Machine Learning Techniques for Vehicle Matching with Non-Overlapping Visual Features

Samuel Thornton, Sujit Dey
Department of Electrical and Computer Engineering
University of California, San Deigo
{sjthornt, dey} @ ucsd.edu

***Abstract*—Emerging Vehicle-to-Everything (V2X) technologies promise to improve the perception of streets by enabling data sharing like camera views between multiple vehicles. However, to ensure accuracy of such enhanced perception, the problem of vehicle matching becomes important; the goal of a vehicle matching system is to identify if images of vehicles seen by different cameras correspond to the same vehicle. Such a system is necessary to avoid duplicate detections for a vehicle seen by multiple cameras and to avoid detections being discarded due to a false match being made. One of the most challenging scenarios in vehicle matching is when the camera positions have very large viewpoint differences, as will commonly be the case when the cameras are in geographically separate locations like in vehicles and street infrastructure. In these scenarios, traditional handcrafted features will not be sufficient to create these correspondences due to the lack of common visual features. In this paper we will examine the performance of random forests and neural networks as classifiers for both learned features and high level visual features when used for this vehicles matching problem. Additionally, a novel dataset of vehicles from cameras with very large viewpoint differences was recorded to validate our method; our preliminary results achieve high classification accuracy with low inference time which shows the feasibility of a real time vehicle matching system.**

I. Introduction

In the United States in 2016, there were over 30,000 motor vehicle related fatalities and over 2 million more cases of bodily injury [1]. While vehicle safety has improved over the years, there are still major improvements that can be made to reduce the number of vehicle accidents; in the past decade, there have been many new systems such as the Advanced Driver Assistance Systems (ADAS) and Automatic Emergency Braking (AEB) systems that have improved vehicle safety. To support the advanced perception required by these types of systems, consumer vehicles come standard with an array of sensors such as cameras, radars, and ultrasounds which provide the necessary input data. However, these sensors all have range limitations and can have their accuracy affected by factors such as occlusion and adverse weather conditions. Even with the most sophisticated array of sensors, a single vehicle can never have a perfect perception of its surroundings; there will always be some gaps or areas beyond what its sensors can see.

One way to address this problem is through collaboration. If a collaborative vehicle environment existed where vehicles share their sensor data with other vehicles, then each vehicle could have a more complete perception of the area they are in. To generate this improved perception, there is a large data fusion challenge of creating correspondences between each vehicles data that must first be solved. One of the most common and studied type of sensors for vehicle perception is a RGB camera, and an important correspondence that must be established for this type of data is knowing when a vehicle seen by one camera is the same as a vehicle seen by another camera. This is the vehicle matching problem and solving it is an important first step to generating the collaborative perception of the vehicle's environment.

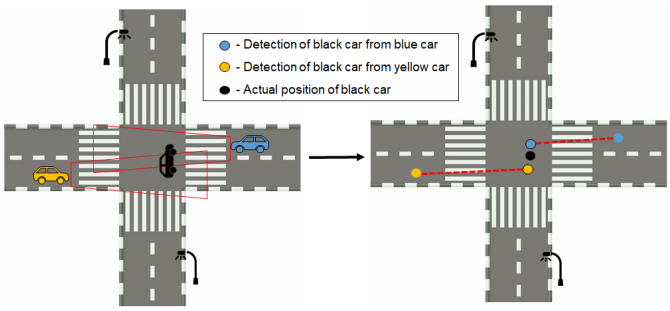

Fig. 1. Example intersection showing the need for vehicle matching: the yellow and blue car both detect different parts of the black car, but without a vehicle matching system there will be two individual detections (blue and yellow circles). If a vehicle matching system could create the correspondence between the two individual detections, then they can be merged into a more accurate detection that is closer to the ground truth (black circle).

