Automatic Running Event Visualization using Video from Multiple Camera

by

Priadi Teguh Wibowo

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Thursday June 27, 2019 at 13.00.

Student number:	4625862	
Thesis committee:	Prof. dr. ir. M.J.T. Reinders,	TU Delft, chair
	Dr. Jan van Gemert,	TU Delft, supervisor
	Dr. Anna Villanova	TU Delft, comittee member

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Preface

The report documents the findings of my master thesis work. The main content of the report is the scientific paper. It includes the motivation, methodology, implementation, and results of this research project. This is followed by brief reviews of the background knowledge needed to understand the work.

I would like to thanks my supervisors, Dr. Jan van Gemert and Yeshwanth Napolean, who were always kind, patient and supportive. They provided me constant and effective guidance throughout the discourse while still allowing me to develop and investigate my idea.

Many thanks to my colleagues and friends I met in TU Delft, especially my fellow Indonesian students. We always helped each other through good and bad days. It has been a wonderful journey shared with them. I am also grateful to Lembaga Pengelola Dana Pendidikan (LPDP) RI that has granted me a scholarship so I can continue my study in the Netherlands.

Last but not least, I would like to thank my parents and my sisters, who always pray for my success and always been my biggest supporter. None of this would have been possible without them.

Priadi Teguh Wibowo
Delft, June 2019

Contents

1 Scientific Paper **1**

2 Deep Learning **15**

2.1 Introduction . . . 15
2.2 Convolutional Neural Networks . . . 15
2.2.1 Convolutional Layer . . . 16
2.2.2 Pooling Layer . . . 16
2.2.3 Activation Layer . . . 17
2.2.4 Fully-connected Layer . . . 17
2.2.5 Normalization Layer . . . 17
2.3 Backbone Model . . . 17
2.4 Optimization and Loss Function . . . 18

3 Person Re-identification **19**

3.1 Introduction . . . 19
3.2 Deep learning for Person Re-ID . . . 20
3.2.1 Classification Model . . . 20
3.2.2 Siamese Model . . . 20

4 Supplementary Figures **23**

Bibliography **31**

1

Scientific Paper

Automatic Running Event Visualization using Video from Multiple Camera

Priadi Teguh Wibowo
TU Delft
Delft, The Netherlands
PriadiTeguhWibowo@student.tudelft.nl

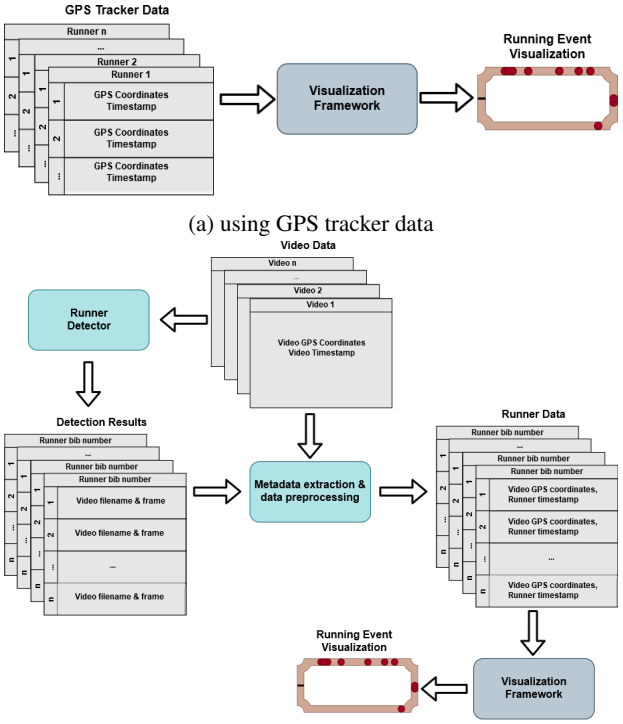
Abstract

Visualizing runners trajectory from video data is not straightforward because the video data does not contain the explicit information of which runners appear in the video. Only the visual information related to the runner, such as runner's unique ID (called bib number), is available. To this end, we propose two automatic runner detection methods, i.e. scene text detection which identifies the runners by detecting their bib number and person re-identification which detects the runners based on their appearance. To evaluate the proposed methods, we create a ground truth database from the video dataset, which consists of video and frame interval information where the runners appear. The video dataset was recorded by nine cameras at different locations during the Campus Run 2018 event. The experimental evidence shows that the scene text recognition method achieves up to 74.05 for F1-score and person re-identification achieves up to 87.76 for F1-score. To conclude, we find that the person re-identification method outperforms the scene text recognition method.

1. Introduction

Running events have gained more popularity due to the increasing awareness of health and its accessibility for all people regardless of their gender, age, or economic status. It is not surprising that there are about 68,832 running races around the world in 2019 [1].

In some marathon events, the athlete's data is collected using GPS tracker for live tracking [30]. Several company vendors for GPS trackers provide a service for data analytics and running race visualization from the race data [5, 6]. Nevertheless, using GPS tracker data can become more expensive as the number of participants grows because every runner must wear a GPS tracker. We aim to utilize the visual data (video) as an alternative approach to visualize the running event. As with the visual data, there could be more



(a) using GPS tracker data

(b) using video data

Figure 1: Pipeline to build a running event visualization.

useful application, either for the runners, spectators, or organizers, such as performance monitoring, live broadcasting, photo tagging, health monitoring, etc. [4]. However, data analytics or other applications are out of the scope of this thesis; we only address the running race visualization.

Although the video data contains the GPS coordinates and timestamps that can be used for race visualization, there is no explicit information about which runner appears in the video. But the video provides visual information of each runner, e.g. runner's ID number or bib number. Thus, we utilize computer vision techniques to detect the runners in

(a) (b)

Figure 2: Examples of runners with their bib number tag in natural scene images. (a) Bib number tags with clear visibility: 405 (green T-shirt), 274 (blue T-shirt), and 8 (white T-shirt). (b) Bib number tag 272 has one of its digits occluded; it looks 27.

video automatically and retrieve the video and frame information where they appear.

The participants in the running event always have their unique identification number (bib number) to identify them during the race. The bib numbers are usually printed on a piece of paper tag, and they are attached at the front of the runner's T-shirt, as shown in Figure 2a. So the most intuitive approach to identify the runner visually is detecting its bib number text [9, 34]. However, during the race, the bib number text might be occluded by the runner's hand, body or by someone else body part. An occlusion at one or more of the digits of bib number text also leads to a false positive, as shown in Figure 2b.

Runner detection also can be viewed from another perspective, i.e. person re-identification (person re-id). Person re-id is a process of identifying the person from images/videos taken from different camera [15]. Instead of using bib number text, person re-id method uses the whole body appearance as its feature to detect person images [7, 22]. Therefore, the person re-id method alleviates the challenges of the scene text recognition method, such as occlusion or low image resolution on bib number text.

In this work, we use scene text recognition and person re-id methods for runner detection task. For the scene text recognition method, we choose deep-learning based text spotter baselines [10, 18]. We also use the text spotter to collect training samples (cropped person images) for person re-id model so that the person re-id method can detect the runners in videos that were recorded by different cameras although the runners might have different appearances under different camera setting (e.g. illumination, viewpoint, pose, etc.) [7]. For person re-id method, we choose Siamese network model [26].

To evaluate the performance of the proposed approach, we created a ground truth database where the videos are annotated with the appearing runners and the frame interval which they belong.

Our main contributions are:

- Running event visualization built from the video data that use runner detection methods to find relevant video and frame and at the same time, the GPS coordinates and individual timestamp for each runner.
- A ground truth database of 5 km category runners from Campus Run 2018 event for evaluating the proposed method.

2. Related Work

Scene Text Recognition Understanding the text in scene image can be useful for a wide range of computer vision applications, e.g. robots navigation or industry automation [14, 28]. The existing scene text recognition methods can be categorized into three types of task, i.e. text detection that detects and localizes the existence of text, text recognition (widely known as Optical Character Recognition (OCR)) that converts text region into linguistic characters, and end-to-end system that performs both text detection and recognition (also called text spotting system or text spotter). To retrieve the text information from an image, we need an end-to-end system. Authors [29, 36] utilized handcraft features (e.g. HOG, color, stroke-width, etc.) to implement a text spotting system. However, because of limited representation ability of handcraft features, these methods barely handle complex scene text image [25], like ICDAR 2015 dataset [21]. Therefore, the deep learning-based end-to-end system or text spotter methods [10, 18, 24, 27] are proposed because they can learn features automatically. Lyu et al. [27] modified the Mask R-CNN to have shape-free text recognition ability. Liu et al. [24] and Busta et al. [10] adopted EAST [41] and YOLOv2 [31] as their detection branch and developed CTC-based text recognition branch. [18] adopted EAST [41] as its text detection branch and developed the attention-based recognition branch. Since in our work, the bib number

text appears on unconstrained scene images (e.g. complex background, varying size, illumination variation, etc.) as shown in Figure 2, so we use the scene text recognition method to detect the bib number text.

Bib Number Detection Using scene text recognition method is an intuitive choice to detect the bib number tag and indirectly identify the runner who wears it. Ben-Ami et al. [9] proposed a pipeline consisting of face detection and Stroke Width Transform (SWT) to find the location and region of bib number tag, and then they applied digit pre-processing and Optical Character Recognition (OCR) engine [35] to recognize the bib number text. Shivakumara et al. [34] proposed to use torso detection and Histogram of Oriented Gradient (HOG) based text detector to find the location and region of bib number tag, and then they use text binarization and OCR engine [35] to recognize the bib number text. Because our focus is not developing an pipeline to detect the bib number, we choose text spotters [10, 18] that publish their code as our scene text recognition baseline to detect the bib numbers. The second reason is that they have a distinct performance; He et al. [18] has a higher F1-score on ICDAR 2015 dataset [21] (around 23%) compared to Busta et al. [10] [25]. We would like to compare their performance in our problem.

Person Re-identification Person re-id is a task of recognizing a pedestrian image over a network of video surveillance cameras with possibly non-overlapping fields of view [22]. Many papers from top conferences (CVPR, ICCV, and ECCV) adopted the deep learning technology as their person re-id method [7]. The authors [37, 38, 39] investigated classification-based CNN model for person re-id task. However, if the dataset lacks training samples per individual, the classification-based model is prone to overfitting [22]. Therefore, a Siamese deep network architecture [12] is proposed for person re-id task. Yi et al. [40] applied Siamese network model with pairwise loss that takes a pair of images to train the model and learn the similarity distance between two images. Hermans et al. [19] showed that using Siamese network model and triplet loss are a great tool for a person re-identification task because it learns to minimize the distance of a positive pair and maximize the distance of a negative pair at once. Luo et al. [26] improved the previous work (Siamese network model with triplet loss) performance with several training tricks, such as center loss or label smoothing. Based on their successful work [26], we adopt it as our person re-id baseline for runner detection task.

3. Data Collection

Before introducing our method, we briefly explain the data we collected related to the running event and used for visualization.

3.1. Video Dataset

The video dataset was recorded during the running event of Campus Run 2018 [1]. The running event was organized at the TU Delft campus, as shown in Figure 3a. At the running track of Campus Run 2018, the video dataset was collected by nine camera phones that have different locations. During the event, the cameras did not always stay at one location; sometimes after recording a video they moved to another location of the running track, but the field of view between the nine cameras is not overlapped.

The video dataset contains the metadata file that has the necessary information for creating the running race visualization. Table 1 shows the description of information from the video metadata. Unfortunately, not all video has GPS coordinates data. So we manually check the visual information of those videos that have no GPS data, predict its location, and add the GPS coordinates for those videos into the database.

3.2. Runners Information

The organizer of the Campus Run 2018 put the result of the running race on their website [2]. By scraping this website, we collected all the runner's information related to the running race. The type and description of runner's published information are shown in Table 1.

Furthermore, there are three categories of race in Campus Run 2018, i.e. 5 km, 10 km, and relay. The participants in 5 km and 10 km category are individual runners. While the participants of the relay category are group runners consist of 4 people running in turn for 10 km distance, so each runner in the relay category run for 2.5 km.

4. Methodology

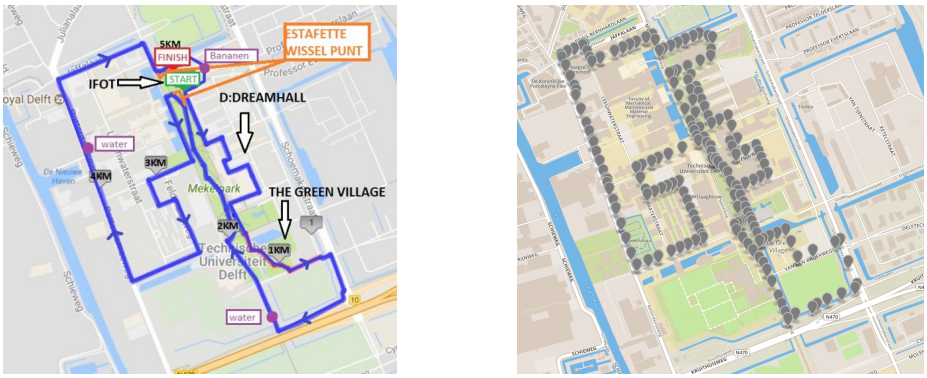
4.1. Running Event Visualization

In this section, we discuss the steps to build a running event visualization using video data.

