RT-VeD: Real-Time VoI Detection on Edge Nodes with an Adaptive Model Selection Framework

Shuai Wang*†
Southeast University
shuaiwang@seu.edu.cn

Junke Lu*
Southeast University
junkelu@seu.edu.cn

Baoshen Guo
Southeast University
guobaoshen@seu.edu.cn

Zheng Dong
Wayne State University
dong@wayne.edu

ABSTRACT

Real-time Vehicle-of-Interest (VoI) detection is becoming a core application to smart cities, especially in areas with high accident rates. With the increasing number of surveillance cameras and the advanced developments in edge computing, video tasks prefer to run on edge devices close to cameras due to the constraints of bandwidth, latency, and privacy concerns. However, resource-constrained edge devices are not competent for dynamic traffic loads with resource-intensive video analysis models. To address this challenge, we propose RT-VeD, a real-time VoI detection system based on the limited resources of edge nodes. RT-VeD utilizes multi-granularity computer vision models with different resource-accuracy trade-offs. It schedules vehicle tasks based on a traffic-aware actor-critic framework to maximize the accuracy of VoI detection while ensuring an inference time-bound. To evaluate the proposed RT-VeD, we conduct extensive experiments based on a real-world vehicle dataset. The experiment results demonstrate that our model outperforms other competitive methods.

CCS CONCEPTS

• **Applied computing** → **Transportation**.

KEYWORDS

VoI detection, real time,edge computing, reinforcement learning

ACM Reference Format:
Shuai Wang*†, Junke Lu*, Baoshen Guo, and Zheng Dong. 2022. RT-VeD: Real-Time VoI Detection on Edge Nodes with an Adaptive Model Selection Framework. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '22), August 14–18, 2022, Washington, DC, USA.* ACM, San Frncisco, CA, USA, 9 pages. https://doi.org/10.1145/3534678.3539183

†Shuai Wang is the corresponding author.
*Equally Contributoin.

KDD '22, August 14–18, 2022, Washington, DC, USA.

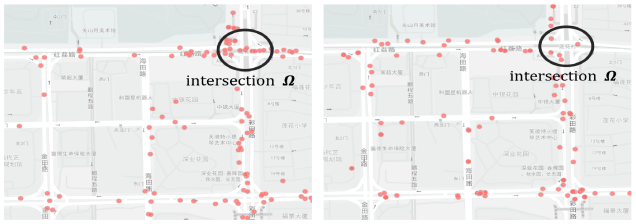
ACM ISBN 978-1-4503-9385-0/22/08…$15.00
https://doi.org/10.1145/3534678.3539183

1 INTRODUCTION

Smart cities are growing worldwide with the rise of artificial intelligence and big data. The proliferation of Internet of Things devices in cities (e.g., surveillance cameras deployed at intersections) has led to an explosion in monitor video data, which contributes to the popularization of road traffic safety applications [19] such as road traffic analysis, anomaly detection, and Vehicle-of-Interest (VoI) detection (i.e., identify target vehicles in the monitor area). Real-time Vehicle-of-Interest (VoI) detection is a core application to public surveillance video analysis, especially in the traffic accident scenes, which utilizes the available video data of cameras at intersections in the road network to acquire real-time passing vehicle information and identify the VoI collaboratively. With advanced computer vision-based technology, such as deep neural networks (DNN), video data are processed to capture the features of vehicles traveling through and search for the candidate targets matching the features of VoI.

Most of existing computer vision based technologies are centralized processing based on a cloud computing platform. These studies focus on fine-grained image processing models running on cloud servers, including vehicle detection [38], identification [3], and tracking model [39]. Researchers propose general models to extract and match vehicle features from images, which focuses on the accuracy of image processing. With the increasing number of cameras, cloud-based video analysis adds up higher latency and consumes a larger amount of bandwidth during data uploading, which is a bottleneck for real-time applications. Considering these constraints, the concept of *edge computing* [36, 37] has been proposed. It brings computing resources close to the data source, which enables fast local computation and provides opportunities for real-time systems.

In this paper, we study how to leverage edge nodes to detect real-time VoIs by implementing a video analysis system on the edge device. Even though edge nodes enable many tasks to be performed quickly with powerful GPU platforms [21], whether real-time tasks can be completed in time is affected by many factors. On the one hand, edge nodes carry a fixed computing budget compared to the cloud server. A typical computation-intensive fine-grained vehicle identification model using deep neural network needs more inference time on the edge due to the limited computing resources. On the other hand, the task load of edge nodes at intersections is dynamic. Figure 1 shows the traffic conditions during morning and evening rush hours in Shenzhen city. At most of the intersections, the passing number of vehicles is small and hence we can



(a) Morning rush hour (b) Evening rush hour

Figure 1: Traffic flow during rush hours in Shenzhen, China

utilize a fine-grained model to process each vehicle and identify the VoI quickly. However, in some cases, when the number of vehicles that appear in the video is too large (as the intersection Ω in Figure 1a), the real-time video processing for all vehicles requires far more computing resources and may not be able to complete in time, leading to a failed retrieval of the VoI. However, in Figure 1b, the volume of traffic at intersection Ω has decreased. Under the dynamic traffic load, whether the task can be completed in real time is a challenge for VoI identification. In view of limited computing resources on edge nodes and dynamic traffic flow, this paper aims to achieve accurate detection of VoI under the completion time bound on resource-constrained edge nodes. To deal with the above issue, we design a smart Real-Time Vehicle-of-Interest Detecting system, RT-VeD, to pinpoint the target vehicles (VoIs) quickly on the edge node at each intersection. Specifically, we have two major components in RT-VeD: (i) *a multi-granularity vehicle identification module* and (ii) *an adaptive real-time vehicle-model matching framework*. When our design executes on the edge nodes, each real-time task is equivalent to a vehicle identification module running on the detected vehicle. The execution time of a real-time task depends on the multi-granularity vehicle identification model selected by the adaptive vehicle-model matching framework. In particular, a vehicle-model matching policy network based on deep reinforcement learning is designed to adaptive select the most suitable model for each task and make the real-time task complete in time. Our contributions are as follows:

- We propose an effective real-time system for detecting target vehicles (VoIs) in an intelligent city transportation system, which enables us to utilize the limited resources of edge nodes to complete the real-time tasks efficiently, especially during heavy traffic hours.
- We consider the inference time of the video processing model and real-time traffic conditions synchronously. A multi-granularity vehicle identification module and an adaptive real-time vehicle-model matching framework are proposed to realize the accurate search of target vehicles (VoIs) in a limited time.
- We have extensively evaluated RT-VeD in real-world vehicle datasets. The experimental result shows that our model outperforms other competitive methods.