Connected vehicle systems such as Vehicle-to-Vehicle (V2V) [2] or Vehicle-to-Everything (V2X) [3] give vehicles that have the appropriate hardware a way to communicate and share information with each other. The motivation for solving the vehicle matching problem can be seen by considering an intersection in a connected vehicle environment such as the one seen in Figure 1; there can be many vehicles at this intersection that are generating camera data and the data from each vehicle can be fused together to create a more complete perception of that area than any of the individual vehicles could create on their own. To generate a comprehensive perception of this intersection, a naive solution might be to assume that every object detected by each vehicle is unique. However if multiple cameras saw the same object, the system's perception of the intersection would be littered with duplicate detections. If we are able to match objects from one camera to another and forward this information to the perception system, then we can avoid these duplicate detections, while ensuring we do not falsely eliminate true objects.

In this paper, we look to solve the vehicle matching problem when the viewpoint difference between the cameras is large and offers few or no overlapping regions and hence no common visual features for the target vehicle; we term this as the *Non-Overlapping Vehicle Matching* (NOVeM) problem. For this problem, we consider multiple types of visual features of vehicles as seen by camera images to determine if it is possible to achieve high classification accuracy in vehicle matching without considering the physics of the vehicles.

II. Related Work

The vehicle matching problem can be considered as a subset of the general object matching problem, the latter having been studied in various contexts for decades [4]. However, most object matching techniques such as [5] rely on common visual features like Oriented FAST and Rotated BRIEF (ORB) [6] and Scale Invariant Feature Transform (SIFT) [7] to create matches which will not be possible in the case of large viewpoint differences. More recently, techniques have been proposed for vehicle re-identification, where images from surveillance cameras in different areas are matched to try and identify if the same vehicle is seen. Liu et al. [8] proposed Deep Relative Distance learning which uses a two branch neural network with a coupled cluster loss function to create a feature representation of vehicle images where distance can be directly used to measure the similarity of arbitrary two vehicles. Chu et al. [9] proposed a viewpoint-aware metric learning approach and associated neural network that classifies if the vehicle images have similar or different viewpoints and has a corresponding metric for each case. Zheng et al. [10] proposes a two-stage progressive training approach to learning more robust visual representations from vehicle datasets. However, none of the vehicle re-identification work consider data from vehicle views, i.e. cameras that are on the vehicles. The appearance of vehicle features from the views of other neighboring vehicles can vary greatly compared to those from surveillance camera views and thus it is unlikely the vehicle re-identification models will perform well in the context of vehicle views. Recently, there has been some work that considered vehicle matching from the views of vehicle cameras. Liu [11] proposed using a pre-trained neural network with the contrastive loss function to classify vehicle images as the same or not; the dataset used for this paper was recorded from the perspective of the vehicles but the problem of Non-Overlapping Vehicle Matching, where there are no overlapping visual features, is not addressed, which is the focus of this paper.

III. Methodology

In this section, we describe our approach of studying different classifiers and feature representations to see which best addresses the NOVeM problem. The general pipeline we use is shown in Figure 2. For each pair of input images, we want to create a feature set that can be used by a classifier to generate a similarity score for the image pair. As such, the two most important design choices to consider is what to use for the feature extractor and what to use for the classifier.

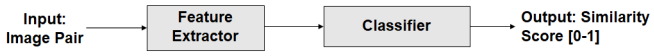

Fig. 2. Overview of vehicle matching approach: the system takes a pair of vehicle images as input and produces a similarity score between 0 and 1.

Note that in general, there can be more than two simultaneous vehicle camera views of the same road scene, and hence more than a single pair of detected objects (vehicles) that need to be matched. In this paper, we assume that such multi-view vehicle matching is performed as multiple pairwise matchings.

We are interested in using a machine learning model as our classifier, so another decision to be made is what dataset to train/test on. While other vehicle image datasets exist, there are none suitable for the NOVeM problem. KITTI [12] and Cityscapes [13] only include data from one vehicle; Vehicle re-identification datasets, such as VeRi [14] and VehicleID [8], only contain street surveillance footage of vehicles and have no images from multiple vehicle cameras. For our problem, we want a dataset that has camera views from multiple vehicles taken simultaneously to replicate the real time vehicle matching that would need to take place in a connected vehicle environment, which none of the previously mentioned datasets have available. As such, it is necessary to create a new dataset for the NOVeM problem.

