Vehicle Verification Using Deep Learning for Connected Vehicle Sharing Systems

Hansi Liu
WINLAB, Rutgers University
North Brunswick, New Jersey
hansiiii@winlab.rutgers.edu

ABSTRACT

Information sharing in connected vehicle systems helps each participating vehicle to have a more complete and expanded sensing range beyond its own sensing capability. When sharing visual traffic information among vehicle nodes, it is of great significance to identify overlapping components and associate objects in common to create an accurate and complete surrounding scene. This paper Extends FusionEye, a study of perception sharing, by exploring deep learning approaches for real time vehicle verification tasks. We propose two deep neural network architectures inspired by ResNet and train the neural networks using FusionEye's dataset. Preliminary results show that when learning from vehicle's appearances and kinematic information, the verification accuracy reaches 92%, which provides possible solution for real time system.

CCS CONCEPTS

• **Computing methodologies → Image representations**; **Neural networks**; *Object identification*; *Matching*.

KEYWORDS

connected vehicles; vehicle verification; vehicle association; vehicle re-identification; deep learning; Siamese network

ACM Reference Format:
Hansi Liu. 2019. Vehicle Verification Using Deep Learning for Connected Vehicle Sharing Systems. In *ACM MobiSys 2019 Rising Stars Forum (RisingStarsForum'19), June 21, 2019, Seoul, Republic of Korea.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3325425.3329944

1 INTRODUCTION

Complete and accurate awareness of the surrounding traffic environment is crucial to the robustness of automated driving and advanced driver assistance systems (ADAS). For a smart vehicle, the particularly important information it needs to acquire while driving on the road is the positions and status of other surrounding vehicles as they are the major moving participants of the traffic. Knowing its surrounding vehicles' positions can help smart vehicles make better decisions of driving behavior and thus reduce the probability of accidents. Current vehicles are capable of capturing its surrounding environments using its on-board sensors, such as cameras or lidars, which rely heavily on line-of-sight sensing. However, these sensors suffer from limited sensing ranges as well as occlusions and thus are incapable of acquiring full information of the surrounding environment, which degrades the performances of smart vehicles or autonomous driving systems.

Besides increasing the quantity of sensors on vehicles, which increases a vehicle's cost and technological complexity, researchers have been exploring approaches leveraging connected vehicle systems to share traffic information between each other so that one vehicle may have information beyond its sensing capability with the help of mobile networks. In such connected vehicle systems, a vehicle can announce its own status including GPS position or velocity to other vehicle nodes [3], broadcast estimated locations of the surrounding vehicles in its view [6], or even share its own perception in the form of dense point clouds among the vehicle network [13].

The sharing of visual information between vehicles need to be both complete and accurate, as there can be overlaps between two views. For these connected sharing systems, how to handle the visual information in common is significant to creating a correct global map or scene of the environment. More recently, FusionEye [10] proposed a framework for sharing and associating surrounding vehicles' locations using different visual features and criteria. It uses a bipartite graph merging algorithm to determine if a detected vehicle's profile image captured by on-board cameras of two different observing vehicles belong to the same vehicle. In the topology association phase, it uses both position information as well as visual representation of the detected vehicle to find an optimal matching of a bipartite graph and makes association decisions based on the matching. However, the algorithm has a limitation in that it assumes there are always vehicles in common in two views. Even if there are no common vehicles, the merging algorithm would still generate a match between two views and thus create false positives as it associates two different vehicles as if they were the same one.

This work seeks to improve FusionEye's accuracy and address the above limitation. Intuitively, our goal is to find a more robust representation for each vehicle's profile image so that it allows generating a dissimilarity score between two vehicles that is large for different vehicles and small for the same ones. This enables further pruning of the bipartite graph links with high dissimilarity scores to avoid false positive predictions. Unlike FusionEye, which adopts hand crafted features to represent each vehicle, we resort to deep neural networks to learn a dissimilarity metric for each vehicle. The idea of metric learning has been explored intensively and achieved remarkable results in the field of computer vision for face verification problems [9][17], in which most of the face images are of unique size and free of occlusion or drastic illumination variance. However, our vehicle images are captured in motion with various image sizes, large illumination variance and cropping or occlusion of the vehicle body. Thus it is worth exploring whether metric learning using deep neural networks can help associating vehicles from different views in real time connected vehicle systems.