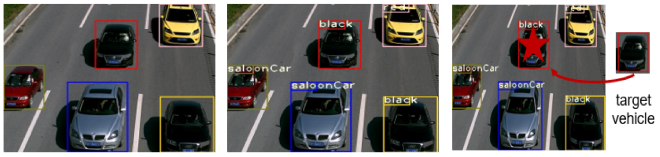
2 PROBLEM STATEMENT

Given the intersections with smart edge devices in the road network of a city, we aim to locate the real-time position of the VoIs by processing the real-time video from the surveillance cameras of related intersections, using the computing resources of edge devices correspondingly. In our setup, we suppose the combination of a surveillance camera and an edge computing platform as an **edge node**, which is deployed at each intersection. A number of video analysis models are deployed on the edge node to process the real-time video streaming captured by the camera.

2.1 Vision-based VoI Detection

VoI detection from surveillance video relies on computer vision-based machine learning algorithms. Especially, deep learning has become increasingly popular in computer vision over the years. When our real-time system works, input is a query request for the target vehicle. Once the input is received, the video analysis algorithm deployed on the edge nodes starts to perform traffic video data analysis in real-time, and then outputs the current possible positions of the target vehicle. More specifically, a target vehicle detection can be roughly divided into the following modules (as Figure 2 shows):

- **Vehicle detection**. For each frame of input video, vehicle detection aims to locate all vehicles in each picture and label the corresponding vehicle bounding box. A typical two-stage vehicle detection network utilizes a Convolutional Neural Network (CNN) based feature extractor to generate feature map of an input video frame and a region proposal network to create candidate vehicle bounding boxes.
- **Vehicle identification**. For each vehicle bounding box, vehicle identification aims to obtain the feature information of the vehicle image, such as color, make or plate number. Traditional vehicle identification usually uses a deep CNN-based feature extractor (e.g. MobileNetV2) to extract fine-grained attribute features of a vehicle image, including global, region and key-point features.
- **Target vehicle matching**. For each target vehicle, vehicle matching aims to find the bounding boxes consistent with the feature of the target vehicle from the real-time video and locate its position, through the feature information obtained from vehicle identification. Query input is a target vehicle with its feature information.



(a) Vehicle detection (b) Vehicle identification (c) Target vehicle matching

Figure 2: Vision-based VoI detection

2.2 Problem Formulation

Considering the inference time of vision-based models on the edge nodes and real-time task processing, the problem is formulated as follows.

Let $I = \{i_1, i_2, \ldots, i_n\}$ be a set of intersections in the area near VoI (e.g., hit-and-run vehicle near the accident area) and $V = \{v_1, v_2, \ldots, v_m\}$ be the set of vehicles in this area. Suppose $v_x \in V$ (which is the VoI) is involved in a accident, we aim to identify the v_x and obtain the intersection i_x where it is from the real-time traffic. For each

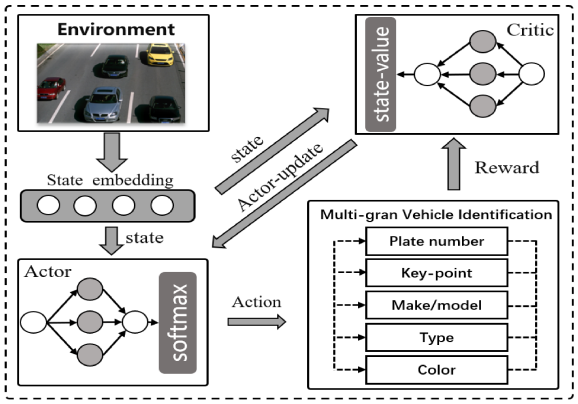


Figure 3: Overall framework

intersection i_k, given the sequence of vehicles at time step t (i.e., $i_k^t = \{v_1^{i_k}, v_2^{i_k}, \ldots, v_n^{i_k}\}$) and the edge device, we need complete vehicle detection and identification module in a time-bound t. Let t^d denote the processing time of vehicle detection on video frame, and t_x^{id} denote the inference time for vehicle identification module in $v_x^{i_k}$. In order to obtain the features of all vehicles in real time, we must make the video analysis time $T = t_d + \sum_{x=1}^{n} t_x^{id}$ less than t.

3 MODEL DESIGN

In this section, we give the overview architecture of our RT-VeD system, then introduce the two main modules of the architecture in detail.

3.1 Overall Architecture

To identify the VoI from all vehicles passing through the intersections in real time, a number of popular video analysis models are deployed on the edge nodes to extract the feature information of all vehicles in each video frame synchronously. To make up for the limited computing capacity of the edge nodes in real-time tasks, we seek to develop an adaptive model to trade-off the vehicle identification performance and the inference time of different models according to the real-time traffic condition. Figure 3 illustrates the overall architecture of the proposed method. There are two major components in RT-VeD: *a multi-granularity vehicle identification module* and *an adaptive real-time vehicle-model matching framework*. The multi-granularity vehicle identification module realizes five-granularity vehicle identification network models to extract five features of vehicles. Each model consumes distinct inference time and outputs vehicle information in different granularity when processing images. Considering the current traffic condition, the adaptive real-time vehicle-model matching framework is a dynamic policy network to match each vehicle bounding box to a model of appropriate granularity for the time constraints, which is realized through an actor-critic reinforcement learning framework[2, 10].

3.2 Multi-granularity Vehicle Identification

We regard the target vehicle identification as a process of feature matching, and divide vehicle features into multiple granularities. Multiple granularities of vehicle features are utilized to match target vehicles in different degrees (e.g., Vehicle color as a coarse-grained feature assists in detecting candidate vehicles with the same color as the VoI. Vehicle plate number as the most fine-grained feature locates the only VoI.). Since the fine-grained vehicle identification algorithm needs an unaffordable amount of time to process video frames (especially during the rush hour), the VoI detecting system can use a coarse-grained algorithm to save time. This paper divides vehicle features into five different granularity as follows:

Table 1: Cost of multi-granularity models

	Inference time	CPU usage
Color	1.0ms	0.3%
Type	30.2ms	15.8%
Make	84.6ms	25.3%
Key-point	180ms	30%
Plate number	514ms	37.9%

