Vehicle Re-Identification and Multi-Camera Tracking in Challenging City-Scale Environment

Jakub Špaňhel Vojtěch Bartl Roman Juránek Adam Herout
Graph@FIT, Brno University of Technology, Czech Republic
{ispanhel,ibartl,ijuranek,herout}@fit.vutbr.cz
https://medusa.fit.vutbr.cz/traffic/

Abstract

*In our submission to the NVIDIA AI City Challenge, we address vehicle re-identification and vehicle multi-camera tracking. Our approach to vehicle re-identification is based on the extraction of visual features and aggregation of these features in the temporal domain to obtain a single feature descriptor for the whole observed track. For multi-camera tracking, we proposed a method for matching vehicles by the position of trajectory points in real-world space (linear coordinate system). Furthermore, we use CNN for the vehicle re-identification task to filter out false matches generated by proposed **positional matching** method for better results.*

1. Introduction

In this submission, we address the tasks of vehicle multi-camera tracking and re-identification of the NVIDIA AI City Challenge 2019 (i.e. *Track1* and *Track2*).

Our approach to visual vehicle re-identification is based on extraction of feature vectors using a convolutional neural network and aggregation of extracted features vectors from observed vehicles in temporal domain. We use standard CNNs [7, 27, 9] trained for the identification task and we employ an LFTD network [25] for feature aggregation.

For the multi-camera tracking part, we propose a method for matching points from vehicle trajectories in real-world linear coordinate system space. This approach is based on projection of 2D image points into the real-world linear space [30] and matching of vehicles in this linear space with respect to time and space constraints. Furthermore, this approach can be also combined with extraction of feature vectors for all observed and pre-matched tracks.

To put our approach to a larger context, we include a brief overview of the state of the art in vehicle re-identification. After that, we describe the used methods for both vehicle re-id and multi-camera tracking in detail.

1.1. Vehicle Re-Identification

Formerly, the methods for vehicle re-identification were based on automatic license plate recognition [5, 11, 31], using hand-crafted visual features (PCA-SIFT, HOG descriptors, color histograms, *etc.*) extracted from vehicle images [1, 6, 38] or just information about date, time, color, speed and vehicles' dimensions [6].

Recently, *deep* features learned by CNNs [16, 20, 29, 33, 40] are being used for this task. Liu *et al.* [17] combine the hand-crafted and deep features. Improvements were also made by exploiting *spatio-temporal* [17, 29] or *visual-spatio-temporal* [20] properties. Some of them benefit from Siamese CNNs for license plate verification [17] or vehicle image similarities [20]. Moreover, introduction of triplet loss [40, 14] or Coupled Cluster Loss (CCL) [16] led to accuracy improvements and faster convergence. Recently, Yan *et al.* [33] propose to use Generalized Pairwise Ranking or Multi-Grain based List Ranking for retrieval of similar vehicles, which performs even better than CCL.

Few person re-identification papers also proposed to use the triplet loss [4, 8] or quadruplet loss [3] instead of training the network in a Siamese setting. There were also attempts to learn a metric for the re-identification like KISSME [13], XQDA [15], You *et al.* [37] learn Mahalanobis distance on LBP and HOG3D features, and finally Shi *et al.* [21] learn Mahalanobis distance in an end-to-end manner.

On the other hand, a group of methods exists which propose to use feature pooling (aggregation) in temporal domain for re-identification task. Such pooling is usually used in the context of person re-identification (with the exception of Yang *et al.* [35] who used it for video face recognition). The methods are often trained by using a Siamese network [18, 39, 34, 2, 32, 35] with contrastive loss and optionally the identification loss as well. Similarly, in our paper [25], we propose a method for feature aggregation in temporal domain of multiple observations of a vehicle in one track.

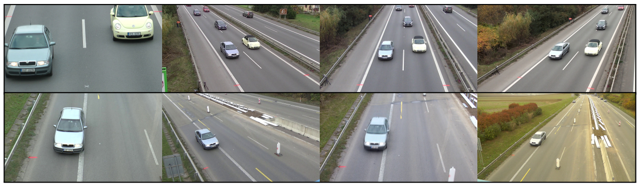

Figure 1. Video screenshots from one recording session with a vehicle with the same identity. Image source: [25].

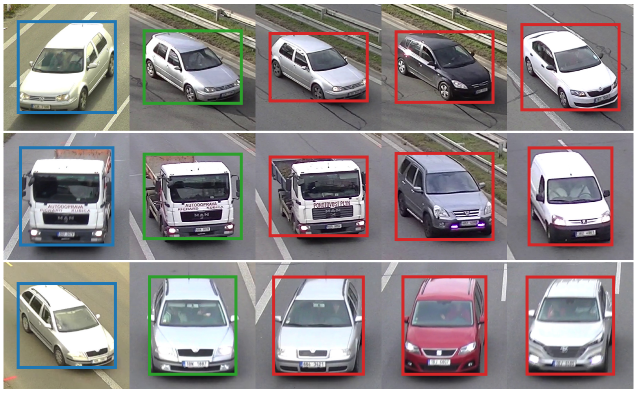

Figure 2. Examples of queries, positive, and negative samples. The negatives are sorted by difficulty from left to right (hard to easy) based on distances obtained from our re-identification feature vectors. It should be noted that the hardest negative sample has usually subtle differences (*e.g*. missing a small spoiler in the first row). Image source: [25].

