Learning to Tell Brake and Turn Signals in Videos Using CNN-LSTM Structure

Han-Kai Hsu[1], Yi-Hsuan Tsai[1], Xue Mei[2], Kuan-Hui Lee[2],
Naoki Nagasaka[2], Danil Prokhorov[3], Ming-Hsuan Yang[1]
[1]University of California, Merced [2]Toyota Research Institute [3]Toyota Research Institute - North America

Abstract— **We present a method that learns to tell rear signals from a number of frames using a deep learning framework. The proposed framework extracts spatial features with a convolution neural network (CNN), and then applies a long short term memory (LSTM) network to learn the long-term dependencies. The brake signal classifier is trained using RGB frames, while the turn signal is recognized via a two-step localization approach. The two separate classifiers are learned to recognize the static brake signals and the dynamic turn signals. As a result, our recognition system can recognize 8 different rear signals via the combined two classifiers in real-world traffic scenes. Experimental results show that our method is able to obtain more accurate predictions than using only the CNN to classify rear signals with time sequence inputs.**

I. Introduction

In recent years, autonomous driving has drawn significant attention, especially on the topic of safety. Human drivers communicate lane changes and turns via the car's rear signal lights; thus, it is important that self-driving vehicles are taught to comprehend what each signal is, and when each signal is used. This is paramount in ensuring the safety of the passenger in either autonomous or assistive driving systems.

Numerous methods have been proposed to recognize brake and turn signals in the past decade. Existing systems mainly use hand-crafted features such as color thresholds or luminance to detect and extract the light regions [5], [15], [18], [13], [4]. As these manually defined features are variant due to different lighting conditions or clutter, existing methods based on such visual cues do not perform well in real-world scenes.

In this work, we identify rear light signals through learning deep features using CNN and LSTM networks. CNNs have been widely used in vision tasks and provide effective representations on learning the spatial features of images. On the other hand, the LSTM networks have been shown to be capable of handling the long-term dependencies in a sequence. Existing CNN-based methods for classification of brake signals [20] operate on the premise that brake signals are static and can be classified using only spatial features. However, turn signals are dynamic and cannot be effectively recognized with one single frame, especially when the signal flashes on and off during that frame. Thus, we combine a LSTM network with a CNN to learn the temporal information of a sequence. Additionally, we develop a two-step localization approach to extract visual cues from the turn signals. First, we use the SIFT flow [14] of two continuous frames and warp the latter one based on the flow to align two frames. Then we compute the absolute difference between the warped image and the first frame to obtain the frame differences. Moreover, we focus on the tail light regions using region of interest (ROI) to crop out the input and send to the network as the additional guidance. As a result, our method is able to achieve higher classification accuracy than using RGB frames as the input.

Our training process uses a data set collected under real-world traffic conditions during the daytime. We define a total of 8 different rear signal states and label the data accordingly for supervised learning. Afterwards, we train classifiers for brake and turn signals individually and then integrate them into our rear signal recognition system. As a result, our proposed system can effectively identify 8 rear signal states, which outperforms the CNN only approach.

II. Related Work

For intelligent vehicles, it is imperative to distinguish between brake and turn signals, and learn when each is used. Numerous brake recognition methods [5], [15], [18] use thresholds on color features in order to detect the states of brake signals. Learning approaches have also been applied to learn feature representations for brake signals. Zhong et al. [22] train a FCN network [16] to identify the light regions and extract features within the region for classification using a linear support vector machine (SVM) classifier. In addition, deep convolution neural networks have been used to tell the state of brake signals after vehicle detection [20].

Turn signal recognition methods based on color thresholds have been developed [13], [4]. In these methods, turn signals are recognized by separating frames into left and right halves to determine the on and off states based on the number of pixels in the frame preserved by thresholds [13]. On the other hand, Chen et al. [4] train a classifier using the AdaBoost algorithm to determine the existence of turn lights, and then use reflectance contrast to tell the directions. Alternatively, the method [10] locates the turn lights by using the Kanade-Lucas-Tomasi feature tracker [19] to compute the feature correspondence between two successive frames. Afterwards, the RANSAC algorithm [9] is used to receive the transformation matrix that maps the correspondence between the two frames. The latter frame is then transformed and subtracted from the first frame to get the absolute difference for locating the turn signal. Features are then extracted from