In this paper we adopt the Siamese architecture, a common approach for object verification tasks, and propose a network architecture based on ResNet-18 [8] to solve the vehicle verification/association problem. We train the network using appearances (i.e., raw pixel information) and kinematic information (GPS measurements and estimated depth information) of the detected vehicles, to learn a robust dissimilarity representation of each vehicle image. In the evaluation, we compute the Euclidean distance between the network output features of vehicle pairs and determine their similarity based on the computed score. We compare our association accuracy with FusionEye's merging accuracy and observe that our method significantly improves the association accuracy.

2 NETWORK ARCHITECTURE

2.1 Siamese Architecture

For our vehicle association problem, we wish to find a representation for each detected vehicle such that their distance is small if they are similar and large if dissimilar. This task of representation learning has also been explored in the field of face/person verification. While a typical pre-trained ImageNet [4] deep neural network is good for classification, the corresponding Siamese architecture can be adopted to solve verification problems. A diagram of Siamese architecture is shown in Fig. 1. In the training phase, two branches of the architecture are trained with shared weights using a batch of image pairs and their similarity ground truth as inputs, and the output features of each branch is fed into the loss function to compute the difference between the similarity of each batch and the ground truth. In the test phase, only one branch of the Siamese architecture is used and serves as a feature extractor that outputs discriminative feature vectors of the input query images, and we can compute Euclidean distance between these features to determine the similarity between the testing images. In the experiment, we adopt and modify the ImageNet pre-trained ResNet-18 [8] as the main body of our Siamese architecture, as shown in Fig. 2. Considering ResNet's outstanding performance on the ImageNet dataset, we anticipate it to behave as a good feature extractor at the starting point of the training process and further fine-tuning it can make the network generalize well to our vehicle dataset.

2.2 Contrastive Loss

Unlike recognition and classification tasks, where the output of a deep network is usually a vector of probability of each category, the output of our network is a high dimensional feature vector that can be used to compute similarity score with other output features under Euclidean distance. We adopt contrastive loss [2][7] during the training process, which "pulls" the features of dissimilar pairs of vehicles further apart and "pushes" those of similar pairs closer together in the high dimensional space. The contrastive loss

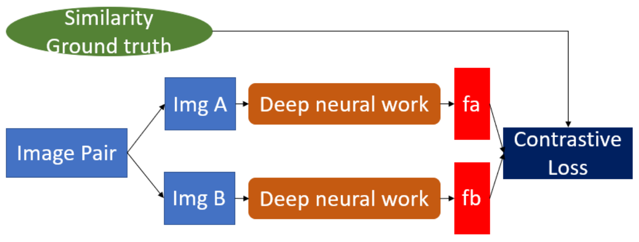


Figure 1: Siamese architecture

function is expressed as:

$$L = \frac{1}{2N}\sum_{i=1}^{N}(1-y)d^2 + (y)max(margin - d, 0)^2 \qquad (1)$$

where N represents the total number of training sample pairs, y denotes the similarity ground truth for each pair. We use "1" to represent dissimilar and "0" to represent similar. $d = ||f_a - f_b||_2$ with f_a and f_b being the output features of the two branches of the Siamese network that is shown in Fig. 1 and the hyper-parameter *margin* denotes the threshold that quantifies the sensitivity of the loss function. For similar pairs ($y = 0$), the distance of their output features f_a and f_b is directly accumulated to the loss. For dissimilar pairs, if the distance of the output features does not exceed the margin, its value will contribute to the loss for optimization. Otherwise, the loss function will ignore the difference. Thus we can understand *margin* as a factor that essentially describes the strictness of "pulling" and "pushing", the larger *margin* becomes, the further the difference between features of a dissimilar pair f_a and f_b will be, which makes the representations of dissimilar vehicles more distinctive. However, *margin* being too large will make the network difficult to train, or even hard to converge. In the experiment, we choose the value of *margin* to be 2.0 after hyper-parameter tuning.

3 EXPERIMENT

3.1 Dataset

We construct our dataset from the frames collected in FusionEye [10]. In FusionEye's experiment setups, two observing vehicles *A* and *B* are driving side by side and recording their front views at the same time so there are some overlaps among the surrounding vehicles in each observing vehicle's recorded frame. The observing vehicle's GPS measurement for every frame is also attached and all the observed surrounding vehicles in that frame are deteced by YOLO [14] and their relative distances to the observing vehicle is also estimated.