1.2. Vehicle Re-Identification Datasets

Comprehensive datasets of vehicles for fine-grained vehicle recognition are available [12, 36, 23] for more than 5 years now. However, when it comes to vehicle re-identification the available datasets are limited in some ways. Liu *et al*. [17] constructed a rather small VeRi-776 dataset containing 50,000 images of 776 vehicles. Liu *et al*. [16] collected VehicleID dataset containing 26,267 vehicles in 220k images taken from a frontal/rear viewpoint above road. Recently, Yan *et al*. [33] published two datasets VD1 and VD2 for vehicle re-identification and fine-grained classification with over 220k of vehicles in total, with make, model, and year annotation. However, both datasets are limited to frontal viewpoints only.

Recently, we collected dataset **CarsReId74k** [25], which contains ≈74k of vehicle tracks from various viewpoints with precise ground truth identity acquired from a zoomed-in camera by license plate recognition.

2. Used Approach

In our submission to the NVIDIA AI City Challenge 2019, we focused on vehicle re-identification (Track 2) and vehicle multi-camera tracking (Track 1). In the following text, we describe our approach to both of these tasks.

2.1. Training Data for Vehicle Re-Identification

Both tasks contain vehicles observed from various viewpoints. It is necessary to acquire a similar dataset for pre-training of the identification and also re-identification networks. We used our dataset CarsReId74k [25] which contains 17,681 unique vehicles, 73,976 observed tracks, and 292,226 positive pairs. For examples of positive and negative pairs, see Figure 2.

The dataset was collected using 8 cameras recording at the same time. Four cameras always observed the same direction of traffic at one location from different viewpoints (left, center, right), and one camera was zoomed in and it was used for license plate detection and recognition by our recent method [24]. The videos at one location were approximately synchronized and the recognized license plates were assigned to the detected vehicles from other cameras, producing the identities for all the vehicles. See Figure 1 for examples of videos from one recording session.

2.2. Vehicle Re-Identification

Following the methodology from our previous paper [25], we first **fine-tuned the CNN** on vehicle *identification* task. We used ResNet-50 [7] and InceptionResNetV2 [26] with 2D detection/cropped images only and the input image size was 331 × 331. The fine-tuning was done with Adam [10] optimizer, learning rate 1e-4 and cross-entropy loss. We were not able to use our previously proposed modification using "unpacked" version of vehicle images [23] which is based on construction of 3D bounding boxes as the input of the CNN due to limitations of viewpoints and already cropped images in *Track2*.

On the identification features we trained **LFTD network** [25] to aggregate the features in temporal domain as there are multiple observations for the vehicle as they pass in front of the cameras. The LFTD network contains one fully connected layer with 1,024 output features and $\tanh$ non-linearity. Furthermore, the network contains feature weighting mechanism which weights different elements of the feature vectors by different weights. The network is trained as a Siamese network.

It is possible to use a different distance function during LFTD network training. We used **Weighted Euclidean distance** which is expressed as

$$d_{\mathrm{WE}}(\mathbf{u},\mathbf{v}) = \sqrt{\sum_{i=1}^{D} w_i (u_i - v_i)^2}, \qquad (1)$$

where $\mathbf{u}$, $\mathbf{v}$ are feature vectors and $\mathbf{w} = [w_1, w_2, \ldots, w_D]$ are learned weights.

We evaluated different variants of backbone networks together with the influence of using *image modifiers* [22, 23] and pre-training the networks on different datasets. Complete results of our experiments are depicted in Table 3.

2.2.1 Vehicle Re-Id Design Changes

Results of our submissions from evaluation server showed unbalanced values compared to our evaluation (see Table 3). This led to designed changes in our methodology proposed above. Inspired by previous works [14, 28], we tried to replace our feature extractor by a much smaller CNN MobileNet [9] with feature vector dimensionality reduced to 128 dimensions. The second change was in replacing cross-entropy loss with triplet loss combined with **semi-hard batch sampling** [19]. The rest of our design remained the same. We tried multiple variants of *image modifiers* and pooling methods. The results can be found in Section 3.1.

2.3. Multi-Camera Tracking

Unlike the vehicle re-identification task, the multi-camera tracking task does not have to be solved by visual comparison of detected vehicles only. The problem can be solved even using **positional matching** with knowledge of GPS coordinates of cameras, known distances and time synchronization between them, and their calibrations. In this case, matching is based on projection of the 2D point of vehicle trajectory from the image space into the world coordinate space (linear system in our case). These projections from multiple cameras can be matched with each other for every time step in order to obtain matching between tracks across multiple cameras.

It should be noted that the approach described below assumes that for each camera within the session, an overlap exists in the view area of the camera with at least one other camera in the same session. This condition is satisfied for almost all cameras in test sessions, as can be seen in Figure 7. In other case, vehicles from camera without any overlap cannot be matched with the rest of the cameras and the matching procedure had to be modified. However, with knowledge of time synchronization and distances between the cameras, this modification is straightforward.

2.3.1 Vehicle Trajectory Estimation

Positional matching counts on estimation of trajectory points of each observed vehicle. The selection of a point from the vehicle detection may influence the precision of point localization in the world space. One solution is to construct the 3D bounding box [22, 23] around the vehicle and select the middle point of the vehicle base laying on the ground plane. However, this 3D bounding box construction is computationally expensive as it relies on the silhouette of the vehicle. We use middle point of 2D detections' bottom-line provided instead, as this point performs the best from available data.