A. Dataset

For data collection, two cameras were set up on opposite sides of an intersection on the UCSD campus in positions that mimic two cars stopped on opposite sides of the intersection; the recorded images of vehicles passing through the intersection have no overlap in their visual features. The dataset is an aggregate of recordings done over 4 separate afternoons. For the recorded data, object detection was applied to every video frame and all detected vehicles were cropped out; each cropped vehicle image from camera 1 is paired with every cropped vehicle image in camera 2 and binary labels are assigned according to if the vehicles are the same (1, positive sample) or not (0, negative sample). In total, 10,816 pairs were labeled with 7795 being negative labels and 3021 being positive labels. A 80%/20% random split was applied to form a training and a testing dataset; an additional 95%/5% random split was applied to the training set to generate a validation set. Additionally, we ensured that the ratio of positive to negative samples remained constant over these random splits to ensure that each subset is not biased more towards one class. Example image pairs can be seen in Figure 3; as shown in the figure, the vehicle image pairs contain two completely different views of the vehicle that have little or no overlap in their visual features.

B. Feature Extraction

In this section, we present our approach to feature extraction and the different kind of feature sets we use. Because this is an image comparison problem, a Siamese architecture [15] is used for feature extraction. With this design, we generate feature vectors for both inputs using the same feature extractor in parallel as shown in Figure 4. Instead of creating our own

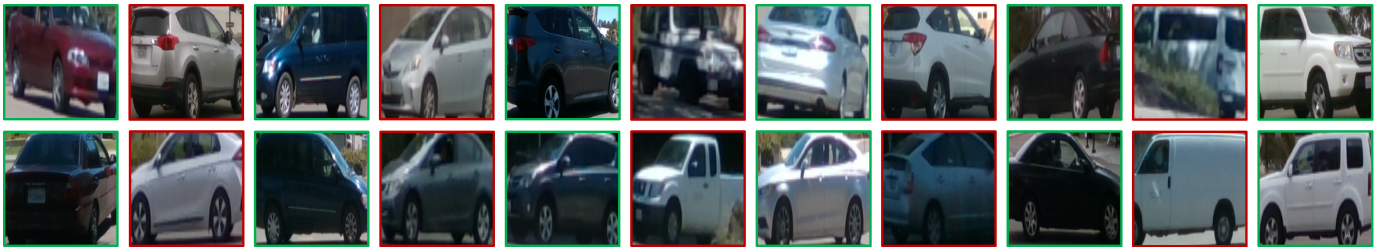

Fig. 3. Example image pairs from our dataset. Each column represents a vehicle image pair; the green boxes are matches and were assigned a positive label and the red boxes are non-matches and were assigned a negative label.

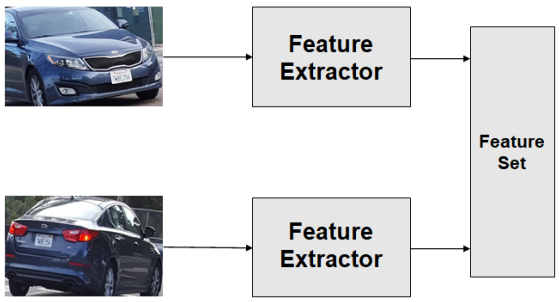

Fig. 4. A siamese architecture where the same feature extractor is applied to both input images in parallel is used to produce two feature vectors that can be combined/compared to form a feature set.

way to extract features from images, we want to use available methods that can produce features that are viewpoint invariant, i.e., the features will be the same for the most part no matter what viewpoint you are viewing the object from. To do this we will utilize transfer learning, where we use knowledge learned from one problem and apply it to a different problem. In this case, we will use feature extractors that have been trained and used for a different purpose, but whose features can be used for vehicle matching. Next we will discuss two possible feature sets that can be used as viewpoint invariant features.

High Level Visual Features: For the first set of features, we create a vector of high level visual features that are inspired by human intuition for vehicle matching. When we as humans look at pictures of two vehicles, we might consider things like the vehicles' colors, makes, and models to figure out if the two images are of the same vehicle. There exist information retrieval methods to generate these high level visual features from image data that can be used as a feature extractor for our problem; one such method is the Sighthound vehicle recognition API [16] which can provide a classification of a vehicle type, color, make and model with confidence scores for many of these features. With this data, a feature vector for every vehicle can be created; pairs of feature vectors are compared element-wise and merged into a combined feature vector as shown in Figure 5. We wanted to include both a field to check if the type, color, make, and model matched but also an average confidence score so that the system may be able to learn to distrust the comparisons with low confidence. These combined feature vectors serve as the input data for the classifier and we have labeled them High Level Visual Features (HLVF).