- **Color identification**. As a basic classifier, color is an essential feature of a vehicle. Color identification [7] measures the similarity between bounding boxes and VoI by color distribution in spaces. Given Hue Saturation Intensity (HSI) features, we use K-Nearest Neighbors (KNN) [14] to identify the color of each bounding box. After the coarse-grained identification model, a set of vehicles of similar color to the target vehicle can be detected, which narrows down the search scope of the target vehicle.
- **Type identification**. There are many types of vehicles, including bus, truck, sedan and so on. Vehicle type features [8] can be distinguished by differences in appearance and silhouette, by which we lock on to vehicles of the same type as VoI. To extract vehicle type features better, we utilize the B-CNN model [18] to process bounding boxes of vehicles.
- **Make identification**. Make and model of a vehicle are more distinctive [6], and such information extracts vehicle features in a finer granularity level. A typical algorithm seeks to use deep learning approaches to generate discriminative features because of the ability to extract features automatically. For instance,we could adopt the MobileNetv2 network [35] for such purpose.
- **Key-point identification**. The above several granularities of vehicle feature are the common features of a class of vehicles. However, each specific vehicle has its unique key-point feature on some key parts, (e.g., windows [25]) which assists in more accurate match to VoI. This paper utilizes the Faster R-CNN network structure to train a key-point classifier to extract the fine-grained feature of vehicles.
- **Plate number identification**. License plate number is a unique identification for a vehicle, acting as its identity card [27]. In other words, as the most fine-grained classifier, plate number feature confirms the location of VoI accurately. In this study, YoLov3 [16] is used to detect the license plate frame, and a CNN model is used to recognize and classify the specific text information of license plate.

Our Multi-granularity vehicle identification module is built upon five algorithms with different granularity levels. To show the computation cost of different granularity algorithms, we implement these algorithms and measure inference time and cpu usage on one vehicle bounding box of different granularity feature extractors in

Table 1. The test environment is AMD Ryzen 7 3700x (CPU) with 4G memory and NVIDIA GeForce RTX 2080 Ti (GPU).

3.3 Adaptive Vehicle-Model Matching Framework

Take into account the dynamics of the traffic condition and the performance of multi-granularity vehicle feature in a real scenario, we propose an adaptive traffic-aware vehicle-model matching framework via actor-critic reinforcement learning. Actor is a core policy network that outputs actions at each time step according to the state. Critic is another function approximator, which receives the state of environment as input and output the state value to evaluate the action and give feedback to the actor. This framework is utilized to select a model of appropriate granularity for each vehicle bounding box under the complete time bounds for real-time tasks.

3.3.1 Problem Formulation as MDP. We model the vehicle-model matching as a sequential decision problem [1] where we consider matching each vehicle bounding box with an appropriate granularity model at each time step. Our goal is to accomplish real-time tasks while maximizing the accuracy of target vehicle detection. Then we model the problem as a Markov Decision Process (MDP), which is characterized by four major components: $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P})$.

- **State** $\mathcal{S}$: We take the environment captured by the monitoring as input, which includes two parts: the feature of vehicle passing through the intersection, and the computing resources of edge device. The state of an intersection i at time step t is defined as $s_t^i = \{v_1^t, v_2^t, \ldots, v_n^t; e_t\}$, where v_n^t is the real-time state of vehicle n' bounding box including different graunlarity features it has obtained. e_t is the current cpu usage state of the edge node at intersection i.
- **Action** $\mathcal{A}$: An action means the matching of all vehicles and models currently. The action space for each vehicle at time step t is define as $A_t = \{a_t^c, a_t^t, a_t^m, a_t^k, a_t^p\}$, which represents five models available for each current vehicle bounding box. At each time step, the appropriate model should be simultaneously matched for each vehicle at the current intersection. The action space is variable due to the dynamic traffic in the road. So the a_t is represented as a action set: $\{a_{v_1}^t, a_{v_2}^t, \ldots, a_{v_n}^t\}$, where n is the number of vehicles.
- **Reward** $\mathcal{R}$: After the agent taking an action a_t at the state s_t, we will receive the immediate reward $r(s_t,a_t)$ based on system's feedback. The reward $r(s_t,a_t)$ is two-fold: (i) The information entropy generated by the multi-granularity information of vehicle identification and (ii) real-time completion of task processing. The details are explained in Section Reward Function.
- **State transition** $\mathcal{P}$: Given the current state s_t, the information of vehicles obtained will change after an action a_t is taken. The intersection state will also change correspondingly. State transition probability $p(s_{t+1} \mid s_t, a_t)$ defines the state transition from s_t to s_{t+1} after taking action a_t. We assume that the MDP satisfies $p(s_{t+1} \mid s_t, a_t, \ldots, s_1, a_1) = p(s_{t+1} \mid s_t, a_t)$.

Given the historical MDP, i.e., $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P})$, our goal is to find an optimal policy $\pi : \mathcal{S} \rightarrow \mathcal{A}$, which can maximize the cumulative reward from the sequential decision process. We design a **traffic-aware actor critic** framework to conduct vehicle-model matching problem considering instant time constraints and matching reward.

3.3.2 Recurrent Matching Network as Actor. Considering the influence between the actions of different vehicles at the same time step due to the competition of limited resources, we learn strategy with recurrent neural network (RNN) [28]. To obtain an optimal policy network (actor) π, we propose an encoder-decoder framework based on RNN, named recurrent matching network, to extract the dynamic traffic feature and output action for each corresponding vehicle bounding box. Given the input state $s_t = \{v_1^t, v_2^t, \ldots, v_n^t; e_t\}$, the actor learns to infer the target action set $a_t = \{a_{v_1}^t, a_{v_2}^t, \ldots, a_{v_n}^t\}$ for every vehicle bounding box at time step t.

As shown in Figure 4, the actor has two RNN networks as the encoder and decoder respectively. The low-level RNN has two inputs: (i) the sequence of vehicle state $V_t = \{v_1^t, v_2^t, \ldots, v_n^t\}$ as the inputs of each RNN cell; (ii) global state embedding as the initial hidden state h_0, representing the current traffic feature at the intersection, as the following

$$h_0^t = f(s_t) \tag{1}$$

$$h_i^t = f(h_{i-1}^t, v_i^t) \tag{2}$$

These inputs are transformed into a sequence of hidden states $(h_1^t, h_2^t, \ldots, h_n^t)$ based on eq.2 and they are aggregated as a hidden vector C_t in the middle layer. The high-level RNN takes C_t as input and generates the sequence of actions $a_{v_1}^t, \ldots, a_{v_n}^t$ continuously. Generation of $a_{v_i}^t$ is conditioned on all previously generated actions $a_{v_1:v_{i-1}}^t$ and the created hidden vector C_t:

$$P(a_t) = \prod_{i=1}^{n} P(a_{v_i}^t \mid a_{v_1:v_{i-1}}^t, C_t) \tag{3}$$

$$P(a_{v_i}^t \mid a_{v_1:v_{i-1}}^t, C_t) = softmax(g(\mathbf{H}_i^t, C_t)) \tag{4}$$

where $g(\cdot)$ is a one-layer neural network, $\mathbf{H}_i^t$ is the hidden state of the high-level RNN which summarizes $a_{v_1:v_{i-1}}^t$.