4.1.1 Running track

The first thing we need is a running track where the virtual athletes will run. The track is formed by an array of GPS coordinates on an interactive map, as shown in Figure 3b. We created the sequence of GPS coordinates using *geojson.io*, a simple web-based tool to create a geospatial data in GeoJSON file format [3]. It provides an interactive map editor to create, view, and share maps by showing the GeoJSON code and its visual output on map side-by-side [17].

[1] https://campusrun.gezelschapleeghwater.nl/en/
[2] https://uitslagen.nl/uitslag?id=2018060621420



(a) (b)

Figure 3: Running event map figures. (a) Campus Run 2018 map [2]. (b) GPS coordinates we collect using *geojson.io* to make a running track for visualization.

Source	Information	Description
Video metadata	Latitude, Longitude	Video GPS coordinates
	Modify Date	The date and time when a video file is created
	Duration	The total duration of a video (seconds)
	Frame rate	The frame frequency of a video (fps)
Campus Run website	Bib number	Text to identify a runner
	Name	Runner's name
	Time	The total duration of a runner to reach the finish line from the start line
	Ranking	Runner's ranking in the same category
	Distance	The distance category (5km, 10km, Relay)
	Speed	The average speed of a runner (km/hour)

Table 1: Information acquired from video metadata and Campus Run website.

4.1.2 Runner timestamp

The important element to visualize the athletes running is the individual timestamp at several different locations. The individual timestamp is determined by the video timestamp and the frame where the runner appears in the video. From video metadata, we use Modify Date t_V and Duration T_V to get the time the video starts recording. Then we convert the the frame f where the runner appears into seconds by dividing it with video frame rate r. To get the individual timestamp, we add the video start time and the converted frame. The formula to calculate individual timestamp is defined as:

$$t_R = t_V - T_V + \frac{f}{r}, \tag{1}$$

Since the runner could appear in multiple frames, we also take into account where the position of a camera and runners to determine which best frame to use. Based on our observation, in most videos, the position of the camera and the athletes resembles Figure 4, in which the runners ran towards the camera. So the last frame where a runner appears would be the most physically accurate choice.

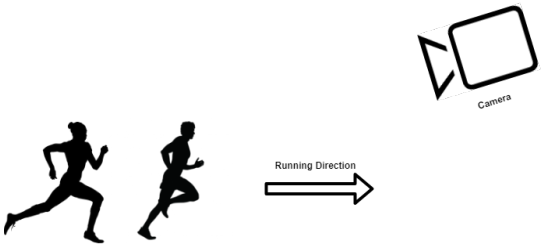


Figure 4: The illustration of position between camera and runners.

4.1.3 Filtering raw GPS data

Figure 5a shows the trajectory of raw GPS data of videos collected by nine cameras, and it can be seen that the cameras did not stay at one location. Although the videos were taken on the running track, the GPS coordinates is not always precisely on the running track. Some of GPS coordinates deviate too far from the original location; sometimes it is on a highway, a river or a building and we call them the jumping points. Therefore, we need to filter the raw GPS data so that it can be used for visualization.

Algorithm 1: Point Matching

Input: P^{raw} (raw video geotag data) and P^{track} (running track points)
Output: P^{pm} (point-matched video geotag data)
for i $\Leftarrow$ 1 to N^{raw}
  for j $\Leftarrow$ 1 to N^{track}
    d_{ij} = earth_distance(P_i^{raw}, P_j^{track})
  m = argmin(d_i)
  P_i^{pm} = P_m^{track}

Algorithm 2: Cosine Law Filter

Input: P^{pm} (point-matched video geotag data)
Output: P^f (refined video geotag data)
for i $\Leftarrow$ 1 to N^f - 2
  θ = angleThreePoints(P_i^{pm}, P_{i+1}^{pm}, P_{i+2}^{pm})
  d_1 = earthDistance(P_{i+1}^{pm}, P_i^{pm})
  d_2 = earthDistance(P_{i+1}^{pm}, P_{i+2}^{pm})
  if $\theta < \theta_{th}$ **and** $d_1 > d_{th}$ **and** $d_2 > d_{th}$

$$\text{avg\_point}_i = \frac{P_{i+2}^{pm} + P_i^{pm}}{2}$$

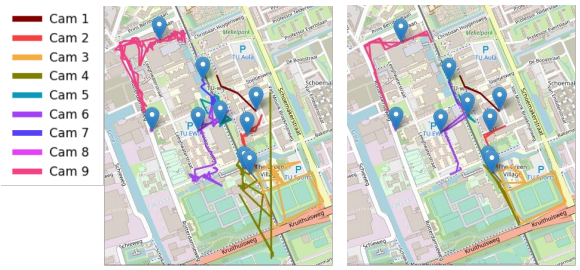
P^f = pointMatch(avg_point, P^{track})

The first filtering step of raw GPS data is quite simple, i.e. replacing the raw GPS coordinates with the nearest running track points as shown by Algorithm 1, so the video GPS coordinates stay in the running track. Afterward, we deal with the jumping points by exploiting cosine law to filter based on the angle and distance formed the jumping point and its two nearby points, as shown in Algorithm 2.

After we apply the filters on raw GPS data, we have a better-looking camera trajectory, as shown in Figure 5b. However, there are still five some jumping points from camera 6 that intersects with the trajectory of camera 5 and 7. We manually predict and replace the remaining jumping points with new GPS coordinates according to what the video shows. Now the filtered video GPS data is ready to use for running event visualization.

4.1.4 Start and finish location

The missing information from the video data is the individual timestamp and GPS coordinates at the start and finish location because there is no video recorded at both locations. So the GPS coordinates and the individual timestamp at the start and finish location are determined by ourselves. We predict the GPS coordinates at the start and finish location based on the start and finish signs in Figure 3a. Meanwhile, the individual timestamp of the start location is determined by zeroing the seconds of the individual timestamp at the camera 1 location. For example, if the individual timestamp at camera 1 is 16:00:10, then the start timestamp is 16:00:00. We assumed that the camera 1 location is only a few meters away (10 - 20 meters) from the start location. The individual timestamp at the finish location is defined by the start time plus the duration required by a runner to finish the race. This duration is obtained from scrapping the Campus Run result website (check Time at Table 1).



(a) (b)

Figure 5: Visualization of camera GPS data. (a) Raw camera trajectory. (b) Filtered camera trajectory. There are nine lines with different color which represents different cameras. The cameras moved according to the line drawn on the map.

By having all the individual timestamps and GPS coordinates at several locations (start, finish, and camera locations), we can visualize the running event. We use javascript library, D3[3] and Leaflet[4] to create the visualization. D3 provides API for animation and Leaflet provides API for interactive maps.

4.2. Runner Detection

In this section, we discuss the proposed methods in detail. In our research problem, the runner detector does not detect a text or a person image *per se*. But it also retrieves the video and frame information where they are detected. We are interested in that information because they are used to visualize the runner's trajectory; the video provides GPS coordinates and video timestamp information, and the frame represents the individual timestamp.

4.2.1 Scene text recognition

We use the end-to-end system or text spotter [10, 18] to identify the runners based on its bib number text. Since we know the bib number of all runners from scrapping the Campus Run website, we do not have to locate the runner's body part first (face [9] or torso [34]) to detect the bib number. So we apply the text spotter directly on the video to detect the bib numbers and perform text retrieval from the video. The steps to use text spotter in our problem is :

[3] https://d3js.org/
[4] https://leafletjs.com/

- we feed the frames of a video into text spotter model,
- we collect all text detected by the text spotter,
- if the bib number we know from scrapping is detected, we retrieve the video and frame information for further evaluation.

The text spotter also collects the training samples for the person re-id model. The training samples are cropped person images from the video dataset. The cropping occurs if the text spotter detects a bib number on the image. Then the bib number is assigned as the label of that cropped image. Cropping a person image is also done programmatically according to the location and the size of the bib number text region.

Although there are two text spotters, we do not merge the training samples collected by the two text spotters. Instead, the person re-id model is trained with each training sample separately so that we compare the results.

4.2.2 Person Re-identification

For the person re-identification method, we choose to adopt the work of Luo et al. [26] which used Siamese network model with triplet loss and cross entropy loss as its baseline. The model is expanded without changing the model architecture by adding a new loss function, i.e. center loss, and other several training tricks, e.g. varying learning rate or label smoothing.

The person re-id model is trained using the cropped person images collected by the text spotter. Meanwhile, to evaluate the person re-id model, the pedestrian images from the video dataset are cropped by an object detector. Those pedestrian images does not have label because the object detector cannot detect a text; we only use it to detects the pedestrian (person) object.

Since there are two text spotters, the person re-id is trained with two training samples separately so that we might have two person re-id model with different performance. Then the person re-id model with the best performance is retrained again to see the possible improvement.

After training the person re-id model, we face three issues, namely:

- the person re-id model we choose is not a classifier, so we cannot directly classify the images from evaluation set using a person re-id model,
- the text spotter produces false positives, so the training samples for person re-id surely have outliers,
- In the evaluation set, the object detector also crops person images that are not runners from the training set.

Therefore, not only to classify the person image, we need to detect whether the person image is an outlier. We incorporate *k-NN* as a classifier and outlier detector. Using *k-NN* to help person re-id method resembles the image matching used in most person re-id task, except *k-NN* use majority vote from k neighbors to classify an image. Also, as an outlier detector, it has a competitive performance compared to other methods [11].

The idea of using *k-NN* as an outlier detector is that the larger the distance of a query point from its neighbor points, the more likely that the query point is an outlier [16]. We adopt the definition of an outlier that considers the average of the distance from its k neighbors [8]. So if the average distance of an image from its k neighbors is larger than a threshold, then it is considered as an outlier.

4.2.3 Performance Evaluation Method

To evaluate the runner detection method, we use the F1-score measure and Intersection over Union (IoU), but we modify both of them to fit our research problem. This section describes the mathematical formulation of the evaluation metrics and their implementation in our problems.

Video-wise metric The video-wise metric is useful to check the number of videos where a runner is detected at least once. Evaluating the performance on the retrieved video information is important because we use the video GPS coordinates to visualize the runners trajectory so we want to retrieve as many as relevant video. At the same time, it is also undesirable to retrieve irrelevant videos (high precision) because the irrelevant videos might have GPS coordinates and timestamps that are not in order with GPS coordinates and timestamps that the relevant videos have. Therefore, it could hinder the runner's trajectory visualization.

In a classification problem, true positive is a correctly predicted image, and false positive is vice versa [**?**]. However, in our problem, the ground truth database consists of the video filename, bib number, and frame. So true positive could be defined as correctly retrieved information (video filename, bib number, frame), and the false positive is defined as vice versa. To evaluate the video-wise metric, the retrieved information we need is only the bib number and its video. We use F1-score for the video-wise metric to to evaluate on both relevant and irrelevant videos retrieved by runner detection methods. The video-wise F1-score formula is defined as followed:

$$\text{Recall}_i^v = \frac{|\text{TP}_i^v|}{|\text{GT}_i^v|}\,, \tag{2}$$

$$\text{Precision}_i^v = \frac{|\text{TP}_i^v|}{|\text{TP}_i^v| + |\text{FP}_i^v|}\,, \tag{3}$$

$$\text{F1-score}_i^v = 2\,.\,\frac{\text{Recall}_i^v\,.\,\text{Precision}_i^v}{\text{Recall}_i^v + \text{Precision}_i^v}\,, \tag{4}$$

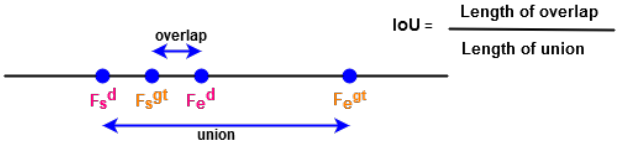


Figure 6: Visualization of frame interval intersection. F_s denotes the frame start, and F_e denotes the frame end. And d denotes that the frame interval belongs to detection result and gt denotes that the frame interval belongs to the ground truth.

where i denotes the runner, v denote a video-wise metric, $|GT|$ denotes the number of runner's video in ground truth, $|TP|$ denotes the total true positive (relevant) videos and $|FP|$ denotes the total false positive (irrelevant) video. The final score is defined as the average over M classes, as shown below :

$$\text{F1-score}^v = \frac{1}{M}\sum_{i=1}^{M}\text{F1-score}_i^v \ . \tag{5}$$

Frame-wise metric We need a frame-wise metric because runner detection methods might produce false positives at an irrelevant frame but at a relevant video. Meanwhile, the video-wise metric cannot evaluate this type of false positive.