Learned Features: The second set of features are features learned by a neural network on a very large image dataset. The HLVF is a small feature set of predetermined features so to contrast that we wanted to have a large set of learned features for the second feature set. We chose to use a ResNet-18 [17] model trained on the ImageNet [18] dataset. This feature set has been used by many others and has proven to be very successful for a wide variety of applications [19]. For the ResNet-18 model, we remove the fully connected (FC) layer and take the flattened output of the last convolutional layer as the feature set; this is because this network was trained for the ImageNet Large Scale Visual Recognition Challenge and this FC layer is making classifications specific to that problem. This feature set is relatively high dimensional, much more so than the HLVF (512 vs 8). Since there are two images, we get two features vectors from the DNN's which are concatenated to form the combined feature vector and we label this feature set as Learned Features (LF).

C. Classification

In this section, will discuss the two classifiers we chose and why they are a good fit for this problem. The feature extractors we use are pretrained with different datasets and since our dataset (Section III.A) is relatively small, we decide not to retrain the feature extractor networks. Instead, we use machine learning models as the classifier that can learn parameters specific to the NOVeM problem to make the classifications based on the generated feature sets.

For the first classifier, we decide to use a Random Forest (RF) model. The main reason for choosing a RF is due to the fact that it can work with almost any type of data. With RF, we can use the same model with the same parameters for both HLVF and LF to get an idea of which feature set is more discriminative for NOVeM. Compared to other decision tree based models, the RF has lower variance and generally a lower overall error due to its use of aggregate bootstrapping. In aggregate bootstrapping, random subsets of features are selected to form decision trees and predictions

Combined Feature Vector	=	Same type?	Same color?	Avg. color confidence	Same make?	Avg. make confidence	Same model?	Avg. model confidence

Fig. 5. HLVF combined feature vector is formed by doing an element-wise comparison of the two vehicle feature vectors. For type,color,make, and model comparison the result is a Boolean variable and the average confidence scores are real number between 0 and 1.

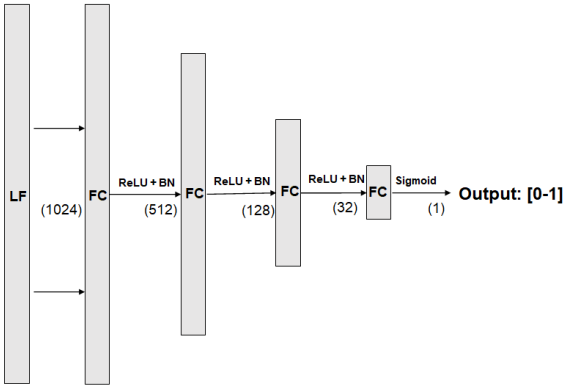

Fig. 6. NN architecture for classifying LF data

are made based off of the average prediction over all the decision trees. There are also very few parameters to tune, the two most important parameters are the number of estimators (trees) and the maximum number of features selected for each estimator; we used 100 estimators with number of features equal to square root of the total number of features.

The second classifier we use is a neural network (NN). Both feature extraction methods involve a deep convolutional NN extracting a feature set for us, so our NN design is just focused on classification. Given a set of input features, a NN classifier is a set of FC layers that map the input feature set to the desired output. We could use as little as 1 FC layer, but to give the model more parameters and allow it to learn more complex patterns in the data we decided to use 4 FC layers. The Rectified Linear Unit (ReLU) activation function and batch normalization are applied between each layer besides the output layer which has the sigmoid function applied to generate the similarity score. Figure 6 shows our NN architecture, including the number of FC layers and their different input and output sizes. We decided to only use the feature set LF for the neural network, since the dimension of the feature set HLVF is so low there would be very few parameters to learn. For the training process, we used a batch size of 256, learning rate of .001, the binary cross entropy loss function, and the Adam optimizer [20]. Random horizontal flip augmentation was added during training to increase the viewpoint diversity in the training set. The network itself was implemented in Python using the PyTorch library and was run on Nvidia GTX 1080 Ti GPU.