Considering our deep neural network's input being a pair of images, each sample in our dataset should consist of a pair of vehicle patches. To construct such a sample of our dataset for the vehicle verification task, we first crop the bounding box patches from a pair of FusionEye frames at one timestamp and extract the two observing vehicles' GPS measurements along with all detected vehicles' estimated distances (with respect to the observing vehicle) from that timestamp's frame pair. Then we pick one vehicle patch from the left observing vehicle's frame and the other from the right observing vehicle's frame along with their estimated distances,

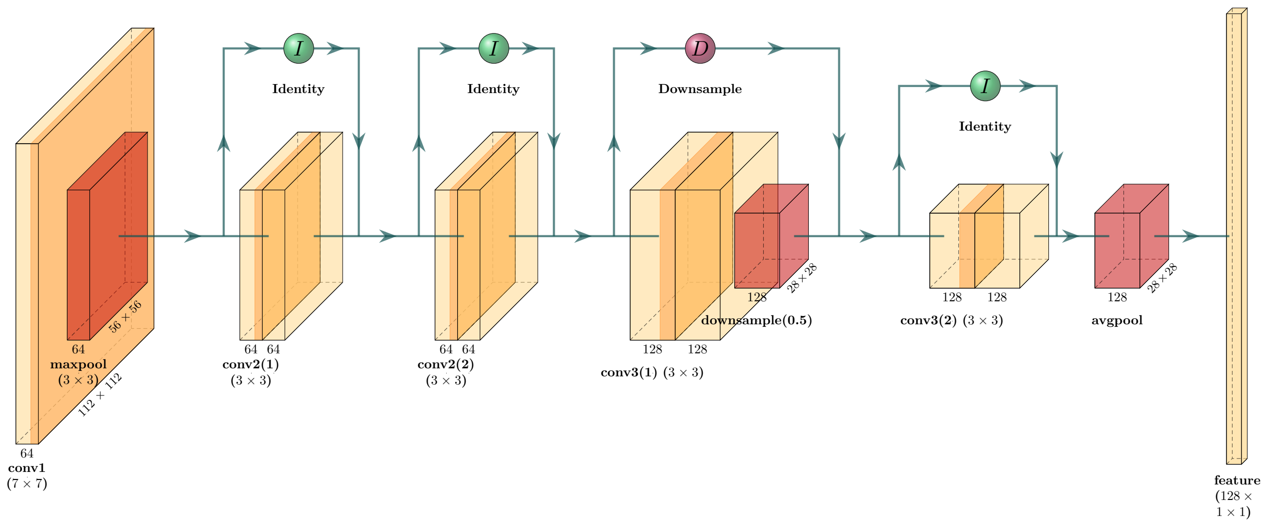


Figure 2: Detailed architecture of the modified ResNet in our experiment. Input is an image of size 224×224, and output is a feature vector of size 128×1

Table 1: The arrangement of meta-information in a sample of our dataset

Vehicle1's directory	Vehicle2's directory	Vehicle1's distance to the observing vehicles	Vehicle2's distance to the observing vehicles	GPS of two observing vehicles	similarity ground truth

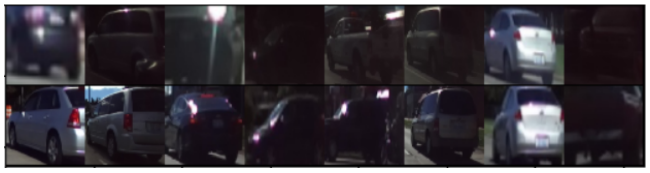


Figure 3: A batch of sample images, batch size: 8. Notice a sample contains two images in a column. We can see the dataset contains vehicle images of various sizes under various illumination conditions and different level of occlusions

GPS measurements and similarity ground truth to form a sample. Table. 1 shows the composition of a typical sample in our dataset. Suppose there are m and n detected vehicles in the left and right frames respectively, we generate $m \times n$ samples from that pair of frames.

It is worth mentioning that the images in our dataset have varying sizes since the sizes of the YOLO bounding boxes are not necessarily the same. In order to let the image fit the input dimension of our deep learning architecture, we resize all images to be 224×224. Fig. 3 shows a batch of samples after resizing. In total, we have 2777 pairs of vehicle patches, and we use 4/5 of the dataset (around 2222 pairs) as training set and the rest 1/5 (around 555 pairs) as test set.