3.3.3 Reward Function. After the action a_t completing at time step t, our system receives the feature information of the selected model corresponding to the current vehicles (e.g., the color of v_1 or the type of v_2). In the real-time VoI detection system, the objective is to (i) maximize the accuracy of VoI detection, and (ii) ensure real-time tasks meet deadlines.

To realize the above goal(i), we utilize a novel concept, **information entropy**, to represent the accuracy of VoI detection. In the action space $A_t = \{a_t^c, a_t^t, a_t^m, a_t^k, a_t^p\}$ of each vehicle, the execution of each action represents that the system obtains corresponding granularity information of a vehicle. e.g., if vehicle v_i selects a_t^c at step t, the system gets the color information for that vehicle. Different actions correspond to different levels of accuracy, just as a license plate number can pinpoint a VoI's location, while color information can only identify candidate vehicles of the same color as the VoI. More practically, vehicles have their own traffic patterns. The distribution of different types (or colors) of vehicles in the road network also has its regularity. That is to say, the probability of vehicles with different attributes appearing at the intersection is different. Each of the information we gain on the vehicle assists in matching the target to varying degrees. So the choice of each action

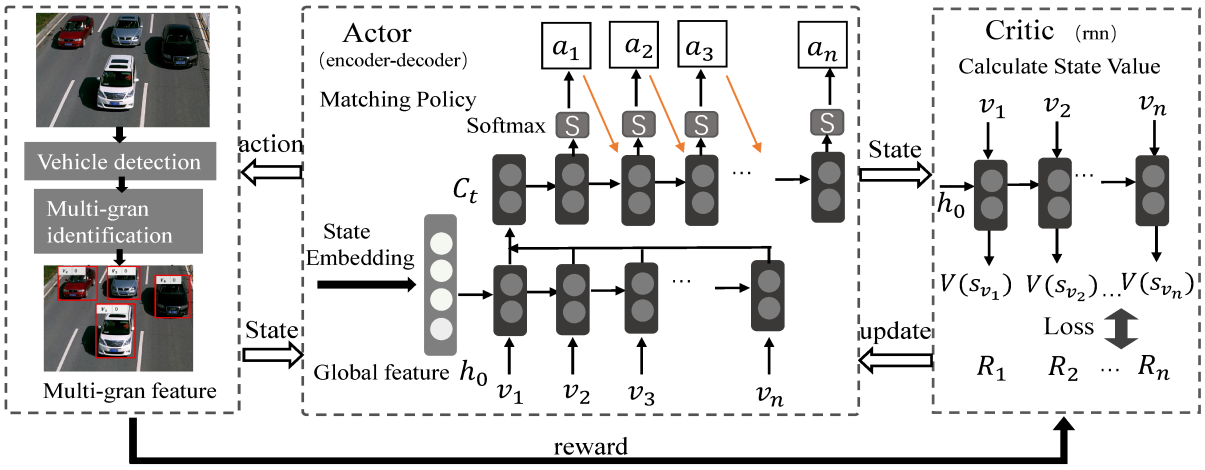


Figure 4: Multi-Granularity Vehicle-Model Matching

is important. Hence, we use information entropy to represent the accuracy of locating VoI with different information obtained.

Information entropy is a quantitative index of the information content of a system and entropy is a measure of the uncertainty of random variables as eq.5. $p(x_i)$ is the probability of x_i occurring. In our scenario, we expressed the probability that the information obtained from each action exists in all vehicles as p. The smaller p is, the higher the accuracy of the information brought by the action is.

We hope to obtain information with high certainty (e.g., plate number) and minimize information entropy. Information gain $g_t = info(s_t) - info(s_t, a_t)$ represents the degree of reduction of information uncertainty under a condition a_t. Therefore, we define the gain from the information obtained for each action a_t as the reward r_t. More specifically, $r_t = [r^t_{v_1}, r^t_{v_2}, \ldots, r^t_{v_n}]$ is a list corresponding to a_t, where $r^t_{v_i} = info(s_t) - info(s_t, a^t_{v_i})$.

$$info(s_t) = -\sum_{i=0}^{n} p(x_i) \log p(x_i) \tag{5}$$

Although we need more fine-grain information, this may cause longer inference time and higer cpu usage, result in the failure to complete the real-time tasks. We set the length of time step as the time-bound. To ensure real-time tasks meet deadlines, we also take into account the inference time of each action. If the inference time of a_t at time step t times out, r_t will be a negative value. Taking into account time constraints, action rewards are defined as:

$$r_t = \begin{cases} \sum_{i=1}^{n} r^t_{v_i}, T \le t \\ -1, T > t \end{cases} \tag{6}$$

3.3.4 Learning from Critic. The critic network is another RNN structure, which receives the same state input as actor and outputs the state value $V(s_t)$. Critic evaluate the action and give actor feeback by state value function $V_{\theta_c}(s_t)$, which is defined as:

$$V_{\theta_c}(s_t) = E[r_t + \gamma r_{t+1} + \cdots + \gamma^n r_{t+n}] \tag{7}$$

where θ_c represents network parameters of critic, γ is the discount factor of reward.

3.3.5 Model Training. We consider the dynamic traffic state and determine vehicle-model matching action at each time step and store the transitions (s_t, a_t, r_t, s_{t+1}) to the experience memory $\mathcal{D}$. Then we sample some transitions for training. The parameters θ_c of critic network are updated by minimizing the following loss function:

$$\mathcal{L}_{\theta_c} = \frac{1}{2}[r_t + \gamma V_{\theta_c}(s_{t+1}) - V_{\theta_c}(s_t)]^2 \tag{8}$$

The training of actor is based on an advantage function, which is used to reduce the high variance of policy networks and stabilize the model:

$$A(s_t, a_t) = r_t + \gamma V_{\theta_c}(s_{t+1}) - V_{\theta_c}(s_t). \tag{9}$$

We utilize actor to choose action and the advantage function evaluate how better the action is by time different method. the gradient of actor is defined as:

$$\nabla_{\theta_a} J(\theta) = \nabla_{\theta_a} \log \pi_{\theta_a}(s_t, a_t) A(s_t, a_t) \tag{10}$$

$$\theta_a \leftarrow \theta_a + \alpha A(s_t, a_t) \nabla_{\theta_a} \log \pi_{\theta_a}(s_t, a_t) \tag{11}$$

where θ_a represents network parameters of actor and $\pi_{\theta_a}(s_t, a_t)$ is the policy probability function. α is the learning rate. The updated process of θ_a is based on the gradient descent. The detailed training process are summarized in Algorithm 1.