IV. Results

When examining the performance results of our model, one thing to note is the difference the effect false positives and false negatives can have on the overall collaborative perception. A false positive would be when two different objects are viewed by two cameras and the system identifies them as the same object. In this case, both detections could be deleted by the collaborative perception system using the results of the vehicle matching method, and a wrong "merged" detection could take their place resulting in unsafe perception information being relayed to the drivers/vehicles. A false negative would be when two cameras view the same object but our system failing to identify that they are the same. In this case, a duplicate detection will appear in the collaborative perception. As such, false negatives in vehicle matching should be considered less dangerous than false positives because while there may be multiple detections for the same object in the case of false negatives, the system will still have information about the object's location and the driver can be made aware of it.

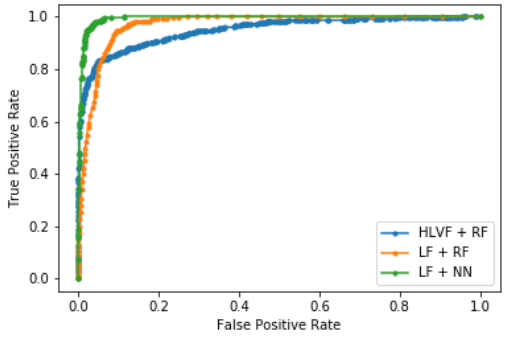

Fig. 7. ROC Curve of the proposed feature extraction and classification techniques.

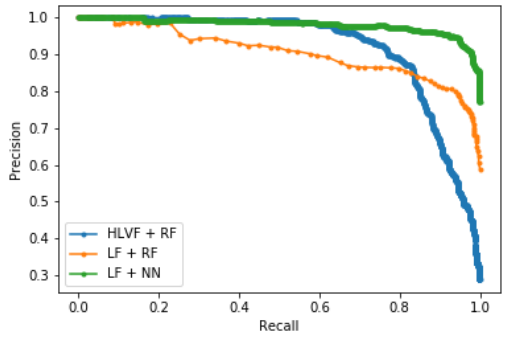

Fig. 8. PR Curve of the proposed feature extraction and classification techniques.

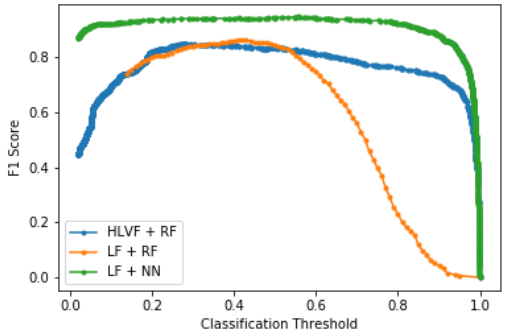

Fig. 9. F1 Score over different classification thresholds for proposed feature extraction and classification techniques.

Figures 7-9 give different interpretations of our model's performances on the test set and the results are summarized in Table 1. For each figure, we show the results for the

TABLE I
ACCURACY RESULTS

Metric	*ROC AUC*	*AP*	$\|F1\|_\infty$
LF + NN	.994	.980	.947
HLVF + RF	.949	.917	.849
LF + RF	.969	.906	.864

three different combinations of features sets and classifiers: HLVF + RF, LF + RF, and LF + NN. Figure 7 shows the Receiver Operating Characteristic (ROC) curve, showing how the different feature sets and classifiers perform in terms of true positive rate and false positive rate over different classification thresholds . All similarity scores greater than or equal the classification threshold are given positive labels and scores less than the classification threshold are given negative labels. As such, the choice of the classification threshold will directly affect the rate of false positives; how to choose an appropriate classification threshold will be discussed at the end of the section. The area under the ROC curve (ROC AUC) gives a general metric for how well the classifier is performing over all classification thresholds. A perfect classifier will have a ROC AUC equal to one and so we want our models to get as close to this value as possible; in this context the LF + NN model performs best at a ROC AUC of .994. While this curve and metrics from it are widely used for evaluating the performance of a binary classifier, it alone does not give a full picture of the system's performance. In our dataset, the number of negative examples outnumbers the number of positive examples by more than 2:1, so the false positive rate can be inflated due to the large number of true negatives. Since we consider reducing false positives to be very important to our problem, we will consider other metrics that give us a more accurate picture of the occurrence of false positives.