2.3.2 Transformation from Image to World Coordinate System

The transformation process assumes that the calibration parameters for each camera are known. We used camera calibration provided with the dataset which was in the form of a homography matrix describing the transformation from the image plane to GPS coordinates in DD (*Decimal degrees*) format. The transformation between coordinate systems is a straightforward operation made only by matrix multiplication — homography matrix $\mathbf{H}$ multiplied by GPS coordinates in homogeneous format to transform from GPS to image plane (*i.e.* inverted homography matrix multiplied by image point in homogeneous format to transform image plane point to GPS coordinates). Two cameras in the dataset (*c005* and *c035*) are fisheye cameras and thus compensation for the distortion of the point is necessary before transformation to GPS coordinate systems.

Projection of GPS to Linear System Since GPS coordinates are known in the DD format and not as positions on the flat plane, the distances and positions do not correspond precisely to the real world because of the curvature of the Earth. Although distances in the DD format can be computed by *Haversine formula*, they can potentially suffer by some inaccuracies, and thus transformation to the linear space was done. We used transformation from *EPSG:4326* to *EPSG:26975* (corresponds to *North Iowa* where the dataset was collected).

Camera View Area Estimation A part of our solution is automatic detection of camera view area (polygon covering part of real world, where objects can be seen by a specific camera). For each of the observed vehicles, the two bottom corners of the vehicle's bounding box are transformed to the linear space. Convex hull of the points is used as a polygon covering camera's view area. An example of the points used during the detection of a camera's viewpoint together with the corresponding linear space is depicted in Figure 3. Examples of localized viewpoints for a part of all cameras in a session can be seen in Figure 7. Our experiments show that it is convenient not to use all detections, but to limit these detections in some way — detections must be larger than 1000 pixels (in area) and all detections should be no further than 300 meters from camera in the linear space.

2.3.3 Multi-Camera Tracks Positional Matching

Positional matching between vehicle tracks observed by multiple cameras at one session is based on comparing mutual positions of individual trajectory points from multiple cameras in the real-world linear coordinate system in each time step. Trajectory points from each camera in a session

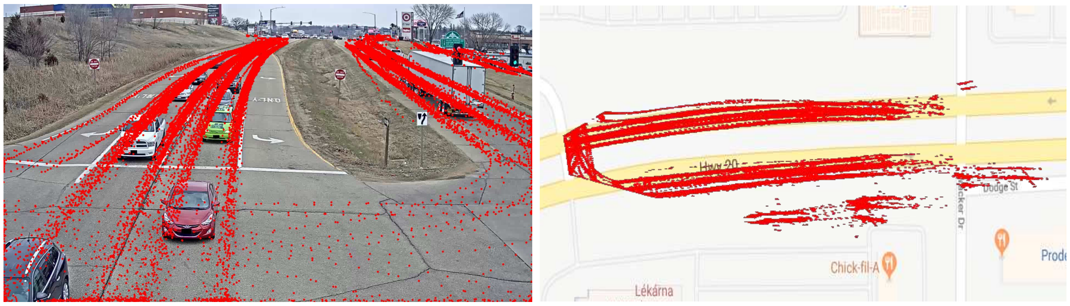

Figure 3. *left:* Detections' bottom points used for localization of camera's view area. *right:* Corresponding points transformed to the linear space (viewpoint selected as convex hull of these points).

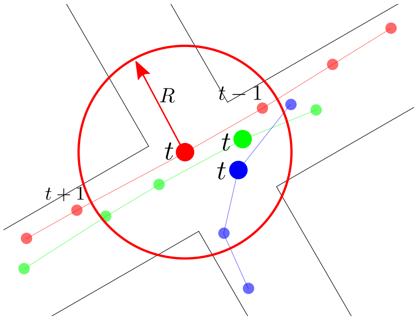

Figure 4. Visualization of the positional matching method. **Red** and **green** tracks corresponds to same vehicle observed by multiple cameras. **Blue** track represents another track of another vehicle. Positional matching iteratively determines if points from one trajectory corresponds to points from another trajectories by constructing a circle with radius R in each time-step t and matches are accumulated to the *matching matrix*. This matrix contains a score for each possible combination of camera-track pairs.

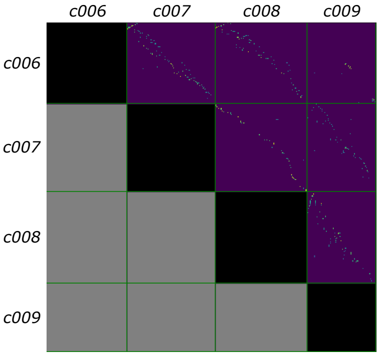

Figure 5. Matching matrix for *S02*. Each block size differs based on a count of tracks detected in single cameras.

are sorted by the time of their observation (*time steps*). For each time step and each trajectory point observed by one camera, we construct a circle in the linear space with radius R and we are looking for trajectory points from other cameras in the session which are contained inside the constructed circle (for better understanding, please see Figure 4). These pairwise camera–track *matches* are accumulated in a *matching matrix* M. This matrix contains all possible matches from each track in one camera to all tracks in the other cameras. An example of the matching matrix can be seen in Figure 5.

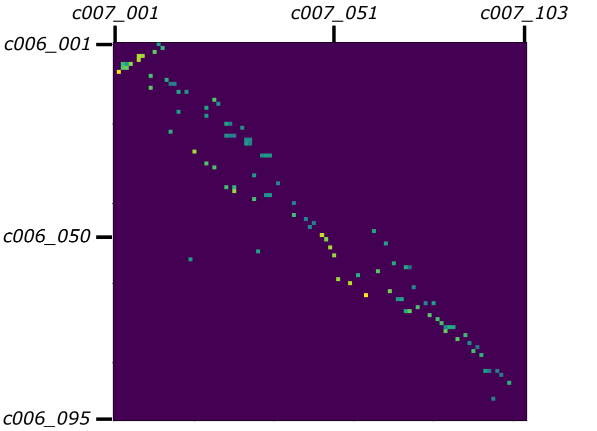

Figure 6. Matrix corresponding to sub-block *c006-c007* from Figure 5. Each cell contains the count of matches between the track in each row and track in each column.