We use temporal Intersection of Union (IoU) for the frame-wise metric, which measures the relevant interval of frames where the runner appears. The formula of temporal IoU is similar to regional IoU used in object detection [**?**], except it is only one-dimensional, as shown in Figure 6. The formula for calculating temporal IoU is defined as followed:

$$\text{overlap}_{ij} = \begin{cases} 0 \quad \text{if } \max(F_{s_{ij}}^{gt_{ij}}, F_{s_{ij}}^{d}) > \min(F_{e_{ij}}^{gt}, F_{e_{ij}}^{d}), \\ \min(F_{e_{ij}}^{gt}, F_{e_{ij}}^{d}) - \max(F_{s_{ij}}^{gt}, F_{s_{ij}}^{d}), \end{cases} \tag{6}$$

$$\text{union}_{ij} = (F_{e_{ij}}^{gt} - F_{s_{ij}}^{gt}) + (F_{e_{ij}}^{d} - F_{s_{ij}}^{d}) - \text{overlap}_{ij} \ , \tag{7}$$

$$\text{IoU}_i = \frac{1}{N_i}\sum_{j=1}^{N_i}\text{IoU}_{ij} = \frac{1}{N_i}\sum_{j=1}^{N_i}\frac{\text{overlap}_{ij}}{\text{union}_{ij}} \ , \tag{8}$$

where i denotes the runner, j denotes the video file, and N is the total of video files where the runner detected. $F_{s_{ij}}^{gt}$ and $F_{e_{ij}}^{gt}$ are the frame start and the frame end of runner i at video j from ground truth database. $F_{s_{ij}}^{d}$ and $F_{e_{ij}}^{d}$ is the first frame and the last frame which the runner i is detected at video j. The temporal IoU per runner is the average of the temporal IoU over the total of retrieved video. If the detection happens at an irrelevant video j, then IoU_{ij} is zero, so it will lower the IoU_i. The final score is calculated as average over M classes, expressed as :

$$\text{mIoU} = \frac{1}{M}\sum_{i=1}^{M}\text{IoU}_i \ . \tag{9}$$

5. Experiments

5.1. Datasets

5.1.1 Ground truth database

Ground truth is needed to evaluate the performance of the proposed approach. It is done through video analysis by defining a range of frames for each runner where he or she is recognizable by a human eye. Due to the massive amount of data and time constraint, only runners from the 5 km category that are fully annotated. Consequently, only videos with 5 km runners (with total of 127 persons) are annotated as ground truth for evaluation.

The ground truth is established by a range of frames where the runners appear. The ranged is defined by the "frame start" and "frame end" for every runner in the video. The frame start is annotated when a runner starts to be recognizable by human-eye. Meanwhile, the frame end is annotated when a runner is out of frame.

Figure 7 shows that runner 156 is first seen at the right edge of the screen at frame 98, so this frame is annotated as frame start for runner 156. Then at frame 232, runner 156 is seen for last time at the video, then this frame is annotated as frame end for runner 156. Meanwhile, for runner 155, the frame start is 96, and the frame end is 248. The annotation result can be seen in Table 2.

Camera	Video	Bib number	Frame start	Frame end
Cam_x	VID_xxx.mp4	155	96	169
Cam_x	VID_xxx.mp4	156	98	248

Table 2: An example of ground truth database of a video with two runners

5.1.2 Evaluation set for person re-id

The person re-id model cannot localize and recognize a person at the same time from a whole frame with multiple objects and background images. It can only recognize a single cropped person image. Therefore, the training and evaluation set must be a collection of cropped person images, not a collection of videos.

Figure 7: An example of a video with frame number for annotation.

We already have the training samples for the person re-id model from the text spotter. So to create an evaluation set for the person re-id, the person images from video dataset are cropped and collected by an object detector. The difference between the person images collected by the object detector and the person images collected by text spotters is that the person images from the object detector do not have a label. So object detector crops any pedestrian images from a video regardless of whether the person has a bib number or runners with occluded bib number text. However, we still store the video and frame information of the cropped pedestrian images for the performance evaluation.

There are a lot of famous object detection method out there, such as Faster R-CNN [32] R-FCN [13] and SSD [23]. However, comparison experiments [20] show that Faster R-CNN has better accuracy compared to the other two if the speed is not a concern. Also, SSD is bad at detecting small objects. Since to create an evaluation, we do not consider speed and we need to detect small images, we choose Faster R-CNN.

5.2. Implementation Details

For text spotter [10, 18] and the object detector Faster R-CNN, we do not reproduce the model but use the available trained model from the author. Meanwhile, to train the person re-id model, we use the same configurations as the author [26], such as using Adam optimizer [33] and the total epochs of 120. Then, the person re-id model use the training samples collected by text spotter.

Additionally, we collect a new training samples to retrain the person re-id model from the evaluation set. We choose the cropped images that are considered as inliers by the previous person re-id model.

For *k-NN* model, we choose $k = 5$. We also cosine distance for the *k-NN* model as its distance metric because using cosine distance during the inference stage is better than using Euclidean distance for person re-id task [26].

The *k-NN* model is trained on the embedding features extracted by the person re-id model. The features are extracted from the training samples (person images) used to train the person re-id model. So the person re-id model is used only as a feature extractor.

5.3. Performance Analysis

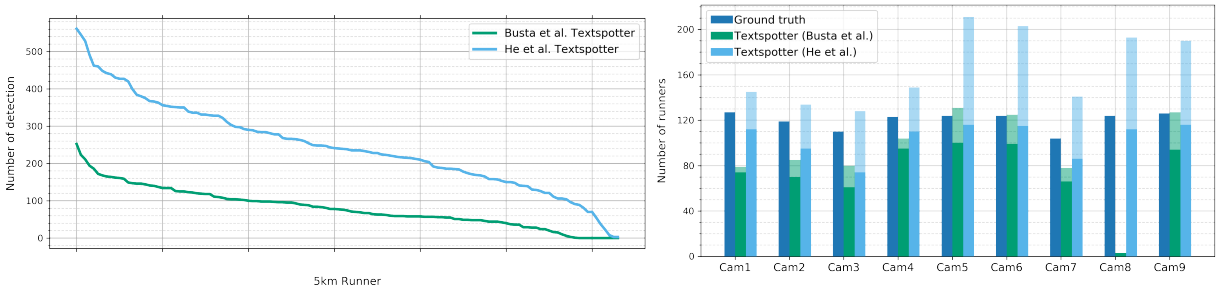
5.3.1 Runner detection: scene text recognition

As expected, due to higher performance on ICDAR2015 dataset, He et al. [18] has has more number of detection compared to Busta et al. [10], as shown in Figure 8a. The plot also shows that Busta et al. [10] has difficulty in detecting bib numbers with digits lower than three. Meanwhile, He et al. [18] is slightly better; it only has difficulty in detecting bib numbers with a single digit. Another finding is that runner with bib number 431 and 432 has a low number of detection, although it has three digits. Runner 431 and 432 put their bib number tag on their back, as shown in Figure 9a and 9b. So the text spotter only detects them at a video that records the runners facing against the camera.

Text Spotter	Recall^v	Precision^v	F1-score^v
Busta et al. [10]	61.03	76.85	66.64
He et al. [18]	86.41	68.41	74.05

Table 3: The average performance of text spotter on our dataset.

Despite the large number of detection, the average Precision^v of text spotter [18] is low as shown in Table 3. Consequently, text spotter [18] produces more false positives, as shown in Figure 8b. It happens because the text



(a) Number of detections per runner.

(b) Number of runners detected per camera.

Figure 8: Text spotter result comparison. (a) The x-axis is runners sorted on the number of detection. The x-axis values are not shown because there are two different x-axes. (Check Section 4 Supplementary Figures for each plot). (b) The stacked bars with lighter color is the false positives produced by the text spotter.

spotter [18] is so good at detecting text. So even if a small part of bib number text is occluded, the text spotter still detects it. But because the occluded bib number looks like another bib number, it will count as false positive for runner detection task, as shown in Figure 9c and 9d.

(a) (b) (c) (d)

Figure 9: Runners with bib number on their back: (a) Runner 431 and (b) Runner 432. False positives from text spotter: (a) bib number 238 detected as 230 and (b) bib number 400 detected as 100.

5.3.2 Runner detection: person re-identification

Distance threshold A threshold is determined to separate the inliers and outliers for the person re-id method. We use F1-score^v determines as a comparison metric between different threshold. The larger the.The performance of person re-id method at different thresholds is given in Table 4. Because the person re-id with training samples collected by He et al. [18] has higher performance, the detection results from this person re-id model are used to train the model for the second time. The performance of retrained person re-id methos is shown in Table 5. We choose each person re-id model with highest F1-score^v to compare with each other and the text spotter.

Comparative results: F1-score^v We also report the F1-score^v between two text spotters and person re-id that are trained with three different training samples. Figure 10 shows that the person re-id group achieves higher performance compared to the text spotter group. It validates our hypothesis that using the whole person appearance is better for runner detection; the person re-id achieves up to 87% for F1-score^v and it means that the person re-id can collect the relevant video almost as good as the ground truth. However, second round training for person re-id only improves the F1-score^v slightly by 2%. Then larger samples from He et al. [18] do not guarantee a significant improvement for person re-id as it only increases the F1-score^v by 7%.

Another important observation is that at the F1-score^v at the lower right side of the plot many, those runners have zero F1-score^v. They are the runners with digits lower than three, so the text spotter hardly detect them, e.g. runner 7 or 9. Consequently, the person re-id models do not have training samples for those runners and have zero performance as well.

Comparative results: temporal IoU To show the performance in retrieving the relevant frames, we present the temporal IoU plot for the text spotter and person re-id, as shown in Figure 11. The person re-id still outperforms the text spotter. It happens because the person re-id use the training samples from the text spotter and it can expand the frame interval by detecting runner from earlier or later frame where the bib number text might be occluded, as shown by Figure 12. Although there is an improvement in the second training, the increase is only 6%. We expected a wider expansion

Distance threshold	0.3	0.25	0.23	0.22	0.21	0.19	0.17	0.15
Person re-id (Busta et al. [10] samples)	76.34	78.80	78.98	78.75	**79.00**	77.53	75.64	72.88
Person re-id (He et al. [18] samples)	79.48	83.90	85.10	**85.77**	85.72	85.55	84.31	80.35

Table 4: F1-score^v at different distance threshold for person re-id model with different training samples. The bold number is the highest F1-score^v.

Distance threshold	0.09	0.08	0.07	0.06	0.05	0.04	0.03	0.02
Person re-id (2^{nd} round)	86.57	86.95	86.83	87.59	**87.76**	86.47	84.13	74.11

Table 5: F1-score^v at different distance threshold for person re-id model trained for the second time. The bold number is the highest F1-score^v.

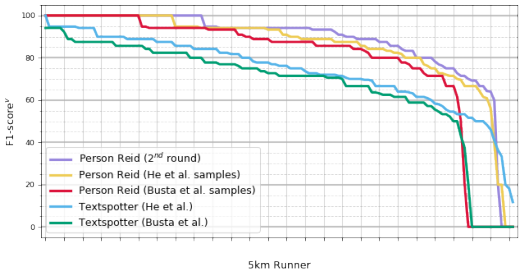


Figure 10: The comparison of F1-score^v between person re-identification and text spotter per runner. The x-axis is runners sorted on its F1-score^v. The x-axis values are not shown because there are five different x-axes. (Check Section 4 Supplementary Figures for each F1-score^v plot)

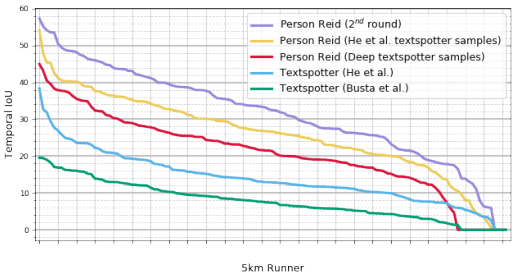
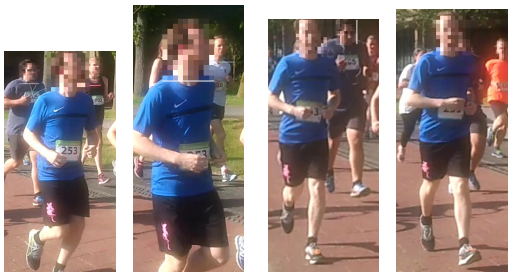


Figure 11: The comparison of temporal IoU between person re-identification and text spotter [18] per runner. The x-axis is runners sorted on its temporal IoU. The x-axis values are not shown because there are five different x-axes. (Check Section 4 Supplementary Figures for each temporal IoU plot).

since the previous person re-id has more relevant detection compared to text spotter.