4 EVALUATION

To evaluate the efficiency of RT-VeD in realistic scenarios, we conduct extensive experiments based on a real-world datasets.

4.1 Experimental Setup

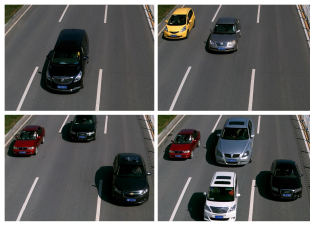
4.1.1 Dataset. To evaluate the performance of our work, we collect a real-world vehicular system dataset, which is provided by the Shenzhen Committee of Transportation (SCT), including one month of GPS traces of more than 30000 vehicles in Shenzhen, China. Figure 5a shows the format of a GPS record. Based on the

Algorithm 1: Traffic-aware actor-critic reinforcement learning for vehicle-model matching

Input: real-time traffic state at the intersection
Output: vehicle-model matching policy for each vehicle
for $i = 0$ *to* ***Iteration number*** **do**
  for $t = 0$ *to episode* **do**
    for *each vehicle* **do**
      Find optimal action: Sample optimal $(s^t_{v_i}, a^t_{v_i})$ pair according to state-action matching probability computed by actor network;
    Execute a_t: and update state s_t;
    Compute $V(s_t)$ **and** $A(s_t, a_t)$, store transitions (s_t, a_t, r_t, s_{t+1}) to replay memory $\mathcal{D}$.
  Updating actor-critic;
  for $j = 0$ *to* ***critic*** **do**
    Sample a batch of (s_t, a_t, r_t, s_{t+1}) from $\mathcal{D}$ Update critic according to eq.8
  for $j = 0$ *to* ***actor*** **do**
    Sample a batch of (s_t, a_t, r_t, s_{t+1}) from $\mathcal{D}$ Update critic according to eq.10

GPS dataset, we obtain the traffic conditions at the city intersections during different periods. In addition, we also utilize a image dataset captured by surveillance cameras deployed at intersections to measure the execution time of the real-time vehicle identification tasks implemented by different models selected by RT-VeD.

Vehicle ID	353211***
Longitude	116.595865
Latitude	23.270327
Time	09:17:39



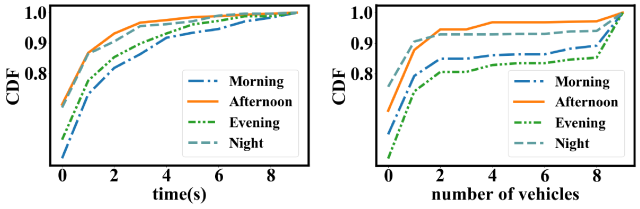
(a) A GPS record **(b) Video frames**

Figure 5: Datasets

4.1.2 Baselines. To evaluate the overall performance of our work, we compare RT-VeD with widely used baselines including two categories: model selection based baselines and non-model selection based baselines.

Model selection baesd baselines:

- **Adadeep[23]** This work proposes an adaptive model parameter selection method, which leverages a DQN based strategy to effectively select a combination of compression techniques that balance user-specified performance goals and resource constraints.
- **SkipRec-RL[5]** This work proposes an adaptive model hidden layer selection framework for deep sequential recommender system. SkipRec-RL is a policy gradient framework to learn to automatically determine which layers should be retained and which layers are allowed to be skipped, so as to achieve user-specific decisions on a per-user basis.
- **Greedy strategy** Regardless of the dynamic task load on edge node, the greedy strategy will select the most fine-grained model first under the time-bound, based on some prior knowledge.
- **RT-VeD** A adaptive model selection strategy to trade off the accuracy and resource according to the real-time traffic conditions based on the traffic-aware actor-critic framework as mentioned above.



(a) Travel time distribution **(b) Number of vehicles**

Figure 6: Travel time and number of vehicles distribution at an intersection

Non-model selection based baselines: On the other hand, in order to verify the effect of the adaptive model selection method, we compare our model with the traditional static fine-grained model approach. Here we utilize the last two fine-grained models mentioned in Section Multi-granularity Vehicle Identification as baselines: keypoint identification ($A_{keypoint}$) and plate number identification (A_{plate}).

4.1.3 Implementation Details. We implement the model and baselines with TensorFlow 1.14 and Python 3.6 environment and train these in an edge server with AMD Ryzen 7 3700x (CPU) and one NVIDIA GeForce RTX 2080 Ti (GPU). We set the time step to one second. The actor has a two-layer RNN and a single-layer RNN structure, and critic has a three-layer RNN as value network. The learning rate α is 0.002.

4.2 Experimental Results

4.2.1 Traffic Measurement. Based on the massive GPS dataset, we measure the dynamic traffic at the city intersections in the monitored areas. Figure 6a shows an example of traveling time taken by vehicles passing an intersection during different periods. We can find vehicles with a longer travel time in rush hour (e.g., morning and evening). In general, about 90 percent of the vehicles across the intersection time is less than 3 seconds, which also motivates us to consider appropriate model deployment to make tasks complete in real-time. According to time and location in the GPS record, we measure the traffic condition at the intersections. Figure 6b shows the distribution of the number of vehicles passing the intersection in one frame. Corresponding to Figure 6a, the larger traffic volume matches the longer traveling time. Furthermore, we can capture the dynamic temporal distribution of traffic in an intersection.

4.2.2 Information Entropy. Information entropy measures the accuracy to locate the target vehicle based on the multi-granularity feature from the model selected. In our experiment, we calculate

the information entropy by using the proportion of vehicle information obtained by each action in the whole vehicle data set in the region. Then we aim to obtain the minimum information entropy to maximize the accuracy of identifying VoI. So we compare the proposed RT-VeD with baselines above and the overall performance of all the methods is shown in Figure 7 to 9.