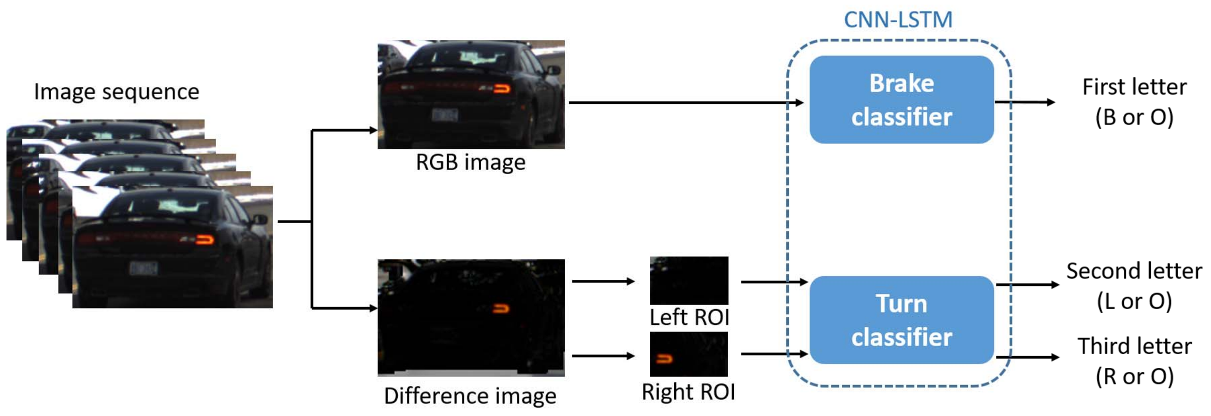


Fig. 1. The brake and turn signal classifiers utilized in this system are based on a CNN-LSTM structure. The proposed system integrates the two classifiers with a 16-frame sequence as the input and outputs the rear light labels represented by three letters: B (brake), L (left) and R (right). We receive O (off) when the corresponding signal is turned off. RGB frames are passed to the brake classifier to obtain the first letter of the sequence state. Then, the computed difference image of the sequence is cropped into two ROI regions as input for the turn classifier and outputs the last two letters of the sequence state.

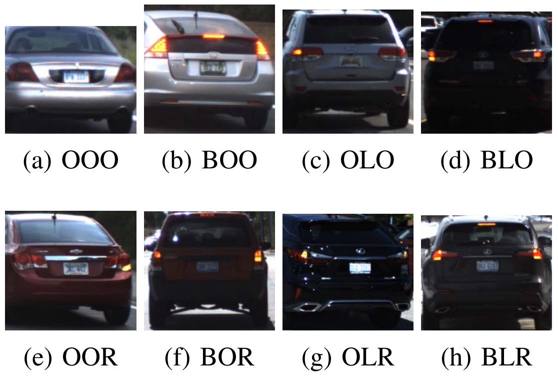


Fig. 2. Examples of rear signal states.

the located regions and transformed to the frequency domain, in which an AdaBoost classifier is trained to identify the turn lights.

Many methods have been proposed for recognizing both brake and turn signals. Chen et al. [6] determine the lighted pixels by defining a response function and apply it to every pixel to locate the rear lights. A high-pass mask is used to find the illuminated region with brighter inner pixels for determining the states accordingly. Kalman filters and codebooks are used in [1], [2], [3] to track rear lights. The states are classified based on the luminance channel observed over time on the two detected light regions. Recently, a method that classifies rear lights into four states with an SVM [7] is developed where the turn lights are detected based on thresholds.

III. Proposed Algorithm

In contrast to existing methods that are mainly based on hand-crafted features, we propose an algorithm to learn feature representations from the training data. Instead of only learning the spatial features through a CNN [20] to recognize brake lights, we also aim to recognize turn signals. Turn signals preserve different characteristics compared to brake signals. While flashing, it is difficult to recognize from one frame whether the turn signal is on or not at that time step. Thus, we use a LSTM module with a CNN to facilitate learning the temporal features of turn signals. We use a similar network as LRCN [8] which is developed for action recognition. This allows us to learn the different actions of rear signal states throughout a sequence. Our network takes a time-sequence as the input and determines the label accordingly. The CNN is used to extract spatial features from an input frame sequence, and the following LSTM module receives the output features from the CNN and learns the temporal information accordingly. Finally, we use the learned classifiers in our recognition system to determine the state of an input sequence in the real-world scenes. In the following section, we first describe our recognition system including the brake and turn signal classifiers, and then proceed with the details of our CNN-LSTM framework used to train the classifiers. Fig. 1 shows the overview of the proposed algorithm.

A. Rear Signal Recognition System

Our recognition system has two main components including the static brake classification and the dynamic turn signal classification. As such, we can focus more on the specific characteristic of each task and achieve better performance. Toward this end, we learn two separate classifiers for brake and turn signals, then integrate them into our recognition system.

We define a total of 8 distinct states based on all combinations of brake and turn lights. Each state is denoted by 3 letters of B (brake), L (left), and R (right). We give either the corresponding letter of the signal when it is on, or a letter O for off. Consequently, there will be 8 different states as shown in Fig. 2. Accordingly, we annotate all frames in the training and testing sets using the state definition.

Brake classifier. RGB frames are used as input to the network for learning the brake classifier. We extract the spatial features through the CNN (see Section III-B), and then exploit additional temporal features in a sequence.