It is important to notice that some runners have zero temporal IoU. They are the same runners have zero F1-score^v and have bib numbers that text spotter hardly detects.

Outlier detection In Figure 14, we show the general trend of outlier detection performance of person re-id and *k-NN*. It can be seen that the number of false positives produced by He et al. [18] is quite high. However, the person re-id method can significantly produce much less false positives although it use the training samples from the text spotter with many false positives. Unfortunately, the outlier detector does not work well for the second training. The reason could be that more false positives are included in the second training set so that they are hardly detected as outliers.



(a) (b) (c) (d)

Figure 12: Person re-id method can detect runner 253 although his bib number is occluded. (a) The bib number 253 is visible. (b), (c), (d) The bib number 253 is fully occluded

2D timeline visualization Figure 13 shows the true positives and false positives more clearly. Any brown strips

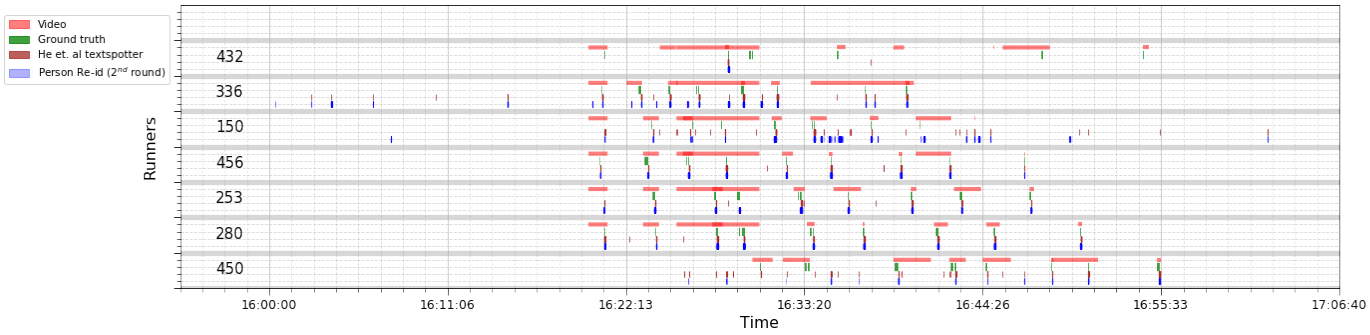


Figure 13: 2D timeline visualization of runner detection. We choose to show the detection result of person re-id model with the best performance and its corresponding text spotter. (Check Section 4 Supplementary Figures for all runners visualization).

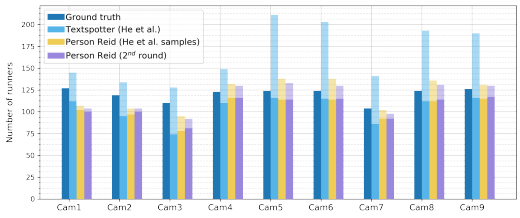


Figure 14: The comparison of the number of detected runners per camera between text spotter [18] and the corresponding person re-id models. The stacked bars with lighter color are false positives. (Check Section 4 Supplementary Figures for the comparison the number of detected runners between text spotter [10] and the corresponding person re-id).

that are not aligned with green strips are false positive from text spotter and vice versa. Meanwhile, any blue strips that are not aligned with green strips are false positive from the person re-id and vice versa. Here we presents the visualization of three highers performance runners, with bib numbers 280, 253, and 456, and four lowest performance runners with bib number 432, 450, 336 and 150. It can be seen that runners 280, 253, and 456 have all their blue strips aligned with their green strips. Meanwhile, runners 336 and 150 have many false positives as many their blue strips are not aligned with their green strips.

6. Limitation

From the experiment, it is important to notice that the performance of the person re-id method depends heavily on the performance of scene text recognition because the scene text recognition collects the training samples for person re-id. For example, person re-id cannot detect the runners that have the bib number with digit lower than three. Another case is the false positives in the training set that are collected by the text spotter. They hinder the performance of person re-id method. Figure 15 shows the false positives produced by the text spotter [18] on runner 450. But the person re-id method fails to detect them as outliers.

(a) (b) (c) (d)

Figure 15: Runners 450 (a) with his false positives. (b) Runner 459. (c) Runner 456 (d) Runner 460.

7. Conclusion and Future Work

In this study, we have proposed an automatic approach to create a running event visualization from the video data. We use scene text recognition and person re-identification method to detect the runners and retrieve the videos and frames information where the runners appear so that we can use the individual timestamp and filtered GPS coordinates from the retrieved data for visualization. The experiments show that the scene text recognition method encounters many challenges in runner detection task. The results also show that the performance of the person re-id method

outperforms the scene text recognition method. The person re-id method can retrieve the relevant video information almost as good as the ground truth. However, the proposed method needs improvement to overcome the false positives or outliers in the training set of person re-id. In addition, it is worth further research to see if using another means to collect training samples for person re-id, such as crowdsourcing labeling, can produce the performance of person re-id.

References

[1] ahotu marathons: International running calendar and runners' resources, . URL https://marathons.ahotu.com/.

[2] The green run, . URL https://campusrun.gezelschapleeghwater.nl/en/.

[3] Geojson, . URL https://geojson.org/.

[4] The latest and greatest in marathon technology, . URL http://www1.pic2go.com/news/latest-greatest-marathon-technology.

[5] . URL https://blog.racemap.com/.

[6] Gps tracking of outdoor sports events, . URL https://tech4race.com/wordpress/en/home/.

[7] Deep learning-based methods for person re-identification: A comprehensive review. *Neurocomputing*, 337:354 – 371, 2019. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2019.01.079.

[8] F. Angiulli and C. Pizzuti. Fast outlier detection in high dimensional spaces. In *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery*, PKDD '02, pages 15–26, 2002. ISBN 3-540-44037-2.

[9] I. Ben-Ami, T. Basha, and S. Avidan. *Racing Bib Number Recognition*. 01 2012. doi: 10.5244/C.26.19.

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

[11] G. O. Campos, A. Zimek, J. Sander, R. J. Campello, B. Micenková, E. Schubert, I. Assent, and M. E. Houle. On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study. *Data Min. Knowl. Discov.*, 30(4):891–927, July 2016. ISSN 1384-5810.

[12] S. Chopra, R. Hadsell, and Y. LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1, June 2005. doi: 10.1109/CVPR.2005.202.

[13] J. Dai, Y. Li, K. He, and J. Sun. R-FCN: object detection via region-based fully convolutional networks. *CoRR*, abs/1605.06409, 2016.

[14] G. N. Desouza and A. C. Kak. Vision for mobile robot navigation: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2):237–267, 2002.

[15] A. Gala and S. K. Shah. A survey of approaches and trends in person re-identification. *Image Vision Comput.*, 32:270–286, 2014.

[16] S. Garcia, J. Derrac, J. Cano, and F. Herrera. Prototype selection for nearest neighbor classification: Taxonomy and empirical study. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3):417–435, March 2012. ISSN 0162-8828. doi: 10.1109/TPAMI.2011.142.

[17] B. A. Hanson and C. J. Seeger. Creating geospatial data with geojson.i, Oct 2016.

[18] T. He, Z. Tian, W. Huang, C. Shen, Y. Qiao, and C. Sun. An end-to-end textspotter with explicit alignment and attention. *CoRR*, abs/1803.03474, 2018.

[19] A. Hermans, L. Beyer, and B. Leibe. In defense of the triplet loss for person re-identification. *CoRR*, abs/1703.07737, 2017.

[20] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, I. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. *CoRR*, abs/1611.10012, 2016.

[21] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, S. Ghosh, A. Bagdanov, M. Iwamura, J. Matas, L. Neumann, V. R. Chandrasekhar, S. Lu, F. Shafait, S. Uchida, and E. Valveny. Icdar 2015 competition on robust reading. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 1156–1160, 2015.

[22] B. Lavi, M. F. Serj, and I. Ullah. Survey on deep learning techniques for person re-identification task. *CoRR*, abs/1807.05284, 2018.

[23] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg. SSD: single shot multibox detector. *CoRR*, abs/1512.02325, 2015.

[24] X. Liu, D. Liang, S. Yan, D. Chen, Y. Qiao, and J. Yan. FOTS: fast oriented text spotting with a unified network. *CoRR*, abs/1801.01671, 2018. URL http://arxiv.org/abs/1801.01671.

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

[26] H. Luo, Y. Gu, X. Liao, S. Lai, and W. Jiang. Bag of tricks and A strong baseline for deep person re-identification. *CoRR*, abs/1903.07071, 2019.

[27] P. Lyu, M. Liao, C. Yao, W. Wu, and X. Bai. Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes. *CoRR*, abs/1807.02242, 2018. URL `http://arxiv.org/abs/1807.02242`.

[28] M. M.aftabchowdhury and K. Deb. Extracting and segmenting container name from container images. *International Journal of Computer Applications*, 74(19):1822, 2013. doi: 10.5120/13001-0039.

[29] L. Neumann and J. Matas. On combining multiple segmentations in scene text recognition. pages 523–527, 08 2013. doi: 10.1109/ICDAR.2013.110.

[30] H. Pielichaty. *Events project management*. Routledge, 2017.

[31] J. Redmon and A. Farhadi. Yolo9000: Better, faster, stronger. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6517–6525, 2017.

[32] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'15, pages 91–99, 2015.

[33] S. Ruder. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016.

[34] P. Shivakumara, R. Raghavendra, L. Qin, K. B. Raja, T. Lu, and U. Pal. A new multi-modal approach to bib number/text detection and recognition in marathon images. *Pattern Recogn.*, 61(C):479–491, Jan. 2017. ISSN 0031-3203.

[35] R. Smith. An overview of the tesseract ocr engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 2, pages 629–633, Sep. 2007. doi: 10.1109/ICDAR.2007.4376991.

[36] K. Wang, B. Babenko, and S. Belongie. End-to-end scene text recognition. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 1457–1464, 2011. ISBN 978-1-4577-1101-5.

[37] L. Wu, C. Shen, and A. van den Hengel. Deep linear discriminant analysis on fisher networks: A hybrid architecture for person re-identification. *CoRR*, abs/1606.01595, 2016.

[38] S. Wu, Y. Chen, X. Li, A. Wu, J. You, and W. Zheng. An enhanced deep feature representation for person re-identification. *CoRR*, abs/1604.07807, 2016.

[39] T. Xiao, H. Li, W. Ouyang, and X. Wang. Learning deep feature representations with domain guided dropout for person re-identification. *CoRR*, abs/1604.07528, 2016.

[40] D. Yi, Z. Lei, S. Liao, and S. Z. Li. Deep metric learning for person re-identification. In *2014 22nd International Conference on Pattern Recognition*, pages 34–39, Aug 2014. doi: 10.1109/ICPR.2014.16.

[41] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, and J. Liang. EAST: an efficient and accurate scene text detector. *CoRR*, abs/1704.03155, 2017.

2

Deep Learning

2.1. Introduction

Deep learning method has been the backbone of the advancement of many applications, such as computer vision, natural language processing, and speech recognition. Deep learning itself is a branch of machine learning that consists of layers of artificial neural networks. The basic computational unit of a deep neural network is a neuron, as shown in Figure 2.1.

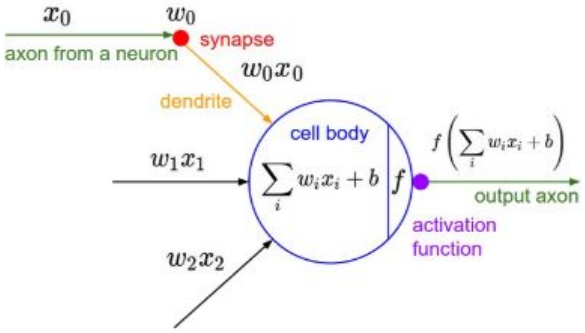


Figure 2.1: Mathematical model of a neuron [2]

The neuron takes input x_i or feature values weighted by the learnable parameter w_i. The weight w_i controls the strength of influence and the direction (positive or negative) of the incoming signal. The operation happens in a neuron is a dot product between w_i and x_i, with an additional bias to avoid zero output. Finally, an activation function f is fed with the output of the dot product to add the non-linear characteristic.

Neurons are stacked vertically or horizontally to form a neural network architecture. Figure 2.2 shows an example of the most common type of artificial neural network i.e. fully-connected layer, in which each circle represents a neuron. In this network, a neuron is fully connected with the neurons from previous layers but share no connection with other neurons in the same layer. The link or connection between neurons in the network is weighted.

Other essential terms used in Figure 2.2 are the input layer which receives the data or feature, the hidden layer which bridges the input and output layer and does all the non-linear computation, and the output layer which produces the results, such as classes or real number. So "deep" in the neural network means that the network has more hidden layers instead of just one.