To reduce over fitting, we apply data augmentation by randomly rotating the image within 20 degree clockwise or anti-clockwise during the training processes.

3.2 Learning from Appearances

We are interested in whether appearances information alone can be a good representation of the vehicle images. Thus we first decide to only use the pixel information from each sample to train the deep architecture. The network architecture is shown in Fig. 1 We made several modifications to the pre-trained ResNet-18 architecture as shown in Fig. 2. Considering the size and complexity of our dataset is much smaller and less complicated compared to ImageNet, we reduce the network's complexity by dropping the "conv4_x" block and the "conv5_x" block from the ResNet-18 architecture. Thus, the output feature has a dimension of $B \times 128$ where B represents batch size.

3.3 Learning from Appearances and Kinematics

FusionEye shows that information such as relative distances can also be used to associate vehicles. Thus, besides appearance information we also incorporate each vehicle's estimated depth as well as observing vehicle's GPS measurements in the training processes. To be more specific, we first compute the vertical relative distance between two vehicles A and B using GPS measurements and lane width:

$$D_{GPS} = \sqrt{(GPS_x^A - GPS_x^B)^2 + (GPS_y^A - GPS_y^B)^2 - L^2} \qquad (2)$$

where L represent the lateral distance between the two observing vehicles. Since in FusionEye's experiment two vehicles are driving

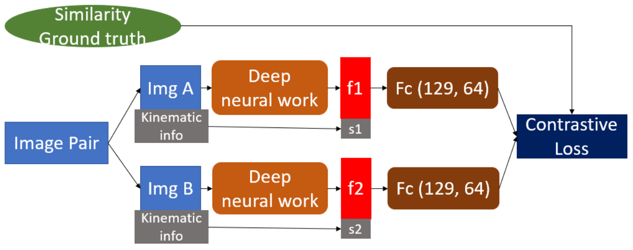


Figure 4: Network architecture for training using appearance and location information. Notice that the kinematic information is computed as a scalar and concatenated to the output feature of the deep network and then fed into another fully connected layer

side by side, the lateral relative distance between A and B is approximately the lane width. Then for a pair of vehicles in the dataset, we compute a one dimensional score for each vehicle patch in the pair (u, v) as the following:

$$[s_u, s_v] = \begin{cases} [|d_u - D_{GPS}|, d_v], & \text{if } d_u \geq d_v. \\ [d_u, |d_v - D_{GPS}|], & \text{if } d_u < d_v. \end{cases} \tag{3}$$

If vehicle A and vehicle B are observing the same vehicle, then theoretically we should have:$||d_u - d_v| - D_{GPS}| = 0$. This means that if image u and image v are the same vehicle's profile, the value of $|s_u - s_v|$ should be small. Based on this observation, we construct a new feature by concatenating this score to the 128 dimensional appearance output feature:

$$[\boldsymbol{f'_u}, \boldsymbol{f'_v}] = [[\boldsymbol{f_u}, s_u], [\boldsymbol{f_v}, s_v]] \tag{4}$$

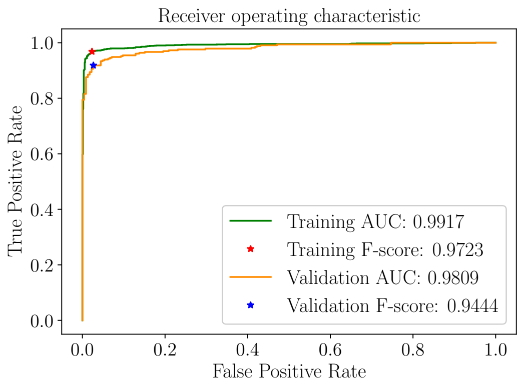
Thus, each vehicle patch is now represented by the new 129 dimensional feature vector and the Euclidean distance between two features would still be small if the two vehicles are similar. Finally, we forward the 129 dimensional features into a fully connected layer (input size: 129, output size: 64) to let the network learn by itself how much the distance score and appearance feature should contribute when determine if two vehicles are the same. Fig. 4 illustrate the architecture.

We train our deep neural network using one NVIDIA 1080Ti GPU with batch size of 32 (which contains 64 images). The starting learning rate is set to be 0.001, we reduce the learning rate by the order of ten every time the loss or validation accuracy plateaued.