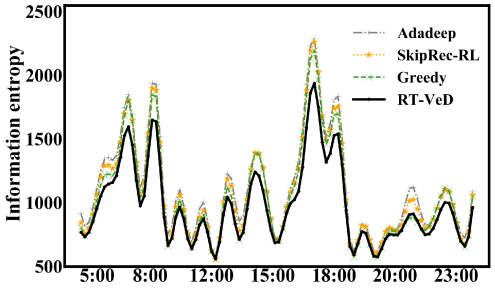
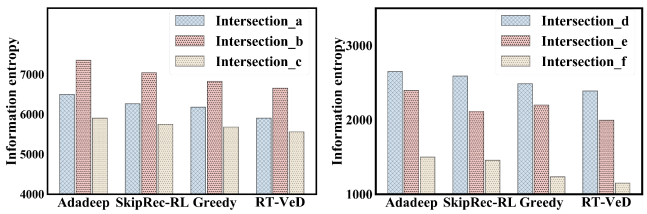


Figure 7: Information entropy performance during a day

Figure 7 shows the advantage of our approach over model selection based baselines in temporal dimension. We evaluate the **cumulative information entropy** every 30 minutes in a day with all the methods and the time variation trend of information entropy as Figure 7 shows. The value of information entropy fluctuates with time and peaks appear in the morning and evening rush hours, which is consistent with the pattern of traffic flow at intersection. Among all the methods, RT-VeD has the lowest information entropy, especially in the peak hours. This shows that our actor-critic reinforcement learning network has learned the traffic patterns at intersections well from the dataset. By combining RNN networks, RT-VeD captures the overall state as well as the fine-grained vehicle state. Adadeep and SkipRec-RL are worse than the others due to suboptimal network structure in our problem. In real scenes, sometimes color is more useful than vehicle make due to the resource constraints, the Greedy strategy fails to capture the dynamic characteristics of the environment. In order to show the impacts of our method in the spatial dimension, we measure the cumulative information entropy of all model selection based baselines in downtown and suburban intersections respectively at the same heavy traffic period in Figure 8. Since the traffic volume in the suburbs is smaller than downtown, the accumulated information entropy obtained is also smaller. As we can see, our method is superior to the other three baselines both in downtown and suburban intersections.

In addition, we compare RT-VeD with two types of baseline under heavy traffic conditions in Figure 9. RT-VeD outperforms other



(a) Downtown intersections **(b) Suburban intersections**

Figure 8: Cumulative information entropy in different areas

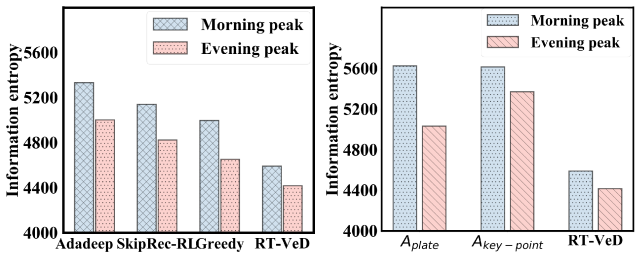


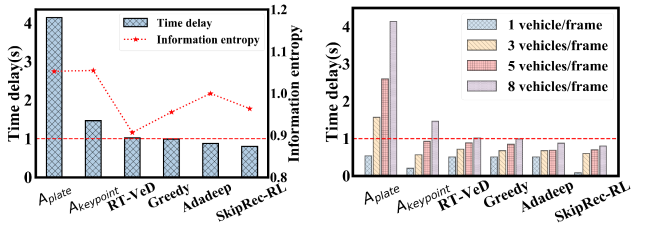
Figure 9: Cumulative information entropy during rush hours

baselines under the morning rush hour and evening rush hour, respectively. Compared with other model selection based baselines, our method has more obvious advantages in peak periods. Due to the time-bound, the traditional static fine-grained vehicle identification model $A_{keypoint}$ and plate number identification A_{plate} running on edge nodes can only accomplish a limited number of tasks in rush hours. RT-VeD makes the most of computing resources of edge nodes through model selection. Figure 9 shows the high efficiency of our method under heavy traffic conditions. The results show that our RT-VeD can detect the target vehicle more accurately in a dynamic traffic environment.

4.2.3 Real-time Performance. To search for target vehicle in real time, the other key performance metric of RT-VeD is the video analysis delay to identify the vehicle in the video frame. In order to study the real-time performance of our method in video analysis, we measure the average vehicle identification delay in one frame, according to the action from model selection framework and the inference time of the multi-granularity model in Table 1.

The bar diagram of Figure 10a shows the average video analysis delay in one frame with different methods during the rush hours. We can see that the RT-VeD and the other three model selection based baselines are able to complete tasks before time-bound while the static fine-grained models are timeout. This is because fine-grained models consume longer network interference time running on edge nodes. The model selection based baselines will give a timeout penalty in training the network so as to reduce the execution time of the method. As shown in the broken line of Figure 10a, we evaluate the proposed approach and other baselines by comparing the relative information entropy with Adadeep methods (which is set as 1.0) during heavy traffic hours. Although the average time delay of Adadeep and SkipRec-RL is shorter than that of RT-VeD, their accuracy is worse because they prefer coarse-grained models, reducing the time cost. To highlight the adaptability of RT-VeD to the dynamic environment, we evaluate the video analytics delay in different traffic conditions. The average delay in one frame corresponding to the different number of vehicle volumes is shown in Figure 10b. When there are few vehicles, all methods can complete the task in a short time. As the number of vehicles increases, $A_{keypoint}$ and A_{plate} are gradually unable to complete the task in real-time, and the others can meet the time-bound. In addition, SkipRec-RL is more sensitive to the number of vehicles. As the number of vehicles increases, SkipRec-RL takes significantly more time. But it prefers models that do not timeout, even though it is not

the most accurate. Among them, RT-VeD can achieve the highest accuracy and meet real-time requirements.



(a) Heavy traffic hour (b) Different traffic conditions

Figure 10: Average delay comparison

Considering the additional running cost of reinforcement learning network, we also measure the inference time of actor network on edge nodes. In particular, the time cost of our vehicle-model matching policy network running on the edge is about 0.001s, which can be ignored.

4.2.4 Summary. In a dynamic traffic environment, RT-VeD with an adaptive model selection policy network realizes the optimal accuracy of VoI detection within a time-bound compared with the other baselines.

5 RELATED WORK

Extensive studies have been conducted to solve target vehicle detection problems. Among existing research, most work focus on capturing targets quickly and accurately at cloud servers. The emergence of edge computing also allows more researchers to participate in related applications. This part investigates some related works of this paper from different three perspectives.