2.2. Convolutional Neural Networks

Convolutional Neural Networks (ConvNet) is a type of neural network that works well with images. Recently, ConvNet is the primary method for many computer vision tasks, such as image classification, image retrieval, object detection, and recognition. Meanwhile, a regular feedforward or fully-connected neural networks are not used for dealing with image data because it does not scale well with images.

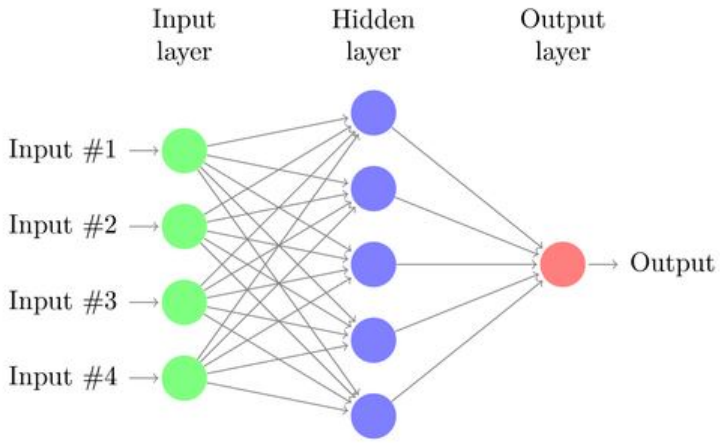


Figure 2.2: Fully-connected neural network [14]

Instead of having a full connection with all image pixels, ConvNet only has a small size kernel to learn. The kernel is convoluted across the image. So the network learns a filter, meanwhile the traditional algorithm design a hand-engineered filter. Thus, ConvNet gives advantages of minimum pre-processing and minimum prior knowledge or human intervention.

The ConvNet architecture consists of a sequence of layers that transforms the original image into the final class score (for classifying task). The types of layers which build the ConvNet are Convolutional layer, Pooling layer, Fully-connected layer, Activation layer, and Normalization layer. Figure 2.3 shows an example of a complete ConvNet architecture for classification task.

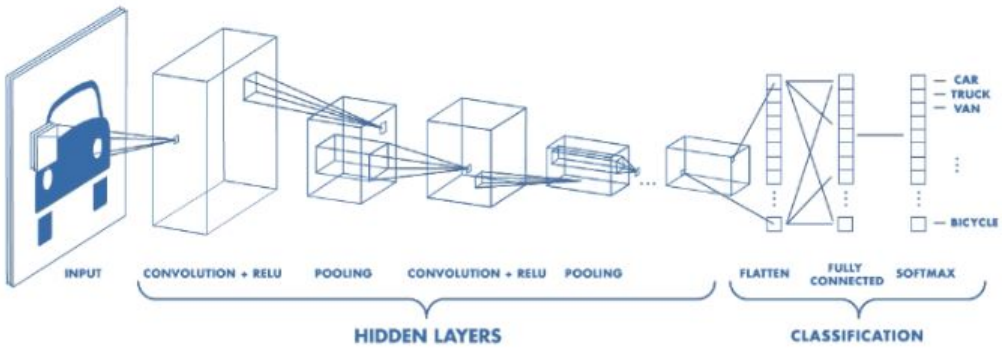


Figure 2.3: ConvNet architecture [1]

2.2.1. Convolutional Layer

The convolution layer is the main element in ConvNet architecture. It consists of a set of learnable kernels with spatially small size (usually a 3x3 or 5x5 matrix), but with full connection along with the depth of input volume. The kernel will do a convolution operation on the image, as shown in Figure 2.4.

As the kernel slides over the input, the convolutional layer produces feature maps that contain the response of the kernel at every position of the input. The goal of the convolution process is so the network learns the filter that can identify a particular visual pattern.

2.2.2. Pooling Layer

The pooling layer reduces the spatial dimension of input but not its depth. The intuition of reducing the spatial dimension is to get the higher level features or pattern which has translation invariance. The other advantage of having a pooling layer is reducing computation burden because of less spatial information and less parameter.

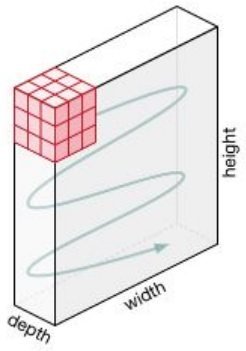


Figure 2.4: Kernel movement [16]

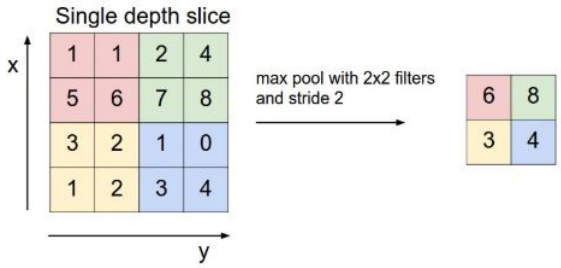


Figure 2.5: Pooling operation [2]

There are two types of pooling layer, i.e. max-pooling which returns the maximum value of the window and average-pooling which returns the average value of the window. The windows size is spatially small, typically $2x2$. The pooling layer operates by sliding the window over the input and applying MAX or AVG operation, as shown in Figure 2.5.

2.2.3. Activation Layer

Activation layer takes a value and passes it into a non-linear function that squashes the value into a range. The most popular activation function is ReLU, which is the abbreviation of rectified linear unit. It calculates the function $f(x) = max(0, x)$. The advantage of ReLU is that it accelerates the convergence of the gradient decent during training [11], and it performs a cheap operation as it simply thresholding the input. The other one is sigmoid function that has the mathematical form $f(x) = \frac{1}{1+e^{-x}}$. However, the sigmoid function is less popular due to the saturation at large value and non-zero centered output [2].

2.2.4. Fully-connected Layer

Fully-connected layers are the last part of classification-based ConvNet architecture. In this layer, the neurons have a full connection to all feature maps from the previous layer. Meanwhile, their output is computed by matrix multiplication between the weights and the input, followed by a bias offset. This layer learns the non-linear combination of the high-level features from the previous convolutional layers. The network uses the output of fully-connected layers and the Softmax classifier to predict the class probability of an image.

2.2.5. Normalization Layer

More modern ConvNet architecture, like ResNet, incorporates batch normalization (BN) layer [9] as a means to do pre-processing at every layer of the network. The benefit of using batch normalization is the robustness towards bad parameters initialization and covariance shift.

First, BN normalizes the output from the previous layer by subtracting it with the batch standard deviation and batch mean. Then BN introduces new learnable parameters, gamma and beta, that scale and shift the normalized value. Since BN normalize the features value (so no feature has a higher influence due to the larger range of value), a higher learning rate can be employed.

2.3. Backbone Model

Instead of building a convolutional neural network from scratch, there are various of ConvNet architecture available which are ready to use for a range of application. We can also use those architectures as a pre-trained model; the model parameters are initialized with the weights that have been optimized on another dataset, like ImageNet [5]. Some of the ConvNet architecture are LeNet, AlexNet, VGG, ResNet, etc. In this section, we are discuss ResNet [6] as it will be used for person re-identification task and considered as the default choice of ConvNet architecture [2].

The core idea of ResNet is the *skip connection* and a heavy use of batch normalization. The skip connection is essentially bypassing the input that skips one or several layers to addition operation, as shown in Figure 2.6. The problem that ResNet tries to solve is the degrading performance of a model with deeper layers due to the vanishing gradient. In other words, the model is unable to learn because gradient shrinks to zero

as it is back-propagated to earlier layers. The skip connection allows the gradient to flow from later layers to the previous layers that have a connection.

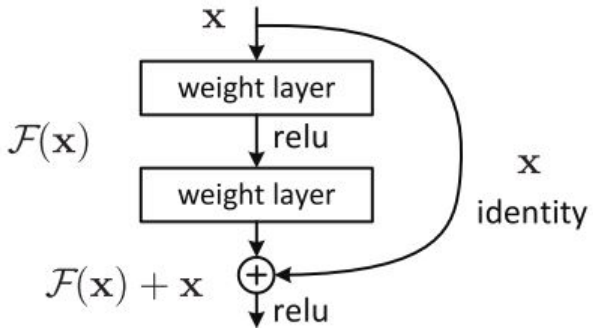


Figure 2.6: Residual Block [6]

2.4. Optimization and Loss Function

One of the most common loss functions for classification tasks in deep learning is cross-entropy loss. The total loss function is the average of loss function for every sample, $L = \frac{1}{N}\sum_i L_i$, where N is the total number of training data. The cross-entropy loss takes the form :

$$L_i = -log\left(\frac{e^{f_{y_i}}}{\sum_j e^{f_j}}\right), \tag{2.1}$$

where f is the output layer of the neural network or a vector of the class score and y_i is ground truth class. The fraction term inside the log function is a softmax function that squashes the input value into a range between 0 and 1. The softmax function gives an intuitive interpretation of loss function; it outputs the normalized class probabilities, so cross-entropy loss increases if the prediction probability deviates from the ground truth class. The regularization term is frequently included into the loss function to prevent the model from over-fitting. The most common regularization form are L1 ($\lambda \| w \|$) and L2 ($\frac{1}{2}\lambda\, w^2$).

After defining the loss function, the next step is minimizing the loss function through an optimization scheme. The first step of optimization in deep learning is backpropagation, which computes the gradient of the loss function with respect to each model parameter through the partial derivative. The gradient tells the sensitivity of the loss function to the parameters. Thus, it allows us to know how much the parameters need to change so that the loss is minimized.

Afterward, the Gradient Descent algorithm uses the gradient computed from backpropagation to update the model parameters. The equation of Gradient Descent takes form as following:

$$\theta = \theta - \eta \,.\, \nabla_\theta L(\theta)\,. \tag{2.2}$$

where θ is the model parameters, η is the learning rate, and $L(\theta)$ is the loss function. The optimization process is done iteratively until convergence. In other words, the weights and biases are updated in favor of finding a global or local minimum of the loss function.

For a large dataset, the mini-batch Gradient Descent is preferred. So instead of calculating the loss function on the whole training data, the parameter updates are calculated based on samples with batch size. On every iteration, the batch training samples are taken from the shuffled training dataset. Furthermore, there are other variants of gradient descent algorithm that have an adaptive learning rate parameter, such as Adam, RMSprop, Adagrad [15].

3

Person Re-identification

3.1. Introduction

In recent years, the intelligent video surveillance system has become an essential technology for public security, such as crime investigation or safeguarding a restricted area. Hence CCTV camera is installed at every corner of public space. However, the data is so massive that it is impossible for a human to analyze the video. The successful computer vision and machine learning techniques could help to extract meaningful information from the large surveillance data; be it online application, like person detection/tracking and recognizing suspicious behavior, or offline application like retrieving the image of the individual of interest. The method that we discuss is an analysis tool for video surveillance, called person re-identification (person re-id).

The goal of person re-id is given a probe image, finding a matching person image from a gallery of images that are collected by multiple cameras with non-overlapping field of views, as shown in Figure 3.1. The most widely use features in person re-id clothing appearance [12]. Biometric feature, like face and gait, is not feasible to exploit because of the uncontrolled environment and low-resolution image [3].

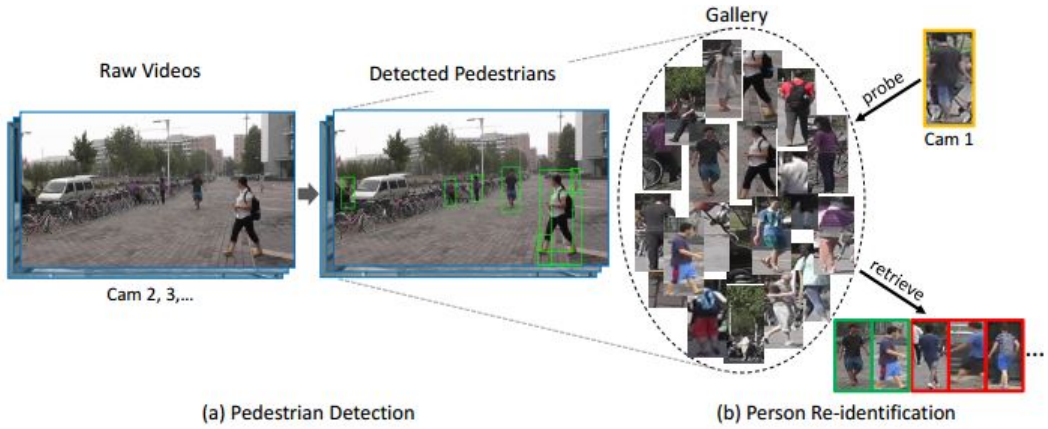


Figure 3.1: Finding a person in gallery set [18]

The main challenge in person re-id is the intra-class variation, i.e. the variation of person's appearance under different camera views. The other challenges are:

- Low resolution: it is harder for the person re-id algorithm to work because of the lack of "information."
- Uniform clothing: it will confuse the person re-id algorithm to distinguish different identities with the same appearance.
- Illumination changes: it causes the same subject to have a different color or appearance.
- Occlusion: partial or full occlusion on a subject image will make it harder to extract the features.

