTC-based              | 65.6   | 8.3      | 10.8                   |
| Attention-based        | 65.8   | 21.2     | 10.0                   |

Table 7 – Comparison of different recognition head. CTC recognition head bring faster speed with parallel decoding dataset. To our knowledge, our work is the first one that applies the semantic representation learning to video text tracking. We hope that similar approaches with semantic knowledge can be applied to many more video-and-language tasks in the future.

A. More Experiments

**Video Text Detection.** Table 6 presents the video text detection performance on ICDAR2013 (video) [15]. The proposed SVRep achieves the best result with F-Score of 74.2% among methods without any bells and whistles. Compare with Free [3] using a powerful base detector EAST [47], our SVRep still achieves 0.6% improvement without hand-crafted components such as NMS. Besides, SVRep presents a great inference speed, with at least 10 fps improvement than the previous works. Therefore, we consider that the visual and semantic representation learning from our SVRep can help the network to detect text instances accurately in a video sequence.

B. More Analyses

**CTC-based v.s Attention-based.** There are two benefits for using the CTC-based recognition head: 1) much shorter inference time using parallel decoding. 2) more discrete semantic representation. As shown in Table 2, compare with attention-based recognition head, CTC-based bring faster speech (8.3 ms v.s 21.2 ms), while shows a competi-
Figure 4 – Tracking Visualization for different methods. Compared with Free (top row), our SVRep (bottom row) shows better performance for solving false positive and ID switch problems. The video frames are from ICDAR2015 (video).

Input Image | Attention-based | CTC-based
--- | --- | ---
EUROLUM Heineken | | |
COCKTAIL | | |

Figure 5 – Visualization of semantic feature from different recognition head. CTC-based recognition head learns more sparse features with location information than that of attention-based.

CTC-based recognition head learns more sparse features with location information than that of attention-based. Besides, as shown in Figure 5, CTC-based recognition head presents a sparse feature via learning a non-fixed stream of characters (including repeated characters and blanks). On the contrary, attention-based recognition head captures the information flow within the input sequence to predict the non-fixed output sequence, which automatically filters much sequence information (e.g., blank, fuzzy text). I argue that the information filtered by attention-based is still useful and valuable to distinguish positive pair and negative pairs.

References

[1] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In ICML, pages 1597–1607. PMLR, 2020.

[2] Zhanzhan Cheng, Jing Lu, Yi Niu, Shiliang Pu, Fei Wu, and Shuigeng Zhou. You only recognize once: Towards fast video text spotting. In ACM MM, pages 855–863, 2019.

[3] Zhanzhan Cheng, Jing Lu, Baorui Zou, Liang Qiao, Yunlu Xu, Shiliang Pu, Yi Niu, Fei Wu, and Shuigeng Zhou. Free: A fast and robust end-to-end video text spotter. TIP, 30:822–837, 2020.

[4] Boris Epstein, Eyal Ofek, and Yonatan Wexler. Detecting text in natural scenes with stroke width transform.

In CVPR, pages 2963–2970, 2010.

[5] Wei Feng, Fei Yin, Xu-Yao Zhang, and Cheng-Lin Liu. Semantic-aware video text detection. In CVPR, pages 1695–1705, 2021.

[6] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. Advances in neural information processing systems, 27, 2014.

[7] Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In ICML, pages 369–376, 2006.

[8] Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. TPAMI, 31(5):855–868, 2008.

[9] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In Proceedings of the thirteenth international conference on artificial intelligence and statistics, pages 297–304, 2010.

[10] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In CVPR, volume 2, pages 1735–1742. IEEE, 2006.

[11] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In CVPR, pages 9729–9738, 2020.

[12] Xian-Sheng Hua, Pei Yin, and Hong-Jiang Zhang. Efficient video text recognition using multiple frame integration. In ICIP, volume 2. IEEE, 2002.

[13] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reduc-
frame tracking. In ICDAR, volume 1, pages 1255–1260, 2017.

[40] Liang Wu, Palaiahnakote Shivakumara, Tong Lu, and Chew Lim Tan. A new technique for multi-oriented scene text line detection and tracking in video. TMM, 17(8):1137–1152, 2015.

[41] Weijia Wu, Debing Zhang, Yuanqiang Cai, Sibo Wang, Jiahong Li, Zhuang Li, Yejun Tang, and Hong Zhou. A bilingual, openworld video text dataset and end-to-end video text spotter with transformer. In NeurIPS, 2021.

[42] Xu-Cheng Yin, Xuwang Yin, Kaizhu Huang, and Hong-Wei Hao. Robust text detection in natural scene images. TPAMI, 36(5):970–983, 2013.

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

[44] Hongyuan Yu, Yan Huang, Lihong Pi, Chengquan Zhang, Xuan Li, and Liang Wang. End-to-end video text detection with online tracking. Pattern Recognition, 113:107791, 2021.

[45] Xu Zhao, Kai-Hsiang Lin, Yun Fu, Yuxiao Hu, Yuncai Liu, and Thomas S Huang. Text from corners: a novel approach to detect text and caption in videos. TIP, 20(3):790–799, 2010.

[46] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Tracking objects as points. In ECCV, pages 474–490, 2020.

[47] Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. East: an efficient and accurate scene text detector. In IEEE conference on computer vision and pattern recognition, pages 5551–5560, 2017.

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