5.1 Vehicle Detection Model in Computer Vision

There have extensive researches on the vehicle detection, identification, tracking and retrieval based on intersection surveillance video data running on cloud servers in the field of computer vision. This kind of work aims to propose a general technical model to extract the precise features from a large number of vehicle images. The present situation of vehicle detection or tracking technology is summarized in [41]. Specifically, [34] and [15] studied the problem of vehicle recognition and classification based on camera in some special environments. On the other hand, vehicle re-identification between multi-camera has been a hot research topic in recent years[24, 39, 40], which becomes challenging task for the intelligent transportation system. [12, 38] utilize the multi-granularity features of vehicles or region feature to identify and track the target vehicle. [31] and [22] make specific study on the problem of multi-target tracking. This paper places target vehicle detection in a new edge computing scenario, creating new challenges.

5.2 Edge-based Real-time Video Analysis

With the popularity of edge computing in recent years, there are a number of researches on real-time task processing models under the limited computing resources of edge nodes. [30] proposes a queuing theory based edge capacity planning solution for real-time computation-intensive applications that takes into account usage of both CPU and GPU. An edge-cloud collaboration method is proposed to minimize the time delay of video processing by reasonably allocating tasks between the cloud and edge[4, 11, 33, 42]. [13] develops an adaptive region of interest (ROI) based image compression scheme to reduce response latency and [32] develops a Face in Video Recognition framework which performs real-time key-frame extraction on IoT edge devices to reduce the data volume. [29] introduces a lightweight convolutional neural network that uses deep separable convolution to speed up computational execution. Most of the work saves running time through model compression, while our work proposes a dynamic model transformation method to make full use of computing resources of edge nodes.

5.3 Model Selection based on Reinforcement Learning

Deep reinforcement learning (DRL) is widely applied in a variety of scenarios to learn actions at different states that maximize reward function[26]. More and more research is devoted to solve decision-making problems using DRL. [17] proposes a scheme for adaptive model selection based on decision making in photonic reservoir computing and [9] develops a Q-learning based dynamic model selection (DMS) framework aiming to choose the best forecasting model from a pool of state-of-the-art machine learning models at each time step. [20] leverage DQN to dynamically select parts of a network to execute according to different input resolution to optimize both accuracy and efficiency of multi-objective optimization problems. [23] leverages a DQN based strategy to effectively select a combination of compression techniques that balance user-specified performance goals and resource constraints. And [5] proposes an adaptive model hidden layer selection framework for deep sequential recommender system, which is a policy gradient framework to learn to skip inactive hidden layers on a per-user basis.

To the best of our knowledge, RT-VeD is the first work to leverage reinforcement learning (RL) for real time VoI detection considering both the accuracy of vehicle matching and computing resource constraints on edge nodes.

6 DISCUSSION

6.1 Lesson learned

Based on the data analysis and evaluation results, we learned the following lessons. (i) Different features of vehicle assists in matching and localization of VoI to varying degrees. (ii) Choosing a suitable model can make full use of the limited resources of edge nodes based on reinforcement learning in our work.

6.2 Future work

Our work is validated on a single edge server, and future research hopes to evaluate distributed multi-edge node collaborative decision-making in a traffic area. At present, edge cameras have not been deployed and applied on a large scale in the city. But with the development of network technology, edge intelligence is bound to become a trend.

7 CONCLUSION

The emergence of edge computing offers opportunities for the implementation of real-time systems on the edge nodes distributed

at the intersections of the road network. In this work, we propose a real-time VoI detection system based on an adaptive model selection framework (RT-VeD) to fully leverage the limited resources of edge nodes, whereby the proper vehicle identification model can be selected on the dynamic traffic condition. We utilize computer vision models with different resource-accuracy trade-offs and propose a multi-granularity model. And then we decompose and schedule vehicle tasks based on the current traffic load with a traffic-aware actor-critic framework to maximize the accuracy of VoI detection in real-time. We evaluate our method based on a real-world vehicle dataset. Experimental results demonstrate that our method outperforms state-of-the-art baselines.

REFERENCES

[1] Oguzhan Alagoz, Heather Hsu, Andrew J Schaefer, and Mark S Roberts. 2010. Markov decision processes: a tool for sequential decision making under uncertainty. *Medical Decision Making* 30, 4 (2010), 474–483.

[2] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. 2017. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine* 34, 6 (2017), 26–38.

[3] Jingye Cai, Jianhua Deng, Muhammad Umar Aftab, Muhammad Saddam Khokhar, Rajesh Kumar, et al. 2019. Efficient and deep vehicle re-identification using multi-level feature extraction. *Applied Sciences* 9, 7 (2019), 1291.

[4] Jianguo Chen, Kenli Li, Qingying Deng, Keqin Li, and S Yu Philip. 2019. Distributed deep learning model for intelligent video surveillance systems with edge computing. *IEEE Transactions on Industrial Informatics* (2019).

[5] Lei Chen, Fajie Yuan, Jiaxi Yang, Xiang Ao, Chengming Li, and Min Yang. 2021. A User-Adaptive Layer Selection Framework for Very Deep Sequential Recommender Models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 3984–3991.

[6] Li-Chih Chen, Jun-Wei Hsieh, Yilin Yan, and Duan-Yu Chen. 2015. Vehicle make and model recognition using sparse representation and symmetrical SURFs. *Pattern Recognition* 48, 6 (2015), 1979–1998.

[7] Pan Chen, Xiang Bai, and Wenyu Liu. 2014. Vehicle color recognition on urban road by feature context. *IEEE Transactions on Intelligent Transportation Systems* 15, 5 (2014), 2340–2346.

[8] Zhen Dong, Yuwei Wu, Mingtao Pei, and Yunde Jia. 2015. Vehicle type classification using a semisupervised convolutional neural network. *IEEE transactions on intelligent transportation systems* 16, 4 (2015), 2247–2256.

[9] Cong Feng and Jie Zhang. 2019. Reinforcement learning based dynamic model selection for short-term load forecasting. In *2019 IEEE Power & Energy Society Innovative Smart Grid Technologies Conference (ISGT)*. IEEE, 1–5.

[10] Ivo Grondman, Lucian Busoniu, Gabriel AD Lopes, and Robert Babuska. 2012. A survey of actor-critic reinforcement learning: Standard and natural policy gradients. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 42, 6 (2012), 1291–1307.

[11] Philipp M Grulich and Faisal Nawab. 2018. Collaborative edge and cloud neural networks for real-time video processing. *Proceedings of the VLDB Endowment* 11, 12 (2018), 2046–2049.

[12] Haiyun Guo, Chaoyang Zhao, Zhiwei Liu, Jinqiao Wang, and Hanqing Lu. 2018. Learning coarse-to-fine structured feature embedding for vehicle re-identification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[13] Yundi Guo, Beiji Zou, Ju Ren, Qingqing Liu, Deyu Zhang, and Yaoxue Zhang. 2019. Distributed and efficient object detection via interactions among devices, edge, and cloud. *IEEE Transactions on Multimedia* 21, 11 (2019), 2903–2915.