The matching matrix is split in pairwise camera blocks (see Figure 6). In each row of these blocks, we are looking for maximal accumulated values in other camera blocks separately using *Linear Sum Assignment* solver. These maximal values correspond to the best matching tracks between all cameras in the session. Best matches are further processed and joined into bigger groups if some element of *pairs, triplets, quadruplets,...* is missing in the other set which has at least one shared element.

Even visual features can be employed in the proposed method for solving multi-camera tracking problem. We are able to extract features (by using the same convolutional neural network with pooling as described in Section 2.2) from vehicle tracks given by camera-track indices of the matching matrix and to construct a pairwise distance matrix with the same shape as the matching matrix. This distance matrix is then used for weighting of elements in the matching matrix.

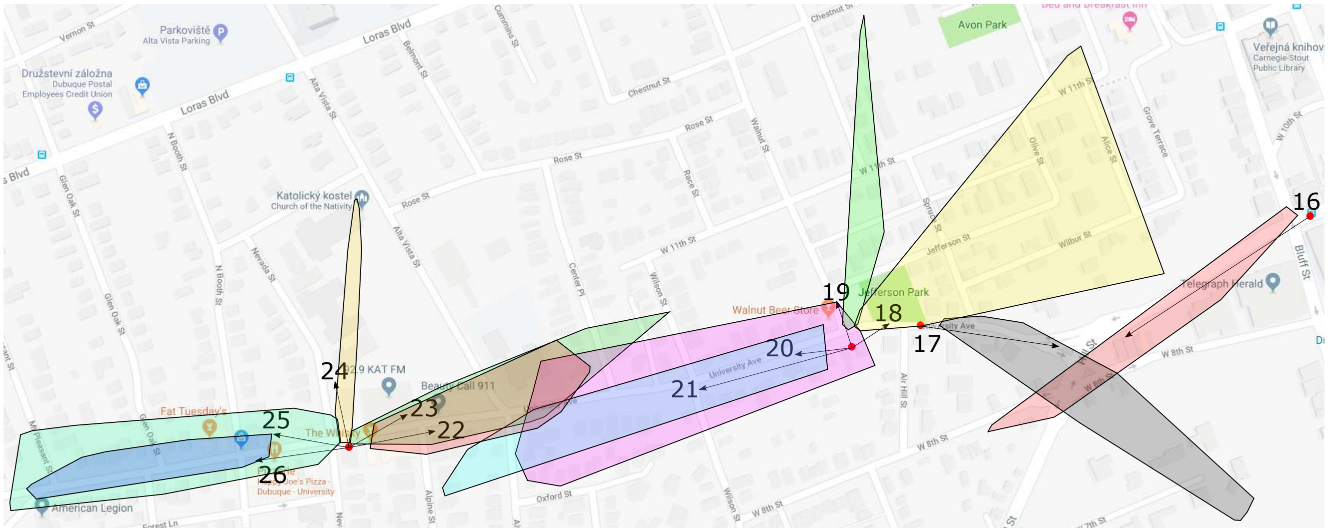

Figure 7. Localized viewpoints for part of all cameras in *S04* and *S05*.