3.2. Deep learning for Person Re-ID

Based on the top three conferences (CVPR, ICCV, and ECCV) in recent years, most of the person re-id papers are based on the deep learning method [3]. Thus we discuss the deep learning implementation for person re-id task in this section. There are two main types of deep learning models practiced in person re-id research, i.e. classification-based and Siamese network model.

3.2.1. Classification Model

Similar to the multi-class deep learning model, the classification-based person re-id model predicts the probability of the corresponding class of individual images. However, the drawback of the classification model is that if the dataset has small training samples per individual, then it might lead the model into over-fitting [12].

The regular loss function to train the classification-based network is a cross-entropy loss. Some other loss function is also incorporated to improve performance. Wen et al. [17] proposed to use centre loss for face recognition task to reduce the intra-class variance. The center loss tries to push the image embedding to its embedding center so that the variation between images of the same class is reduced. Then Jin et al. [10] combined the classification loss and center loss to train the person re-id model to reduce the intra-class variance.

3.2.2. Siamese Model

Siamese network is preferred for person re-id tasks due to lack of training samples. Siamese model itself is a type of convolutional neural network that has two or more identical sub-network which share the same model parameters [12]. The output of the Siamese model is a similarity distance between the input images. The distance metric could be Euclidian distance or cosine function.

There is two types of Siamese model, i.e. Pairwise model and Triplet model. The Pairwise model takes a pair of images as input and computes the similarity score between them. An example of a loss function to train the Pairwise model is cosine similarity loss, as shown in Equation 3.1. Cosine similarity loss function maximizes the cosine value for a positive pair and minimizes the cosine value for a negative pair. The total loss function would be $L = \frac{1}{N}\sum_{i=1}^{n} L_i$.

$$L_i(I_i^1, I_i^2, y_i) = \begin{cases} max(0, cos(I_i^1, I_i^2) - m) & \text{if } y_i = 1 \\ 1 - cos(I_i^1, I_i^2) & \text{if } y_i = -1 \end{cases} \tag{3.1}$$

The combination between the classification model and the pairwise model for person re-id task has a promising result. Zheng et al. [19] proposed to use classification loss (cross-entropy loss) and pairwise model loss to train the Caffe-Net for person re-id task. e

Meanwhile, the triplet model takes three images at the same time, i.e. an anchor (reference) image, a positive pair image, and a negative pair image. The triplet loss function is used to train this model, which make the distance between the same person smaller and the distance between different person larger, as shown in Figure 3.2. To improve the performance of the Triplet model, some authors proposed to use both triplet loss and classification loss [13]. The triplet loss function takes form as following:

$$L = \frac{1}{N}\sum_{i=1}^{n} max\left(\left\| F(I_i) - F(I_i^+) \right\|^2 - \left\| F(I_i) - F(I_i^-) \right\|^2, C \right) \quad , \tag{3.2}$$

where $F(.)$ is the feature generated by the network and $\|.\|^2$ is L2 norm. During training, hard-mining strategy [8] is used to form the triplet units. So the algorithm picks P class identities randomly then samples K images from each class; thus, it forms a mini-batch contains PK images. Afterward, the algorithm selects the hardest negative sample and the hardest positive sample from the mini-batch to form the triplet pair.

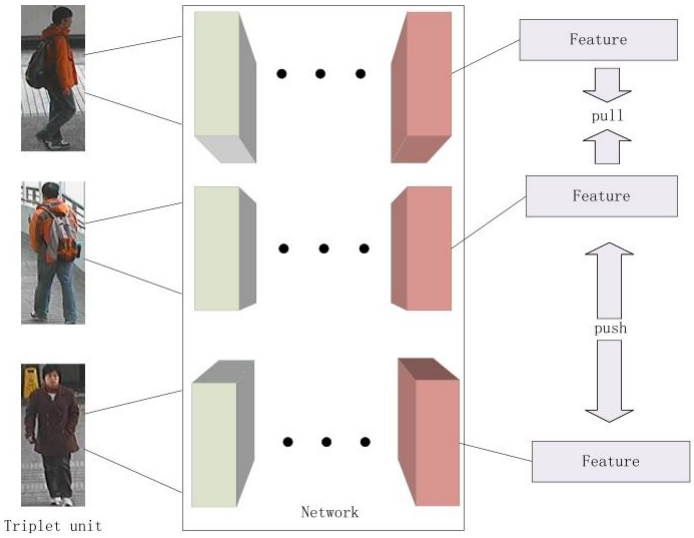


Figure 3.2: Person ReID Triplet Model [3]

4

Supplementary Figures

Figure 4.1: Number of detection of text spotter [4] per runner. The x-axis is runners sorted based on number of detection.

Figure 4.2: Number of detection of text spotter [7] per runner. The x-axis is runners sorted based on number of detection.

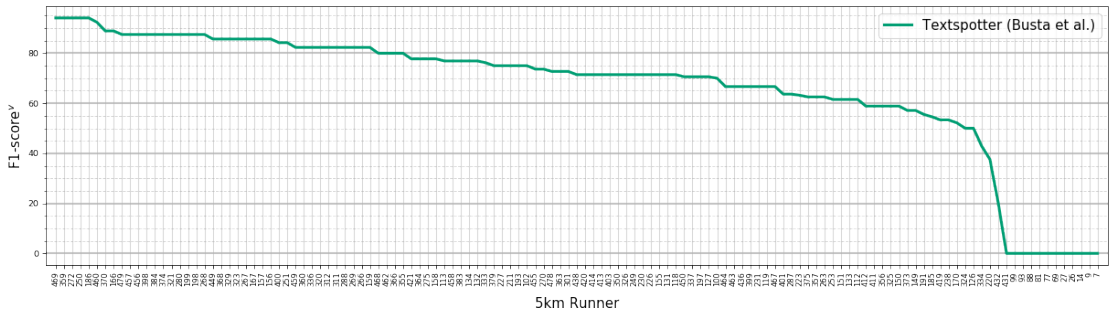


Figure 4.3: F1-score$^{\nu}$ plot of text spotter [4]. The x-axis is runners sorted based on its F1-score$^{\nu}$.

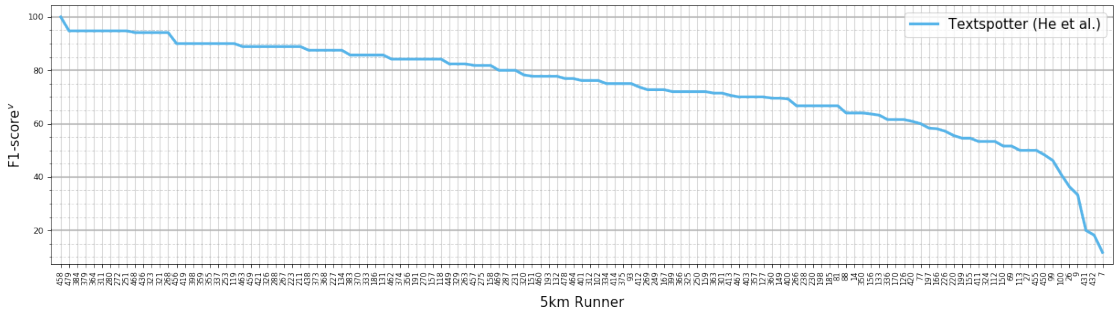


Figure 4.4: F1-score$^{\nu}$ plot of text spotter [7]. The x-axis is runners sorted based on its F1-score$^{\nu}$.

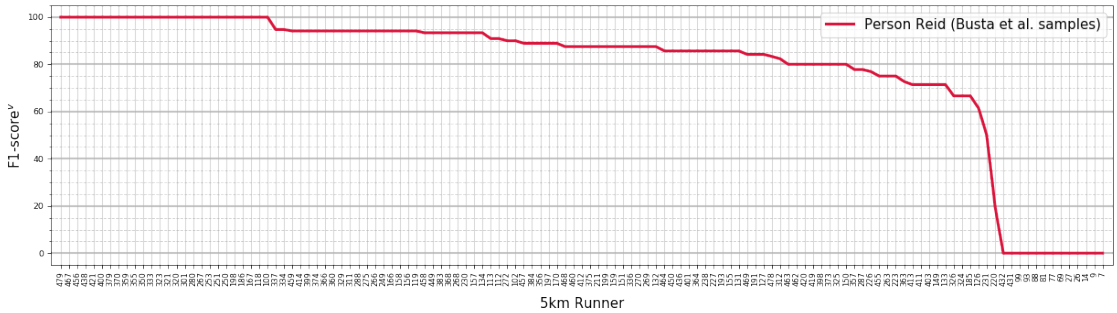


Figure 4.5: F1-score$^{\nu}$ plot of person re-id with training samples from text spotter [4]. The x-axis is runners sorted based on its F1-score$^{\nu}$.

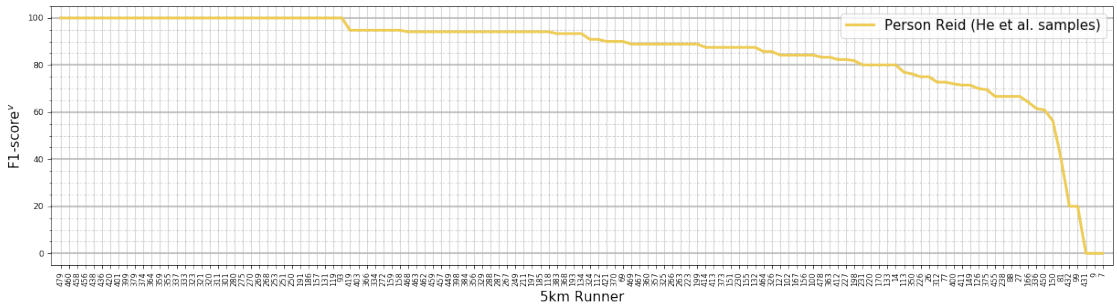


Figure 4.6: F1-score$^{\nu}$ plot of person re-id with training samples from text spotter [7]. The x-axis is runners sorted based on its F1-score$^{\nu}$.

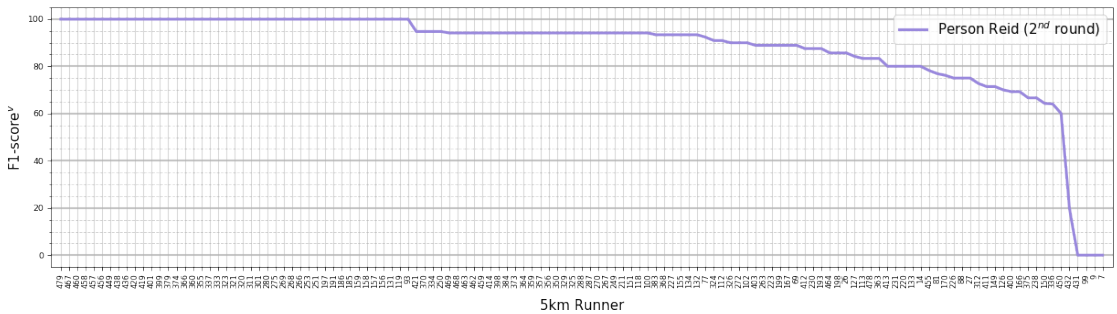


Figure 4.7: F1-score$^{\nu}$ plot of retrained person re-id. The x-axis is runners sorted based on its F1-score$^{\nu}$.

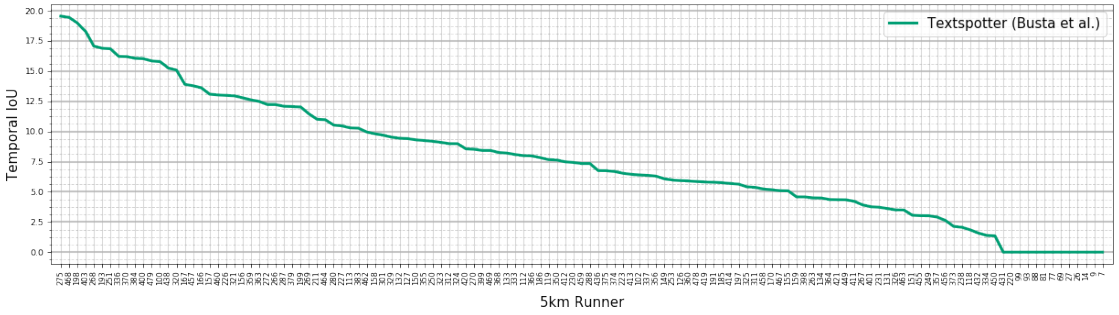


Figure 4.8: Average temporal IoU plot of text spotter[4]. The x-axis is runners sorted based on its average temporal IoU value.