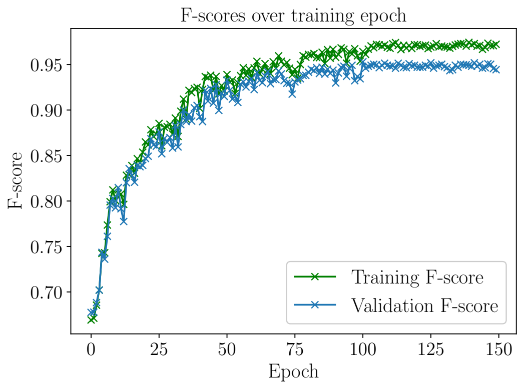
4 PRELIMINARY RESULTS & ANALYSIS

4.1 Training Results

First we train the architecture discussed in Sec. 3.2 using only appearance information from the dataset. During training we perform validation on both training set and validation set after each epoch, and compute vehicle verification accuracy in terms of F-score in order to have a direct comparison with FusionEye's results. Fig. 5(a) shows the final receiver operating characteristic (ROC) curve for both training set and validation set after 150 epochs of training. Noticed there are many F-scores corresponding to a single ROC curve as each point on the ROC curve represents a threshold for the binary classification (similar or dissimilar). The red and blue



(a) Training and validation accuracy

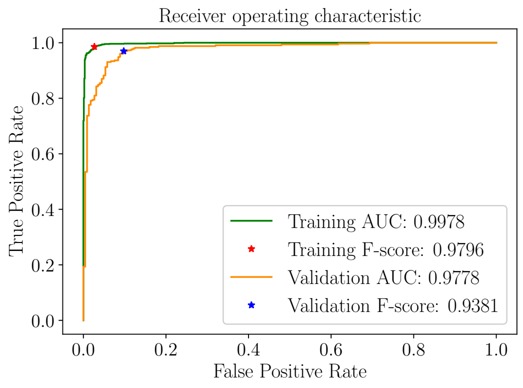


(b) History of F-scores

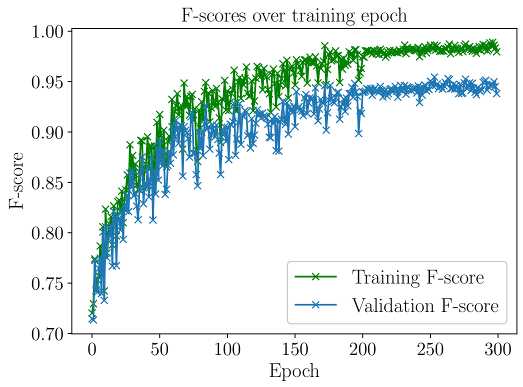
Figure 5: ROC curve and history of F-scores for training using appearance information only

star on the ROC curves represent the optimal threshold for the classifiers to have the maximum F-score. Fig. 5(b) shows the history of these F-scores. We can see from the plot that the training process converges well and reaches high accuracy around 92%.

Then we train the architecture discussed in Sec. 3.3 to learning a metric from both pixel values of the images as well as kinematic information such as each vehicle's estimated distance and GPS measurements of the camera equipped vehicles. Fig. 6 shows the ROC curve and training history in terms of F-scores after 300 epochs of training. Since we add another fully connected layer to the pretrained ResNet, the training process takes more time than simply training on appearance information. Here we observe similar performance in terms of area under curve for ROC curves as well as F-scores compared to the results of appearance based training. To compare our approach with FusionEye's bipartite association algorithm, Table 2 shows that using deep learning we can learn a more robust representation. It is worth mentioning that such representation is also compatible with bipartite association algorithm in FusionEye since we can compute edge scores using the Euclidean distances of the output features of the network. Another advantage



(a) Training and validation accuracy



(b) History of F-scores

Figure 6: ROC curve and history of F-scores for training using appearance information and kinematic information

Table 2: Vehicle verification accuracy comparisons

Method	Accuracy (F-scores)
FusionEye's bipartite merging	0.87 ± 0.1
Learning from appearance information	0.94 ± 0.02
Learning from appearance and kinematic information	0.93 ± 0.02

of these representation is that it can reduce the number of false positive in FusionEye's prediction, as we could apply the threshold derived from ROC curve and further filter out matches that are false positive in FusionEye's bipartite graph. The 64 dimensional feature representation also provide potential opportunities for faster real time transmission, as FusionEye transmitted 128 dimensional SIFT features under certain configurations.