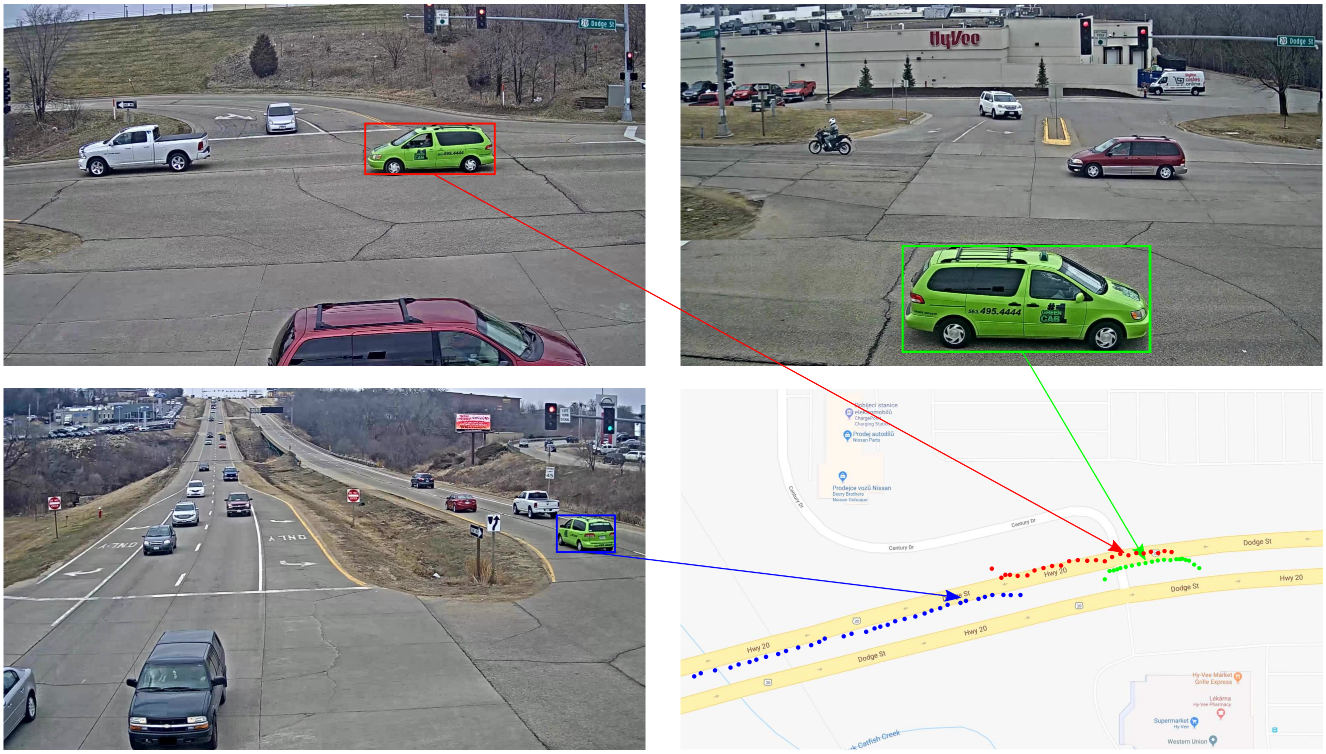

Figure 8. Example result of our positional matching method with projection of trajectories points into linear coordinate system. Arrows depict transformed points from image space to real-world space in specific time-step (displayed camera frame).

3. Experiments

This section describes the experiments done while evaluating both challenge tracks.

3.1. Evaluation of Vehicle Re-Identification

We employed our own version of evaluation on the **training** data in same manner as official evaluation is performed. We extract 1,000 images from training data which were used as our *query* images. This query set was used for evaluation of networks performance on the training set.

Comparison of different variants of our trained networks and big differences between the performance on the training and the testing data can be seen in the Table 3. Our original network design (CNN trained for the identification task with the cross-entropy loss and addition CNN for time-pooling –

Variant	Sess	IDF1	IDP	IDR	Prec	Rec
R=5m	02	0.0640	0.1240	0.0431	0.1560	0.0543
R=5m + feats	02	0.0664	0.1315	0.0444	0.1608	0.0543
R=5m	02,05	0.0480	0.0264	0.2623	0.0510	0.5064
R=10m	02,05	0.0340	0.0181	0.2865	0.0365	0.5792
R=10m + feats	02,05	0.0358	0.0190	0.3015	0.0365	0.5792

Table 1. Results for different variants of *positional matching* from AIC evaluation server. R represents different radius of circle for positional matching. **Feats** represent using of visual features extracted from tracks.

Rank	Team ID	Team Name	IDF Score
1	21	UWIPL	0.7059
2	49	DDashcam	0.6865
3	12	Traffic Brain	0.6653
4	53	Desire	0.6644
5	97	ANU AI city tracking and Re-ID team	0.6519
6	59	Zero_One	0.5987
7	36	DGRC	0.4924
8	107	IIAI-VOS	0.4504
9	104	Owlish	0.3369
10	52	CUNY-NPU	0.2850
11	48	BUPT-MCPRL	0.2846
12	115	KITMCT	0.2272
13	108	FirstBird	0.2183
14	7	iter1004	0.2149
15	60	i-TRACK	0.1752
16	87	DukBaeGi	0.1710
17	79	Alpha	0.1634
18	**64**	**GRAPH@FIT**	**0.0664**
19	43	VPUteam	0.0566
20	128	YXWM	0.0544
21	68	BUPT_MCPRL_MTMCT	0.0473
22	45	Insight DCU	0.0326

Table 2. Final ranking for multi-camera tracking part (Track 1) of NVIDIA AI City Challenge 2019.

LFTD) was tested with a different combination of training data for both these tasks (feature extraction, time pooling). We trained our network the CarsReId74k [25] or we pre-trained the network on this dataset and we fine-tune on AIC-ReID training data after that.

The results on our evaluation set were promising. However, after evaluating on the testing set, the results were very unsatisfactory. A big performance drop can be seen when training and testing evaluation is compared. This is probably caused by the size of the AIC-ReID dataset as the number vehicles and their images included in the dataset is rather small.

We tried to replicate at least the baseline results provided by the authors of the challenge [28]. We trained MobileNet with triplet loss function for feature embedding (128 dimensions) with semi-hard batch sampling. Again, results based on our evaluation procedure were promising, contrary to the final obtained results. However, the performance on the testing set is better. The final rank for this part (Track 1) can be found in the Table 4.

Training setup

A feature extractor for the CarsReId74k dataset was trained with LR 0.0001, Adam optimizer, batch size 16 for 50 epochs, while fine-tuning on the AIC-ReID dataset was done for 20 epochs with the same hyperparameters. We use image modifications (IM) during training as proposed by Sochor *et al*. [22, 23] — specifically we use **alterHSV** and **imageDrop**.

In the case of MobileNet with triplet loss trained for feature embedding, batch size 80 (4 samples for 20 vehicle identities) was used. The network was trained with LR 0.0003 with Adam optimizer for 150 epochs.

Feature aggregation network (LFTD) was trained for weighted euclidean distance (WE) with LR $10^{-4.4}$ using Adam optimizer, contrastive loss with margin 2.0, 30 rounds of hard negative mining and final features length 1024.

3.2. Evaluation of Multi-Camera Tracking

We tried to compute positional matching for different circle radius $R = \{5, 10\}$ with and without visual features. Generated files with results contained ≈ 3 millions of rows and the obtained results are unsatisfactory. This was probably caused by *joining* obtained matching set to bigger sets as this results into corruption of time constrains. This led to selection of large number of false positive tracks which was confirmed by evaluation of session S02 only with better IDF1 score. Unfortunately, due to time reasons we were not able to process more experiments and our method still needs more evaluation. All evaluated variants can be found in Table 1. The final rank for this part (Track1) can be found in Table 2.