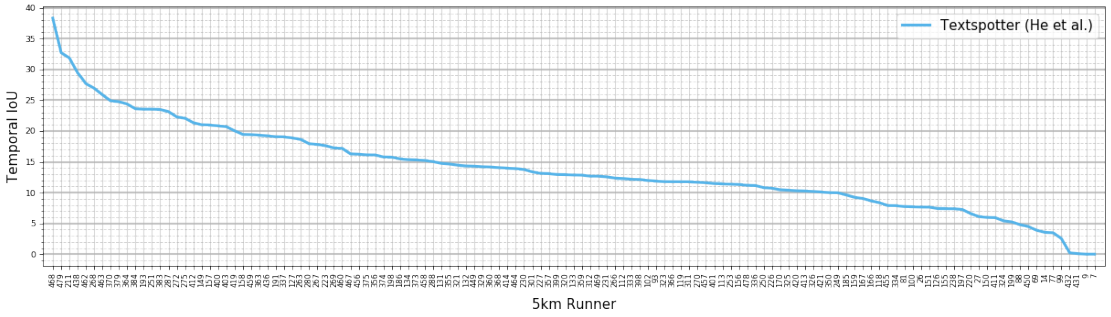


Figure 4.9: Average temporal IoU plot of text spotter[7]. The x-axis is runners sorted based on its average temporal IoU value.

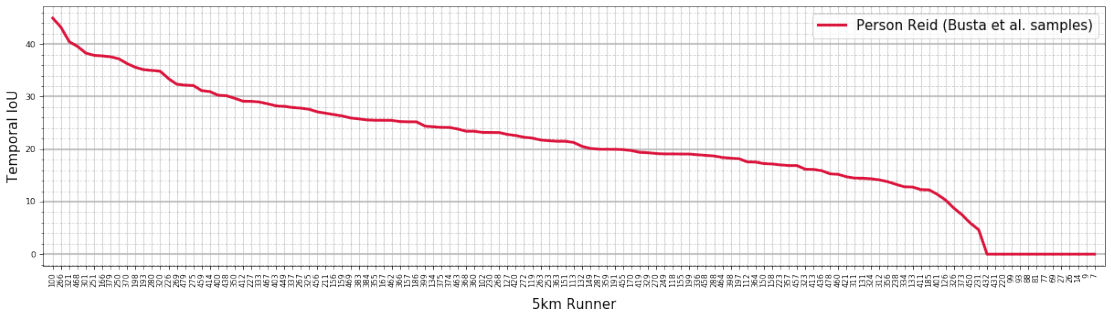


Figure 4.10: Average temporal IoU plot of person re-id with training samples from text spotter [4]. The x-axis is runners sorted based on its average temporal IoU value.

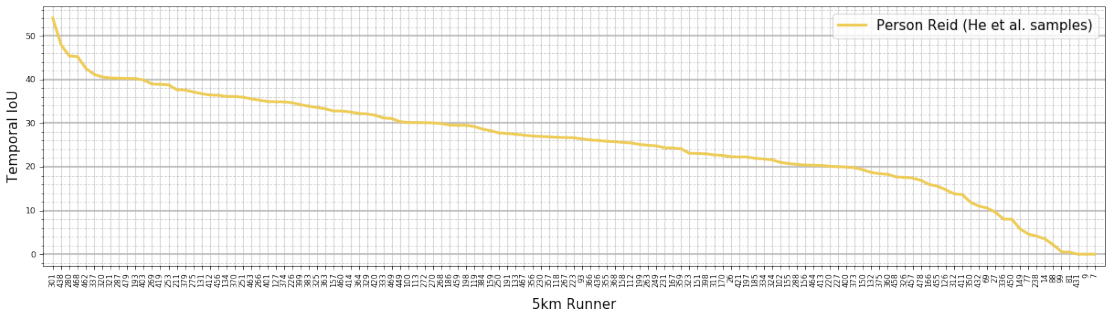


Figure 4.11: Average temporal IoU plot of person re-id with training samples from text spotter [4]. The x-axis is runners sorted based on its average temporal IoU value.

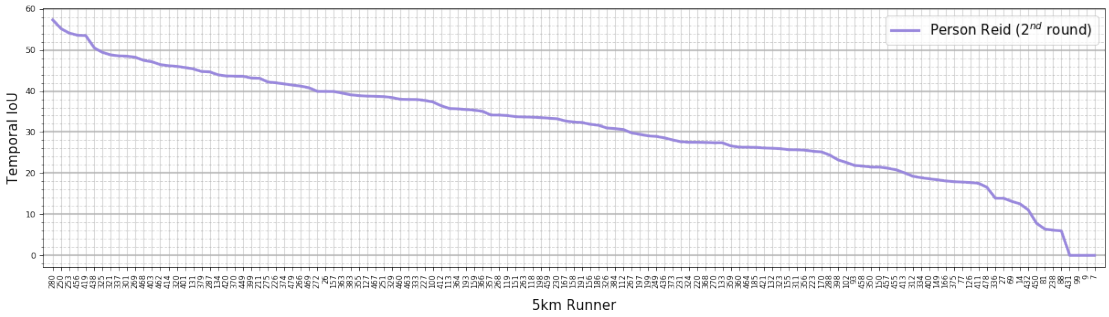


Figure 4.12: Average temporal IoU plot of retrained person re-id. The x-axis is runners sorted based on its average temporal IoU value.

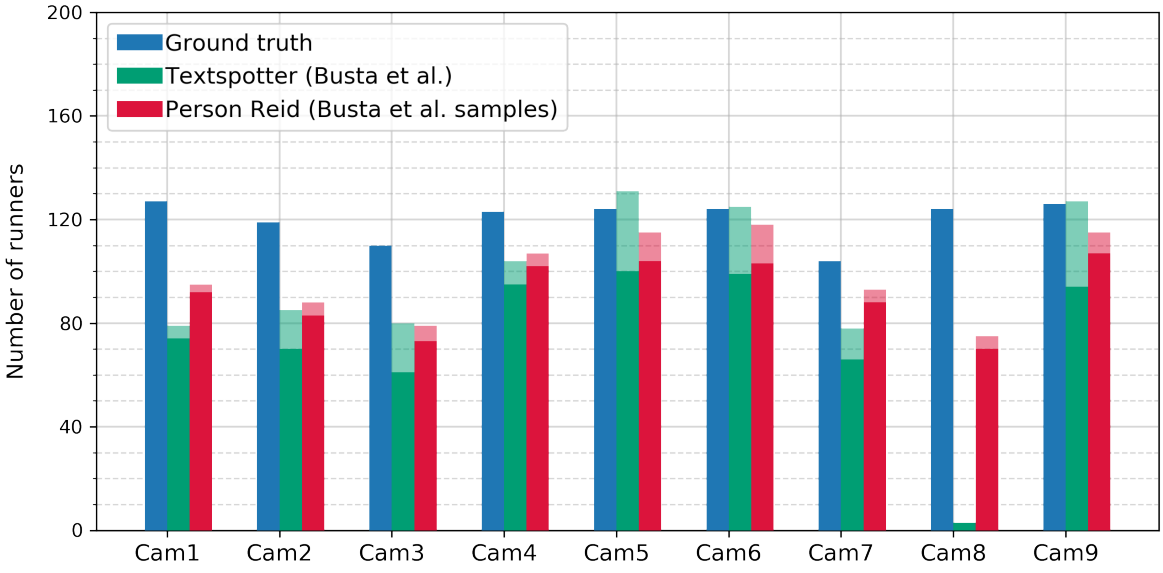


Figure 4.13: The comparison of the number of detected runners per camera between text spotter [4] and person-reid trained with samples from text spotter [4]. The stacked bars with lighter color are false positives.

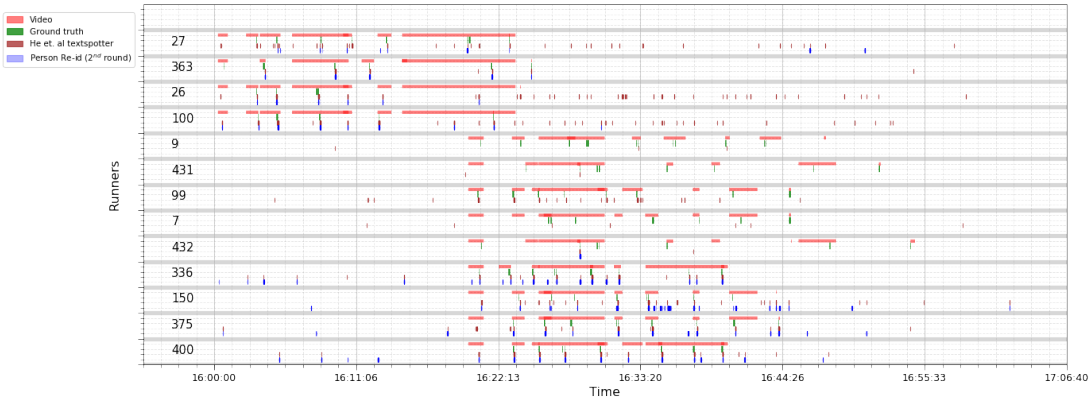


Figure 4.14: 2D Timeline visualization (Part 1) of detection result from text spotter [7] and retrained person re-id. The bib numbers are on the left side of the plot

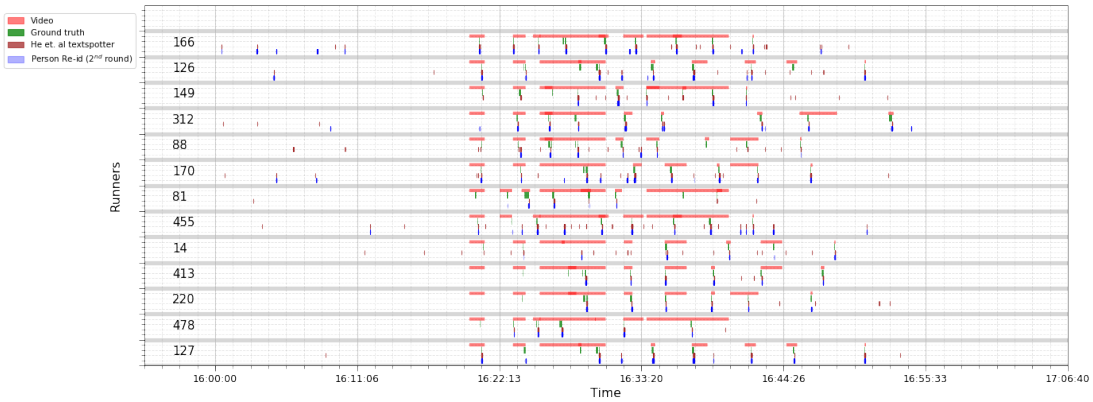


Figure 4.15: 2D Timeline visualization (Part 2) of detection result from text spotter [7] and retrained person re-id. The bib numbers are on the left side of the plot

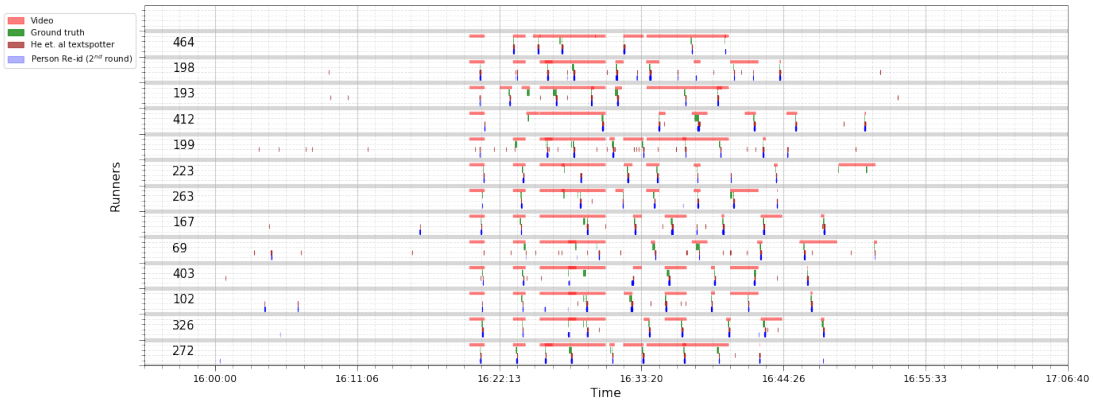


Figure 4.16: 2D Timeline visualization (Part 3) of detection result from text spotter [7] and retrained person re-id. The bib numbers are on the left side of the plot