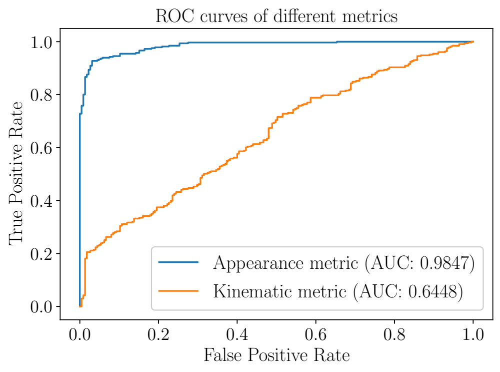


Figure 7: Two ROC curves of verification using different metrics. Blue: metrics learned from appearance information. Orange: metrics learned from kinematic information.

4.2 discussions

Considering kinematic information (GPS readings and estimated distances) is the core metric in FusionEye that helps to distinguish two vehicle's similarity at an accuracy of 86%, it is counter-intuitive to observe that the accuracy of learning from appearance is no less or even slightly better than the accuracy of learning from using both appearance and kinematic information. It seems that in the context of deep learning, the kinematic information did not contribute or even prevent the network to learn a more robust representation. one possible explanation for this inconsistency could be that similarity scores in FusionEye's bipartite merging algorithm is only compared with other scores "locally", i.e., within each pair of frames, while in our deep learning experiments the scores from all pairs of vehicles are taken into account when plotting the ROC curve. Because GPS measurement and distance estimation in FusionEye has noises, different frame pairs at different timestamps may have different noises of GPS readings. Although these noises won't affect the comparisons of scores for that timestamp's bipartite graph and the kinematic information is useful to determine whether two vehicles are same or not, it does not guarantee a good ROC curve when we use all the kinematic information scores to build a classifier. To further substantiate this analysis, we predict each vehicle pair's similarity solely using their kinematic information and compute verification accuracy as showed in Fig. 7. Compared with the blue ROC curve resulted from appearance metrics, the orange ROC curve with significantly lower area under curve (AUC) indicates that metrics learned from kinematic information failed to separate themselves apart in terms of Euclidean distance for similar and dissimilar vehicle profile images. Unlike metrics learned from appearance information, where a similar vehicle pair has a smaller similarity score (distance) and a dissimilar vehicle pair has a greater one, the scores under the kinematic metrics don't have such a good binomial distribution for accurate binary classification. Since we only show preliminary results in this paper, we are still analyzing how we can fully utilize the kinematic information to help us further increase the verification accuracy.

5 RELATED WORK

For object verification, association or re-identification tasks, deep learning approaches and Siamese architectures were first explored in face domain instead of vehicle domain. DDML [9] used a Siamese network with a generalized logistic loss function to learn a Mahalanobis metric to obtain robust feature representations for different face images. Yi et al [19] proposed a Siamese architecture with Cosine layer to compute similarity between two features and adopts binomial deviance cost function for network training. Similarly, Ahmed et al [1] adopted a Siamese network followed by a cross-input neighborhood difference layer and a summarize layer to describe the differences of two feature vectors for person re-identification problems. Other variations on the Siamese architectures have also been extensively explored recently. In FaceNet [15], a triplet loss function was proposed for robust similarity metric learning between a triplet of images. Ding et al [5] proposed a similar approach where a 3-branch deep architecture was adopted along with the triplet loss function to address person re-identification tasks.

More recently, researchers have begun to explore deep learning methods in the vehicle domain. DRDL [11] proposed a two-branch deep neural network model with coupled cluster loss function that is inspired by triplet loss and incorporate each vehicle's attribute labels to learn a robust metric. Furthermore, studies have shown that besides pixels values, spatial or temporal information is also valuable for vehicle verification tasks. Wang et al [18] used deep neural networks to extract features from 20 keypoints from a vehicle and aggregated them to construct a discriminative feature vector and refer to spatial and temporal information of each vehicle to further verify the vehicles' similarity. Liu et al [12] proposed PROVID, in which appearance features resulted from a deep convolutional neural network (CNN), license plate difference measured by a Siamese network and spatial-temporal information are fused to address vehicle verification problems. Shen et al [16] adopted a chain MRF model with a deeply learned pair-wise potential function to generate visual-spatial-temporal path proposals, which are further evaluated by a Siamese-CNN+Path-LSTM model to obtain similarity scores between pairs of query vehicle images.