4. Conclusions

We participated in two tasks of the NVIDIA AI City Challenge 2019: the vehicle re-identification task and multi-camera tracking task. Our solution for vehicle re-identification is based on convolutional neural network and time pooling of the feature vectors extracted from the observed vehicles. For the multi-camera tracking part, we propose a method for matching of vehicle trajectory points in the real-world linear coordinate system space. This approach can be also combined with extraction of feature vectors for all observed and pre-matched tracks.

Acknowledgements

This work was supported by The Ministry of Education, Youth and Sports of the Czech Republic from the National Programme of Sustainability (NPU II); project IT4Innovations excellence in science - LQ1602. Also, this work was supported by TACR project "SMARTCarPark", TH03010529.

						Train evaluation					**Test evaluation (server)**				
Net	**Loss**	**Pooling**	**Mods**	**Ext. Data**	**Pool. Data**	**mAP**	**H@1**	**H@5**	**H@10**	**H@20**	**mAP**	**H@1**	**H@5**	**H@10**	**H@20**
RN-50	Xent	avg	-	CR	-	0.306	0.213	0.377	0.484	0.641	-	-	-	-	-
RN-50	Xent	LFTD	-	CR	CR	0.331	0.238	0.403	0.522	0.667	-	-	-	-	-
RN-50	Xent	LFTD	-	CR	CR+AIC	0.786	0.711	0.881	0.933	0.97	-	-	-	-	-
IRN-v2	Xent	avg	-	CR	-	0.357	0.256	0.447	0.562	0.699	-	-	-	-	-
IRN-v2	Xent	LFTD	-	CR	CR	0.375	0.273	0.468	0.592	0.73	-	-	-	-	-
IRN-v2	Xent	LFTD	-	CR	CR+AIC	0.766	0.678	0.875	0.933	0.969	-	-	-	-	-
RN-50	Xent	avg	IM	CR	-	0.297	0.209	0.361	0.472	0.623	-	-	-	-	-
RN-50	Xent	LFTD	IM	CR	CR	0.311	0.219	0.381	0.492	0.645	-	-	-	-	-
RN-50	Xent	LFTD	IM	CR	CR+AIC	0.789	0.715	0.88	0.928	0.965	-	-	-	-	-
IRN-v2	Xent	avg	IM	CR	-	0.346	0.251	0.423	0.543	0.69	-	-	-	-	-
IRN-v2	Xent	LFTD	IM	CR	CR	0.362	0.259	0.451	0.577	0.723	0.0568	0.1141	0.1141	0.1179	0.1331
IRN-v2	Xent	LFTD	IM	CR	CR+AIC	0.766	0.683	0.87	0.925	0.968	-	-	-	-	-
RN-50	Xent	avg	-	CR+AIC	-	0.844	0.768	0.942	0.972	0.99	-	-	-	-	-
RN-50	Xent	LFTD	-	CR+AIC	CR	0.741	0.655	0.85	0.91	0.956	-	-	-	-	-
RN-50	Xent	LFTD	-	CR+AIC	CR+AIC	0.983	0.976	0.993	0.996	0.998	-	-	-	-	-
IRN-v2	Xent	avg	-	CR+AIC	-	0.988	0.981	0.997	0.999	1	-	-	-	-	-
IRN-v2	Xent	LFTD	-	CR+AIC	CR	0.978	0.968	0.99	0.994	0.996	0.2329	0.3536	0.3555	0.3650	0.4068
IRN-v2	Xent	LFTD	-	CR+AIC	CR+AIC	0.992	0.989	0.995	0.995	0.996	0.2420	0.3498	0.3508	0.3574	0.3926
RN-50	Xent	avg	IM	CR+AIC	-	0.829	0.752	0.928	0.965	0.988	-	-	-	-	-
RN-50	Xent	LFTD	IM	CR+AIC	CR	0.726	0.638	0.833	0.896	0.948	0.1428	0.2861	0.2871	0.2928	0.3137
RN-50	Xent	LFTD	IM	CR+AIC	CR+AIC	0.982	0.972	0.994	0.996	0.997	-	-	-	-	-
IRN-v2	Xent	avg	IM	CR+AIC	-	0.986	0.978	0.997	0.999	1	-	-	-	-	-
IRN-v2	Xent	LFTD	IM	CR+AIC	CR	0.976	0.963	0.992	0.997	0.998	0.2311	0.3622	0.3631	0.3641	0.3992
IRN-v2	Xent	LFTD	IM	CR+AIC	CR+AIC	0.991	0.986	0.996	0.999	1	0.2449	0.3707	0.3717	0.3755	0.4240
MobNet	Tri	avg	-	AIC	-	0.973	0.953	0.997	0.999	0.999	0.2883	0.3916	0.3916	0.4002	0.4496
MobNet	Tri	LFTD	-	AIC	AIC	0.976	0.959	0.995	0.998	1	0.2582	0.3432	0.3451	0.3489	0.3850
MobNet	Tri	avg	IM+Flip	AIC	-	0.962	0.934	0.995	0.998	0.999	-	-	-	-	-
MobNet	Tri	avg	Flip	AIC	-	0.989	0.978	1	1	1	**0.3157**	**0.4221**	**0.4221**	**0.4278**	**0.4270**

Table 3. Results for different variants of CNN feature extractors trained using different training setups (dataset used, network design, time pooling, data augmentation) and big gaps in our evaluation on training data and official evaluation.