Figure 8 shows the Precision Recall (PR) curve, showing how our models perform in terms of precision and recall over different classification thresholds. The average precision (AP) metric represents the average precision value over all classification thresholds; the precision metric is inversely proportional to false positives, so the higher the AP the less false positives the system is experiencing on average. In the context of this metric, the LF + NN is the best with an AP of .980. However, we do not want to over bias the system to maximize AP; for example, if a model predicts a negative label for every image pair we would have zero false positives, but now the system would be overwhelmed with false negatives. As such, the metric of F1 score is calculated as well. The F1 score is the harmonic mean of precision and recall, so having a high F1 score is what allows the high values of AP to be meaningful since we can be assured that the system is not overly biased towards reducing false positives. A graph of each model's F1 score vs the classification threshold is shown in Figure 9. We use the infinity norm of the F1 scores as a way to compare the performance of the different models; in this context the LF + NN model is once again the best with a $\|F1\|_\infty$ equal to .947. Additionally, the LF + NN model achieves much more consistent performance over different thresholds shown by the relatively flat F1 curve around the maximum.

These metrics give an overall idea of the model performance, but for a real world implementation a classification threshold must be selected; Figure 10 shows the precision and recall values for different thresholds for the LF + NN model. Choosing a specific value of the threshold depends on how many false positives are allowed to pass through. Table 2 shows the specific precision and recall values for different classification thresholds. With the application of collaborative perception and automotive safety in mind, a very high precision would be required to minimize the false positives and a high classification threshold ($>$.8) may be chosen accordingly; however, for a different application where false positives are not viewed so harshly, one may want to choose a more balanced classification threshold ($\sim$.5) that maximizes the F1 score.

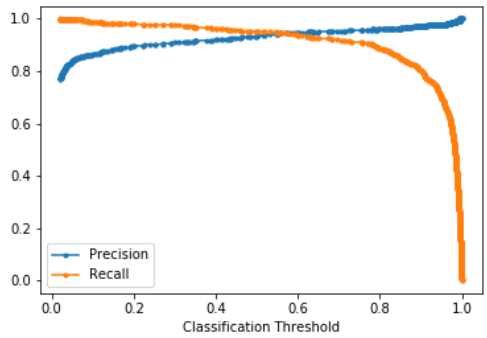

Fig. 10. Precision and Recall over different classification thresholds for the LF + NN model

TABLE II
COMPARISON OF DIFFERENT CLASSIFICATION THRESHOLDS

Classification Threshold	*Precision*	*Recall*	*F1 Score*
.5	.938	.953	.945
.6	.950	.939	.944
.7	.957	.924	.940
.8	.965	.896	.929
.9	.979	.834	.901

V. CONCLUSION

In this paper, we compared the performance of two different features sets and classification models and found that a deep neural network feature extractor trained on ImageNet provides a very good set of features for NOVeM. While the HLVF are more intuitive in vehicle matching for humans, the LF perform better overall when applied to a machine learning classifier. Additionally, neural networks achieve higher accuracy as compared to Random Forests when used as a classifier in this context. In the end, we have a system that is capable of achieving a very high level of precision with a good F1 score that shows the feasibility of vehicle matching even in scenarios with very large viewpoint differences. While the proposed methods were able to achieve varying degrees of success with the defined task, they all have low inference times ($<$ 10 ms for a single instance on a Nvidia GTX 1080 Ti GPU) which is a positive sign that real time systems are achievable.