6 CONCLUSIONS

In this paper we propose an approach for vehicle verification under the framework of connected vehicle systems. The proposed deep learning architectures generalize well and achieve high verification accuracy on real world dataset of reasonable complexity. The results shown in this paper is only preliminary and more experiments are expected to fully utilized the kinematic information besides pixel information of vehicle's profile image to further improve the association accuracy under more complicated circumstances. The high accuracy and low dimensionality of the feature representation indicate the potential deployment in the real time systems. For future work, it is worth exploring in real time scenarios by deploying the deep learning architectures to connected vehicle systems and evaluating communication performances in terms of transmission latency and bandwidth requirement.

ACKNOWLEDGMENTS

I would like to thank my advisor Marco Gruteser and anonymous reviewers for their valuable comments. This material is based in part upon work supported by the National Science Foundation under Grant No. CNS-1329939.

REFERENCES

[1] Ejaz Ahmed, Michael Jones, and Tim K Marks. 2015. An improved deep learning architecture for person re-identification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 3908–3916.

[2] Sumit Chopra, Raia Hadsell, Yann LeCun, et al. 2005. Learning a similarity metric discriminatively, with application to face verification. In *CVPR (1)*. 539–546.

[3] Soteris Demetriou, Puneet Jain, and Kyu-Han Kim. 2018. Codrive: Improving automobile positioning via collaborative driving. In *IEEE INFOCOM 2018-IEEE Conference on Computer Communications*. IEEE, 72–80.

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 248–255.

[5] Shengyong Ding, Liang Lin, Guangrun Wang, and Hongyang Chao. 2015. Deep feature learning with relative distance comparison for person re-identification. *Pattern Recognition* 48, 10 (2015), 2993–3003.

[6] Sae Fujii, Atsushi Fujita, Takaaki Umedu, Shigeru Kaneda, Hirozumi Yamaguchi, Teruo Higashino, and Mineo Takai. 2011. Cooperative vehicle positioning via V2V communications and onboard sensors. In *Vehicular Technology Conference (VTC Fall), 2011 IEEE*. IEEE, 1–5.

[7] Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, Vol. 2. IEEE, 1735–1742.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[9] Junlin Hu, Jiwen Lu, and Yap-Peng Tan. 2014. Discriminative deep metric learning for face verification in the wild. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1875–1882.

[10] Hansi Liu, Pengfei Ren, Shubham Jain, Mohannad Murad, Marco Gruteser, and Fan Bai. 2019. FusionEye: Perception Sharing for Connected Vehicles and its Bandwidth-Accuracy Trade-offs. In *2019 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON) (to appear)*. IEEE, 1–9.

[11] Hongye Liu, Yonghong Tian, Yaowei Yang, Lu Pang, and Tiejun Huang. 2016. Deep relative distance learning: Tell the difference between similar vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2167–2175.

[12] Xinchen Liu, Wu Liu, Tao Mei, and Huadong Ma. 2016. A deep learning-based approach to progressive vehicle re-identification for urban surveillance. In *European Conference on Computer Vision*. Springer, 869–884.

[13] Hang Qiu, Fawad Ahmad, Fan Bai, Marco Gruteser, and Ramesh Govindan. 2018. AVR: Augmented Vehicular Reality. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services (Mobisys)* (2018-01-01) *(MobiSys '18)*. ACM, Munich, Germany, 81–95. https://doi.org/10.1145/3210240.3210319

[14] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788.

[15] Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 815–823.

[16] Yantao Shen, Tong Xiao, Hongsheng Li, Shuai Yi, and Xiaogang Wang. 2017. Learning deep neural networks for vehicle re-id with visual-spatio-temporal path proposals. In *Proceedings of the IEEE International Conference on Computer Vision*. 1900–1909.

[17] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. 2014. Deepface: Closing the gap to human-level performance in face verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1701–1708.

[18] Zhongdao Wang, Luming Tang, Xihui Liu, Zhuliang Yao, Shuai Yi, Jing Shao, Junjie Yan, Shengjin Wang, Hongsheng Li, and Xiaogang Wang. 2017. Orientation invariant feature embedding and spatial temporal regularization for vehicle re-identification. In *Proceedings of the IEEE International Conference on Computer Vision*. 379–387.

[19] Dong Yi, Zhen Lei, Shengcai Liao, and Stan Z Li. 2014. Deep metric learning for person re-identification. In *2014 22nd International Conference on Pattern Recognition*. IEEE, 34–39.