Net: RN-50 – ResNet50, IRN – InceptionResNet, MobNet – MobileNet.

Loss: Xent – cross-entropy loss, Tri – Triplet loss.

Pooling: avg – average time-pooling, LFTD – our time-pooling method [25].

Mods (data augmentation used while training): IM – Image modifications [22, 23], Flip – Horizontal flip of image.

Extractor/pooling data (data used for training): CR – CarsReId74k, AIC data, CR+AIC combination of them.

Rank	Team ID	Team Name	mAP Score	Rank	Team ID	Team Name	mAP Score
1	59	Zero_One	0.8554	43	80	IFP	0.3266
2	21	UWIPL	0.7917	44	1	SJSU_Anastasiu	0.3242
3	97	ANU AI city tracking and Re-ID team	0.7589	**45**	**64**	**GRAPH@FIT**	**0.3157**
4	4	expensiveGPUs	0.7560	46	104	Owlish	0.3090
5	12	Traffic Brain	0.7302	47	33	HRI-SH	0.3081
6	53	Desire	0.6793	48	50	AHUer	0.3047
7	131	XINGZHI	0.6091	49	76	GOGOGO	0.3039
8	5	UMD_RC	0.6078	50	79	Alpha	0.2965
9	78	MVM	0.5862	51	63	QMUL	0.2928
10	127	flyZJ	0.5827	52	6	UWACS	0.2912
11	92	APTX	0.5725	53	108	FirstBird	0.2867
12	154	XJTU-SMILES Lab	0.5693	54	46	SkyRoads	0.2766
13	27	INRIA STARS	0.5344	55	87	DukBaeGi	0.2763
14	107	IIAI-VOS	0.5229	56	120	YXX	0.2713
15	132	AlphaVehicle	0.5096	57	117	AI Pioneers	0.2693
16	114	Casia&Sg.panasonic&Bjtu	0.5040	58	145	Luo Jia Team	0.2599
17	23	KFC	0.5028	59	68	BUPT_MCPRL_MTMCT	0.2531
18	24	Avengers5	0.4998	60	43	VPUteam	0.2505
19	40	AI Bandits	0.4631	61	57	UTF-Puma	0.2481
20	48	BUPT-MCPRL	0.4610	62	55	reiddoneright	0.2451
21	7	iter1004	0.4406	63	18	Team Argus	0.2347
22	37	VCA	0.4195	64	62	CQUPT_EINI	0.2345
23	52	CUNY-NPU	0.4096	65	91	SJK	0.2228
24	14	CVHCI-KIT	0.4014	66	85	Bohemian Rhapsody	0.2184
25	113	HCMUS	0.4008	67	49	DDashcam	0.2176
26	70	helloketty	0.3960	68	25	GIST	0.2110
27	54	zhengge	0.3922	69	159	Walrus	0.2063
28	36	DGRC	0.3887	70	146	NCTUAI	0.2018
29	35	VD-blue	0.3814	71	163	TeamFellows	0.1748
30	41	SYSUITS	0.3769	72	139	Alpha_TSZ	0.1627
31	30	CheeseEgg	0.3741	73	125	BDTitan	0.1598
32	17	CSAI	0.3723	74	28	228Office	0.1583
33	51	ZJU	0.3689	75	15	ReId-this	0.1559
34	22	singlerace	0.3675	76	116	Conduent Labs India	0.0852
35	89	MMVG-AlibabaAIC-INF	0.3566	77	44	BUPT-CSD-Vision	0.0782
36	26	SYSU-ISENET	0.3503	78	58	Ann Arbor AI Amateurs	0.0340
37	124	BUPTCP	0.3496	79	45	Insight DCU	0.0322
38	96	SDU&Oeasy	0.3430	80	60	i-TRACK	0.0146
39	72	VehicleJian	0.3378	81	19	UCF_reid	0.0025
40	20	TJU0432	0.3339	82	128	Robint	0.0022
41	29	NCTU-NOL	0.3325	83	13	KAIST MSC	0.0004
42	47	ZJU_ReID	0.3317	84	133	AIIT-Jack	0.0003

Table 4. Final ranking for the re-identification part (Track 2) of NVIDIA AI City Challenge 2019.

References

[1] C. Arth, C. Leistner, and H. Bischof. Object reacquisition and tracking in large-scale smart camera networks. In *2007 First ACM/IEEE International Conference on Distributed Smart Cameras*, pages 156–163. IEEE, 2007.

[2] L. Chen, H. Yang, J. Zhu, Q. Zhou, S. Wu, and Z. Gao. Deep spatial-temporal fusion network for video-based person re-identification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, July 2017.

[3] W. Chen, X. Chen, J. Zhang, and K. Huang. Beyond triplet loss: A deep quadruplet network for person re-identification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[4] D. Cheng, Y. Gong, S. Zhou, J. Wang, and N. Zheng. Person re-identification by multi-channel parts-based cnn with improved triplet loss function. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[5] S. Du, M. Ibrahim, M. Shehata, and W. Badawy. Automatic license plate recognition (ALPR): A state-of-the-art review. *IEEE Trans. on Circuits and Systems for Video Technology*, 23(2):311–325, 2013.

[6] R. S. Feris, B. Siddiquie, J. Petterson, Y. Zhai, A. Datta, L. M. Brown, and S. Pankanti. Large-scale vehicle detection, indexing, and search in urban surveillance videos. *IEEE Trans. on Multimedia*, 14(1):28–42, 2012.