VI. Future Work

Overall, the results are very promising and give us a baseline for future work in which we will consider more complex and broader vehicle matching scenarios, like matching vehicle camera and street camera views, and extending to cameras views from moving vehicles. Our goal is to create a general vehicle matching system that can handle any viewpoint situation, whether it be nearly identical or completely nonoverlapping, so that the candidate images can be matched regardless of whether the camera is on the vehicle or on a street camera. We plan to use the NOVeM dataset as the starting point for a much larger and all-encompassing dataset that will include these other situations of camera placements and mobility. We also plan to incorporate the physical aspects of the vehicles as well including the position, orientation, velocity, and 3D structure of the vehicles. To do this we will need to look temporarily across images through tracking as well as spatially using GPS, IMU, and depth sensors. By combining these physical aspects of the vehicles along with visual features, further improvements can be made to the vehicle matching system that will allow it to work in a more diverse set of scenarios.

Acknowledgments

This material is based upon work supported by the UC San Diego Center for Wireless Communications, and the Smart Transportation Innovation Program (STIP).

References

[1] NHTSA, "Traffic Safety Facts: 2016 Data," NHTSA's National Center for Statistics and Analysis, DOT HS 812 580, September 2018. [Online] Available: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812580.

[2] A. Bazzi, B. M. Masini, A. Zanella and I. Thibault, "On the Performance of IEEE 802.11p and LTE-V2V for the Cooperative Awareness of Connected Vehicles," in IEEE Transactions on Vehicular Technology, vol. 66, no. 11, pp. 10419-10432, Nov. 2017.

[3] G. Naik, B. Choudhury and J. Park, "IEEE 802.11bd 5G NR V2X: Evolution of Radio Access Technologies for V2X Communications," in IEEE Access, vol. 7, pp. 70169-70184, 2019.

[4] M. -P. Dubuisson and A. K. Jain, "A modified Hausdorff distance for object matching," Proceedings of 12th International Conference on Pattern Recognition, Jerusalem, Israel, 1994, pp. 566-568 vol.1.

[5] M. Cho, Jungmin Lee and K. M. Lee, "Feature correspondence and deformable object matching via agglomerative correspondence clustering," 2009 IEEE 12th International Conference on Computer Vision, Kyoto, 2009, pp. 1280-1287.

[6] E. Rublee, V. Rabaud, K. Konolige and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," 2011 International Conference on Computer Vision, Barcelona, 2011, pp. 2564-2571.

[7] Lowe, D. G., "Distinctive Image Features from Scale-Invariant Keypoints", International Journal of Computer Vision, 60, 2, pp. 91-110, 2004.

[8] H. Liu, Y. Tian, Y. Wang, L. Pang and T. Huang, "Deep Relative Distance Learning: Tell the Difference between Similar Vehicles," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 2167-2175.

[9] R. Chu, Y. Sun, Y. Li, Z. Liu, C. Zhang,and Y. Wei, "Vehicle re-identification with viewpoint-aware metric learning," IEEE International Conference on Computer Vision (ICCV), pp. 8282–8291, 2019.

[10] Z. Zheng, T. Ruan, Y. Wei, and Y. Yang, "Vehiclenet: learning robust feature representation for vehicle re-identification," CVPR Workshops, 2019.

[11] H. Liu, "Vehicle Verification Using Deep Learning for Connected Vehicle Sharing Systems," ACM MobiSys 2019 on Rising Stars Forum (RisingStarsForum'19), Association for Computing Machinery, New York, NY, USA, pp. 7–12, 2019.

[12] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," International Journal of Robotics Research (IJRR), pp. 1231-1237, 2013.

[13] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[14] X. Liu, W. Liu, H. Ma and H. Fu, "Large-scale vehicle re-identification in urban surveillance videos," 2016 IEEE International Conference on Multimedia and Expo (ICME), Seattle, WA, 2016, pp. 1-6.

[15] S. Chopra, R. Hadsell and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 2005, pp. 539-546 vol. 1.

[16] A. Dehghan, S. Z. Masood, G. Shu, and E. G. Ortiz, "View independent vehicle make, model and color recognition using convolutional neural network." [Online]. Available: https://arxiv.org/abs/170, 2017.

[17] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 770-778.

[18] J. Deng, W. Dong, R. Socher, L. Li, Kai Li and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database," 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, 2009, pp. 248-255.

[19] M. Huh, P. Agrawal, A. Efros, "What makes ImageNet good for transferlearning? ", NIPS workshop on LSCVS, 2016.

[20] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," International Conference for Learning Representations, 2015.