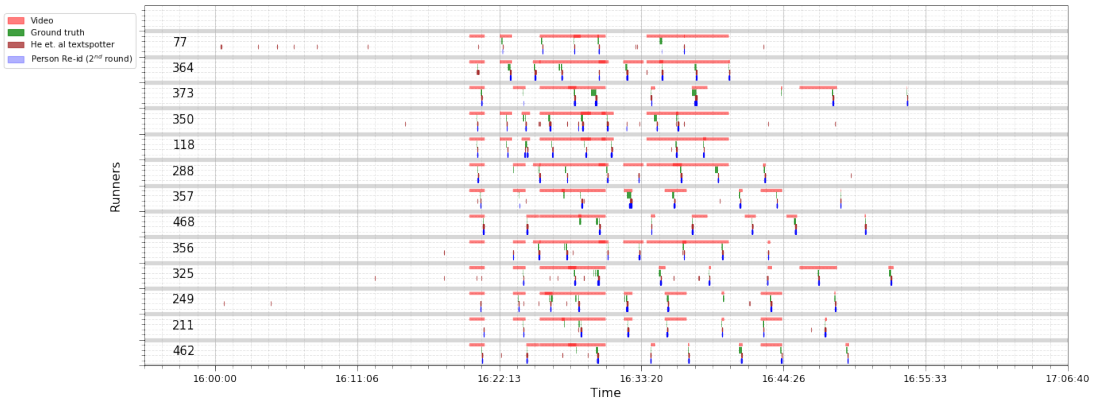


Figure 4.17: 2D Timeline visualization (Part 4) of detection result from text spotter [7] and retrained person re-id. The bib numbers are on the left side of the plot

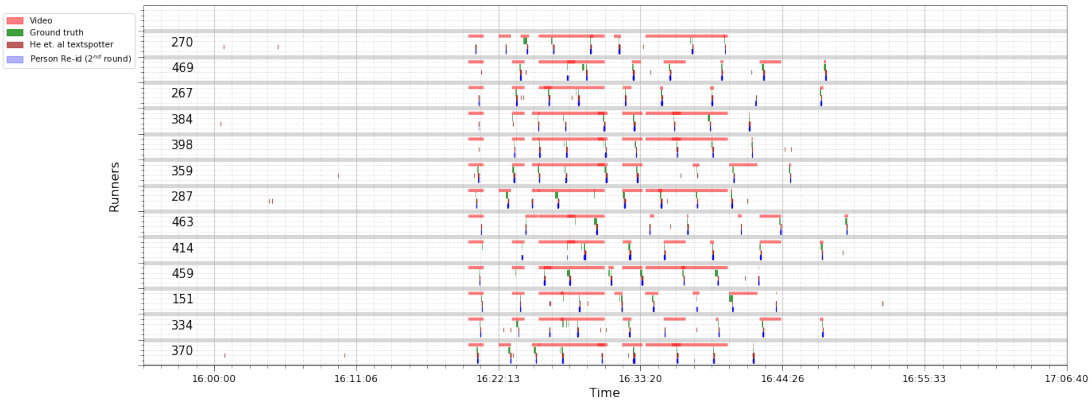


Figure 4.18: 2D Timeline visualization (Part 5) of detection result from text spotter [7] and retrained person re-id. The bib numbers are on the left side of the plot

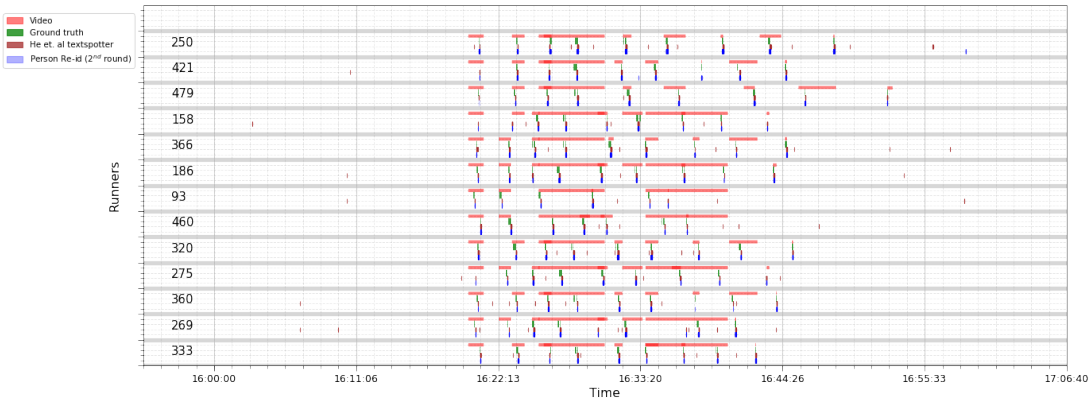


Figure 4.19: 2D Timeline visualization (Part 6) of detection result from text spotter [7] and retrained person re-id. The bib numbers are on the left side of the plot

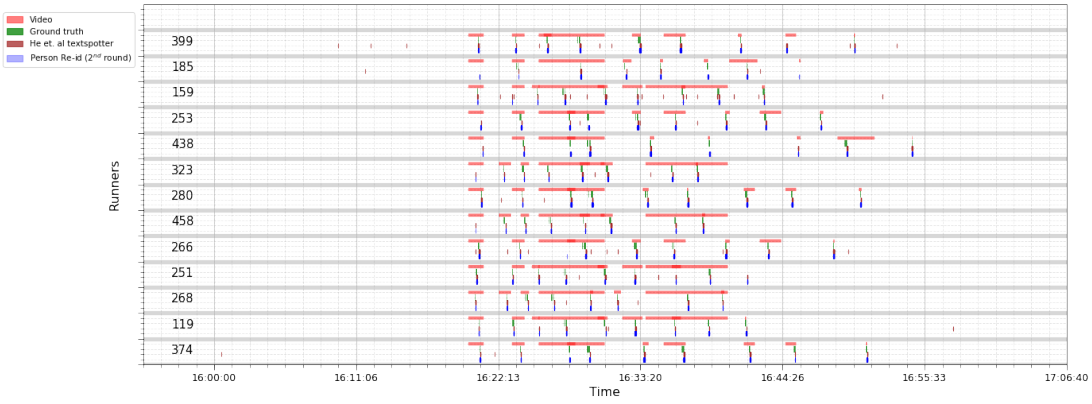


Figure 4.20: 2D Timeline visualization (Part 7) of detection result from text spotter [7] and retrained person re-id. The bib numbers are on the left side of the plot

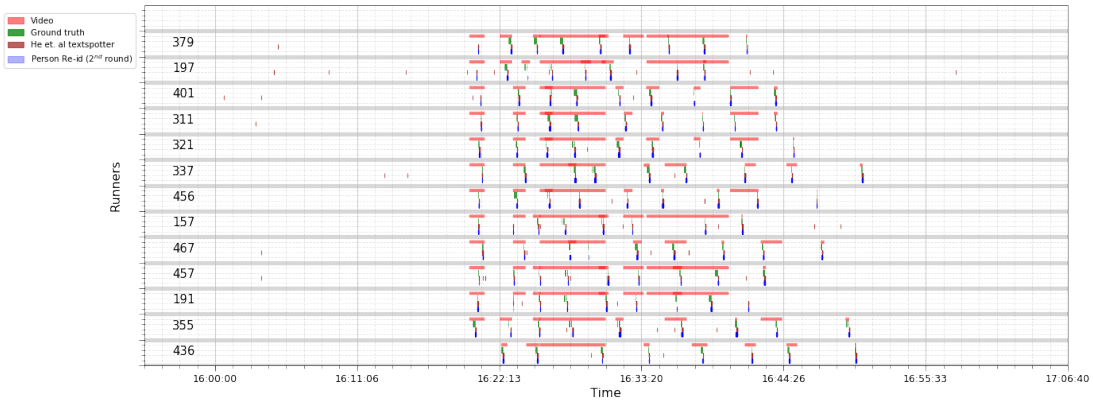


Figure 4.21: 2D Timeline visualization (Part 8) of detection result from text spotter [7] and retrained person re-id. The bib numbers are on the left side of the plot

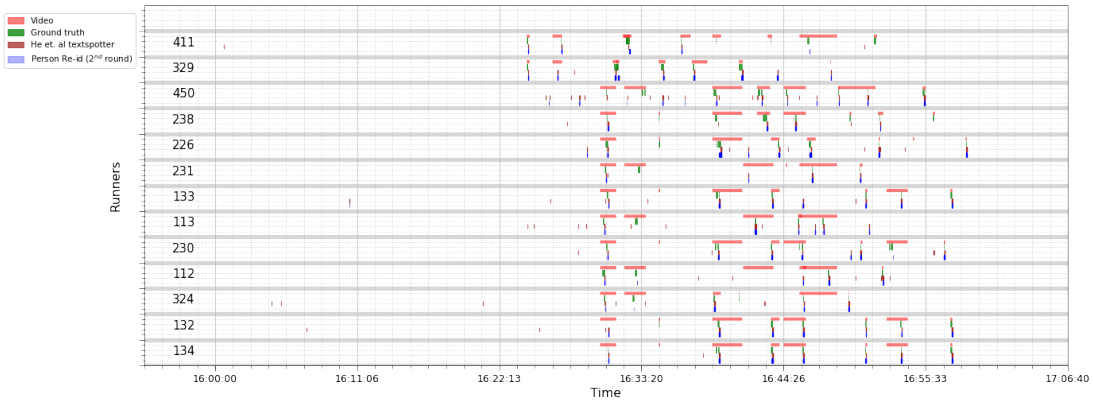


Figure 4.22: 2D Timeline visualization (Part 9) of detection result from text spotter [7] and retrained person re-id. The bib numbers are on the left side of the plot

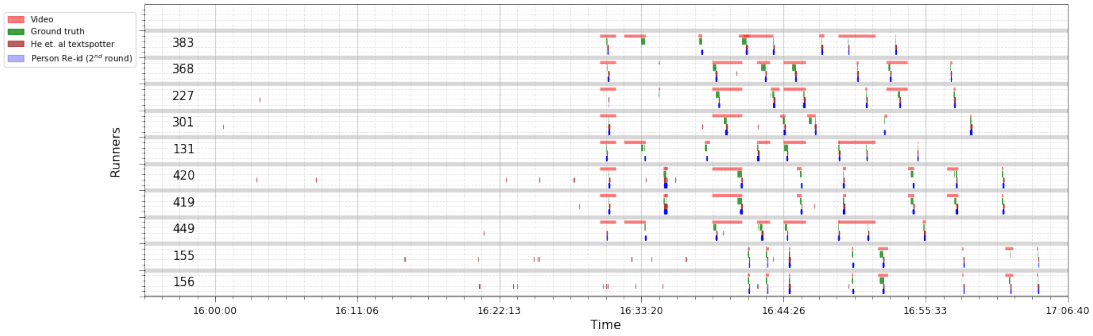


Figure 4.23: 2D Timeline visualization (Part 10) of detection result from text spotter [7] and retrained person re-id. The bib numbers are on the left side of the plot

Bibliography

[1] Introduction to deep learning: What are convolutional neural networks? video, . URL https://www.mathworks.com/videos/.html.

[2] Cs231n convolutional neural networks for visual recognition, . URL https://cs231n.github.io/.

[3] Deep learning-based methods for person re-identification: A comprehensive review. *Neurocomputing*, 337:354 – 371, 2019. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2019.01.079.

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

[5] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009. doi: 10.1109/CVPR.2009.5206848.

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[7] Tong He, Zhi Tian, Weilin Huang, Chunhua Shen, Yu Qiao, and Changming Sun. An end-to-end textspotter with explicit alignment and attention. *CoRR*, abs/1803.03474, 2018.

[8] Alexander Hermans, Lucas Beyer, and Bastian Leibe. In defense of the triplet loss for person re-identification. *CoRR*, abs/1703.07737, 2017.

[9] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[10] Haibo Jin, Xiaobo Wang, Shengcai Liao, and Stan Z. Li. Deep person re-identification with improved embedding. *CoRR*, abs/1705.03332, 2017.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017. ISSN 0001-0782.

[12] Bahram Lavi, Mehdi Fatan Serj, and Ihsan Ullah. Survey on deep learning techniques for person re-identification task. *CoRR*, abs/1807.05284, 2018.

[13] Hao Liu, Jiashi Feng, Meibin Qi, Jianguo Jiang, and Shuicheng Yan. End-to-end comparative attention networks for person re-identification. *CoRR*, abs/1606.04404, 2016.

[14] Aditya Rohilla and Aditya Rohilla. What is deep learning and why you should know about it!, May 2018. URL https://medium.com/mindorks/what-is-deep-learning-and-why-you-should-know-about-it-1dcabac4ab9c.

[15] Sebastian Ruder. An overview of gradient descent optimization algorithms. *CoRR*, abs/1609.04747, 2016.

[16] Sumit Saha and Sumit Saha. A comprehensive guide to convolutional neural networks-the eli5 way, Dec 2018. URL https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[17] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao. A discriminative feature learning approach for deep face recognition. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 499–515, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46478-7.

[18] Liang Zheng, Yi Yang, and Alexander G. Hauptmann. Person re-identification: Past, present and future. *CoRR*, abs/1610.02984, 2016.

[19] Zhedong Zheng, Liang Zheng, and Yi Yang. A discriminatively learned CNN embedding for person re-identification. *CoRR*, abs/1611.05666, 2016.