[7] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[8] A. Hermans, L. Beyer, and B. Leibe. In defense of the triplet loss for person re-identification. arXiv:1703.07737, 2017.

[9] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[10] D. Kinga and J. B. Adam. A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[11] K. Kluwak, J. Segen, M. Kulbacki, A. Drabik, and K. Wojciechowski. *ALPR - Extension to Traditional Plate Recognition Methods*, pages 755–764. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[12] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3D object representations for fine-grained categorization. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, June 2013.

[13] M. Köstinger, M. Hirzer, P. Wohlhart, P. M. Roth, and H. Bischof. Large scale metric learning from equivalence constraints. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2288–2295, June 2012.

[14] R. Kumar, E. Weill, F. Aghdasi, and P. Sriram. Vehicle re-identification: an efficient baseline using triplet embedding. *arXiv preprint arXiv:1901.01015*, 2019.

[15] S. Liao, Y. Hu, X. Zhu, and S. Z. Li. Person re-identification by local maximal occurrence representation and metric learning. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[16] H. Liu, Y. Tian, Y. Yang, L. Pang, and T. Huang. Deep relative distance learning: Tell the difference between similar vehicles. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[17] X. Liu, W. Liu, T. Mei, and H. Ma. A deep learning-based approach to progressive vehicle re-identification for urban surveillance. In *ECCV*, pages 869–884. Springer, 2016.

[18] N. McLaughlin, J. Martinez del Rincon, and P. Miller. Recurrent convolutional network for video-based person re-identification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[19] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[20] Y. Shen, T. Xiao, H. Li, S. Yi, and X. Wang. Learning deep neural networks for vehicle re-id with visual-spatio-temporal path proposals. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[21] H. Shi, Y. Yang, X. Zhu, S. Liao, Z. Lei, W. Zheng, and S. Z. Li. *Embedding Deep Metric for Person Re-identification: A Study Against Large Variations*, pages 732–748. Springer International Publishing, Cham, 2016.

[22] J. Sochor, A. Herout, and J. Havel. BoxCars: 3D boxes as CNN input for improved fine-grained vehicle recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[23] J. Sochor, J. Špaňhel, and A. Herout. Boxcars: Improving fine-grained recognition of vehicles using 3-d bounding boxes in traffic surveillance. *IEEE Transactions on Intelligent Transportation Systems*, PP(99):1–12, 2018.

[24] J. Špaňhel, J. Sochor, R. Juránek, A. Herout, L. Maršík, and P. Zemčík. Holistic recognition of low quality license plates by cnn using track annotated data. In *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 1–6. IEEE, Aug 2017.

[25] J. Špaňhel, J. Sochor, R. Juránek, P. Dobeš, V. Bartl, and A. Herout. Learning feature aggregation in temporal domain for re-identificationn. arXiv:1903.05244, 2019.

[26] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, volume 4, page 12, 2017.

[27] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014.

[28] Z. Tang, M. Naphade, M.-Y. Liu, X. Yang, S. Birchfield, S. Wang, R. Kumar, D. Anastasiu, and J.-N. Hwang. Cityflow: A city-scale benchmark for multi-target multi-camera vehicle tracking and re-identification. *arXiv preprint arXiv:1903.09254*, 2019.

[29] Z. Wang, L. Tang, X. Liu, Z. Yao, S. Yi, J. Shao, J. Yan, S. Wang, H. Li, and X. Wang. Orientation invariant feature embedding and spatial temporal regularization for vehicle re-identification. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[30] Y. Wei, N. Song, L. Ke, M.-C. Chang, and S. Lyu. Street object detection / tracking for ai city traffic analysis. *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pages 1–5, 2017.

[31] Y. Wen, Y. Lu, J. Yan, Z. Zhou, K. M. von Deneen, and P. Shi. An algorithm for license plate recognition applied to intelligent transportation system. *IEEE Trans. on Intelligent Transportation Systems*, 12(3):830–845, 2011.

[32] S. Xu, Y. Cheng, K. Gu, Y. Yang, S. Chang, and P. Zhou. Jointly attentive spatial-temporal pooling networks for video-based person re-identification. In *ICCV (to appear)*, 2017.

[33] K. Yan, Y. Tian, Y. Wang, W. Zeng, and T. Huang. Exploiting multi-grain ranking constraints for precisely searching visually-similar vehicles. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[34] Y. Yan, B. Ni, Z. Song, C. Ma, Y. Yan, and X. Yang. *Person Re-identification via Recurrent Feature Aggregation*, pages 701–716. Springer International Publishing, Cham, 2016.

[35] J. Yang, P. Ren, D. Zhang, D. Chen, F. Wen, H. Li, and G. Hua. Neural aggregation network for video face recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[36] L. Yang, P. Luo, C. Change Loy, and X. Tang. A large-scale car dataset for fine-grained categorization and verification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[37] J. You, A. Wu, X. Li, and W.-S. Zheng. Top-push video-based person re-identification. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[38] D. Zapletal and A. Herout. Vehicle re-identification for automatic video traffic surveillance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 25–31, 2016.

[39] W. Zhang, X. Yu, and X. He. Learning bidirectional temporal cues for video-based person re-identification. *IEEE Transactions on Circuits and Systems for Video Technology*, PP(99):1–1, 2017.

[40] Y. Zhang, D. Liu, and Z.-J. Zha. Improving triplet-wise training of convolutional neural network for vehicle re-identification. In *Multimedia and Expo (ICME), 2017 IEEE International Conference on*, pages 1386–1391. IEEE, 2017.