[14] DS Guru, YH Sharath, and S Manjunath. 2010. Texture features and KNN in classification of flower images. *IJCA, Special Issue on RTIPPR (1)* (2010), 21–29.

[15] Jun-Wei Hsieh, Shih-Hao Yu, Yung-Sheng Chen, and Wen-Fong Hu. 2006. Automatic traffic surveillance system for vehicle tracking and classification. *IEEE Transactions on Intelligent Transportation Systems* 7, 2 (2006), 175–187.

[16] Yi-Qi Huang, Jia-Chun Zheng, Shi-Dan Sun, Cheng-Fu Yang, and Jing Liu. 2020. Optimized YOLOv3 algorithm and its application in traffic flow detections. *Applied Sciences* 10, 9 (2020), 3079.

[17] Kazutaka Kanno, Makoto Naruse, and Atsushi Uchida. 2020. Adaptive model selection in photonic reservoir computing by reinforcement learning. *Scientific reports* 10, 1 (2020), 1–12.

[18] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. 2015. Bilinear cnn models for fine-grained visual recognition. In *Proceedings of the IEEE international conference on computer vision*. 1449–1457.

[19] Honghai Liu, Shengyong Chen, and Naoyuki Kubota. 2013. Intelligent video systems and analytics: A survey. *IEEE Transactions on Industrial Informatics* 9, 3 (2013), 1222–1233.

[20] Lanlan Liu and Jia Deng. 2018. Dynamic deep neural networks: Optimizing accuracy-efficiency trade-offs by selective execution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 32.

[21] Liangkai Liu, Yongtao Yao, Ruijun Wang, Baofu Wu, and Weisong Shi. 2020. Equinox: A road-side edge computing experimental platform for cavs. In *2020 International Conference on Connected and Autonomous Driving*. IEEE, 41–42.

[22] Qiankun Liu, Bin Liu, Yue Wu, Weihai Li, and Nenghai Yu. 2019. Real-time online multi-object tracking in compressed domain. *IEEE Access* 7 (2019), 76489–76499.

[23] Sicong Liu, Yingyan Lin, Zimu Zhou, Kaiming Nan, Hui Liu, and Junzhao Du. 2018. On-demand deep model compression for mobile devices: A usage-driven model selection framework. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*. 389–400.

[24] Xinchen Liu, Wu Liu, Tao Mei, and Huadong Ma. 2017. Provid: Progressive and multimodal vehicle reidentification for large-scale urban surveillance. *IEEE Transactions on Multimedia* 20, 3 (2017), 645–658.

[25] Xiaobin Liu, Shiliang Zhang, Qingming Huang, and Wen Gao. 2018. Ram: a region-aware deep model for vehicle re-identification. In *2018 IEEE International Conference on Multimedia and Expo (ICME)*. IEEE, 1–6.

[26] Nguyen Cong Luong, Dinh Thai Hoang, Shimin Gong, Dusit Niyato, Ping Wang, Ying-Chang Liang, and Dong In Kim. 2019. Applications of deep reinforcement learning in communications and networking: A survey. *IEEE Communications Surveys & Tutorials* 21, 4 (2019), 3133–3174.

[27] Khalid W Maglad. 2012. A vehicle license plate detection and recognition system. *Journal of Computer Science* 8, 3 (2012), 310.

[28] Larry R Medsker and LC Jain. 2001. Recurrent neural networks. *Design and Applications* 5 (2001), 64–67.

[29] Seyed Yahya Nikouei, Yu Chen, Sejun Song, Baek-Young Choi, and Timothy R Faughnan. 2019. Toward intelligent surveillance as an edge network service (isense) using lightweight detection and tracking algorithms. *IEEE Transactions on Services Computing* (2019).

[30] Marius Noreikis, Yu Xiao, and Yuming Jiang. 2019. Edge capacity planning for real time compute-intensive applications. In *2019 IEEE International Conference on Fog Computing (ICFC)*. IEEE, 175–184.

[31] Fabio Previtali, Domenico D Bloisi, and Luca Iocchi. 2017. A distributed approach for real-time multi-camera multiple object tracking. *Machine Vision and Applications* 28, 3-4 (2017), 421–430.

[32] Xuan Qi, Chen Liu, and Stephanie Schuckers. 2018. IoT edge device based key frame extraction for face in video recognition. In *2018 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*. IEEE, 641–644.

[33] Jinke Ren, Guanding Yu, Yinghui He, and Geoffrey Ye Li. 2019. Collaborative cloud and edge computing for latency minimization. *IEEE Transactions on Vehicular Technology* 68, 5 (2019), 5031–5044.

[34] Reyes Rios-Cabrera, Tinne Tuytelaars, and Luc Van Gool. 2012. Efficient multi-camera vehicle detection, tracking, and identification in a tunnel surveillance application. *Computer Vision and Image Understanding* 116, 6 (2012), 742–753.

[35] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*.

[36] Mahadev Satyanarayanan. 2017. The emergence of edge computing. *Computer* 50, 1 (2017), 30–39.

[37] Weisong Shi, Jie Cao, Quan Zhang, Youhuizi Li, and Lanyu Xu. 2016. Edge computing: Vision and challenges. *IEEE internet of things journal* 3, 5 (2016), 637–646.

[38] Sayanan Sivaraman and Mohan M Trivedi. 2012. Real-time vehicle detection using parts at intersections. In *2012 15th international ieee conference on intelligent transportation systems*. IEEE, 1519–1524.

[39] Jakub Španhel, Vojtech Bartl, Roman Juránek, and Adam Herout. 2019. Vehicle re-identification and multi-camera tracking in challenging city-scale environment. In *Proc. CVPR Workshops*, Vol. 2. 1.

[40] Xiao Tan, Zhigang Wang, Minyue Jiang, Xipeng Yang, Jian Wang, Yuan Gao, Xiangbo Su, Xiaoqing Ye, Yuchen Yuan, Dongliang He, et al. 2019. Multi-camera vehicle tracking and re-identification based on visual and spatial-temporal features.. In *CVPR Workshops*. 275–284.

[41] Anshul Vishwakarma and Amit Khare. 2008. Vehicle detection and tracking for traffic surveillance applications: A review paper.

[42] Baofu Wu, Yuankai He, Zheng Dong, Jian Wan Wan, Jilin Zheng, and Weisong Shi. 2022. To Turn or Not To Turn, SafeCross is the Answer. In *2022 IEEE 42th International Conference on Distributed Computing Systems (ICDCS)*. IEEE.