Turn signal classifier. When training the turn signal classifier, we notice that extracting spatial and temporal information from RGB images is not sufficient for the system

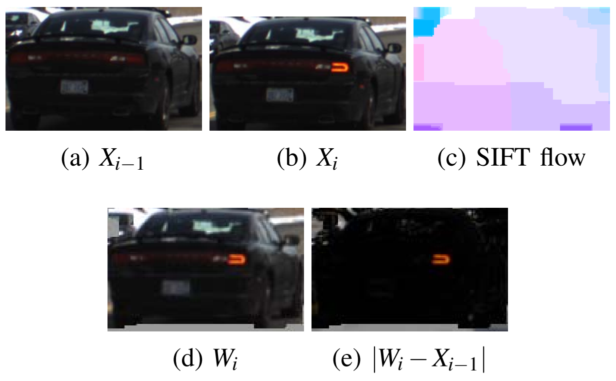


Fig. 3. Computing the difference between two consecutive frames. (a) Frame from previous step X_{i-1}. (b) Current frame X_i. (c) The computed SIFT flow [14] between X_{i-1} and X_i. (d) W_i is the warped image of X_i using SIFT flow in (c). (e) Absolute difference of W_i and X_{i-1}.

to recognize turn signals. We observe that there are different ways for vehicles to signal a turn. One type uses an individual light as the turn signal, and the other type utilizes the flashing brake lights when the turn signal is on. This makes it more difficult to distinguish between the brake and turn lights. In addition, the flashing turn signal occupies a relatively small area of the whole rear region and it is hard to extract a sufficient amount of visual cues from the turn lights. To resolve these problems, we propose a two-step turn signal localization method by first replacing RGB frames to the difference image of subsequent frames aligned using the SIFT flow [14]. As a result, there will be no differences on states besides turning and we can emphasize turn signals without being affected by other information like brake lights. In addition, we use ROI to exploit more information from small regions of turn lights in the frames.

Two-step turn signal localization. The first step is to find the frame difference. A difference image contrasts the change in turn signal while other parts of the image are subtracted and have no differences. However, we may obtain noisy results by directly computing the absolute difference because the vehicles in the two frames are not aligned well. To obtain better difference images, we use the SIFT flow algorithm [14] to help align the vehicles in successive frames. It has been shown that images in the same scene can be effectively aligned using the SIFT flow [14]. As the input sequences considered in this work are from the same scene, the SIFT flow is likely to help obtain better difference images. We take two adjacent frames in the video sequence X_{i-1} and X_i, and compute the SIFT flow. Next, we warp X_i to W_i based on their SIFT flow to align two images. We then obtain the difference image by computing $|W_i - X_{i-1}|$. An example of a difference image based on the above operations is shown in Fig. 3(e).

In the next step, we use ROI to focus on the rear light regions for better feature extraction. Our ROI is represented as the blue highlighted region in Fig. 4. During the learning process, we take the two regions of our ROI from the computed difference image as separate inputs to our network. Consequently, the network learns to distinguish whether that

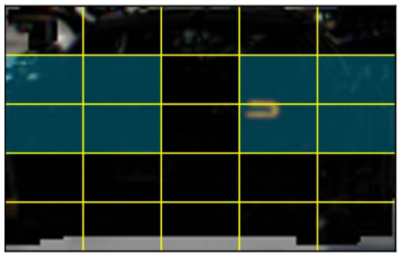


Fig. 4. ROI is indicated as the blue highlighted area, where the left and right blue regions are propagated through the network separately.

ROI region is flashing or not.

Our recognition system utilizes two classifiers to successfully recognize 8 different signal states. First of all, we pass the RGB frames directly to the brake classifier and determine the state of the brake signal. Second, we compute difference images from the given sequence and extract the left and right ROIs as two inputs. These two light regions then propagate through our turn classifier to determine the states of the left and right signals respectively. The states (represented by three letters) from both classifiers form the prediction for an image sequence.

B. Convolution Neural Network (CNN)

The CNN is able to learn the spatial features of images and has been successfully applied to many different image classification challenges. Therefore, we apply the CNN to learn the underlying features of our input. The CNN serves to exploit the spatial features within every frame in order to further learn the long-term dependencies in the sequences.

Our network adopts the CNN-LSTM structure as shown in Fig. 5. The CNN network for the spatial features extraction is a variant based on CaffeNet [12], which is also utilized in [8]. There are nine layers in total, including five convolutional layers, three pooling layers and a fully-connected layer. We utilize this CNN to learn the tail light features of the vehicles for localization and classification. After obtaining these spatial features, we feed the information of each frame to the LSTM module in order to learn the temporal information in each image sequence.

C. Long Short-Term Memory (LSTM)

The LSTM model, derived from the the recurrent neural network (RNN), has been used in various tasks (e.g. speech recognition, action recognition) to learn the temporal information of the sequence inputs. The advantage of the LSTM over the RNN is that it effectively alleviate the vanishing gradient problem in the RNN. The LSTM unit incorporates several non-linear activation gates to determine whether to retain or discard the information. There are numerous variants since the first LSTM model [11]. In this work, we use the LSTM unit proposed in [21].

The LSTM module is used in our network to maintain the long-term dependencies of our input sequences. Since obtaining the spatial features from the CNN is not sufficient to classify turn signals, it is essential for the classifier to

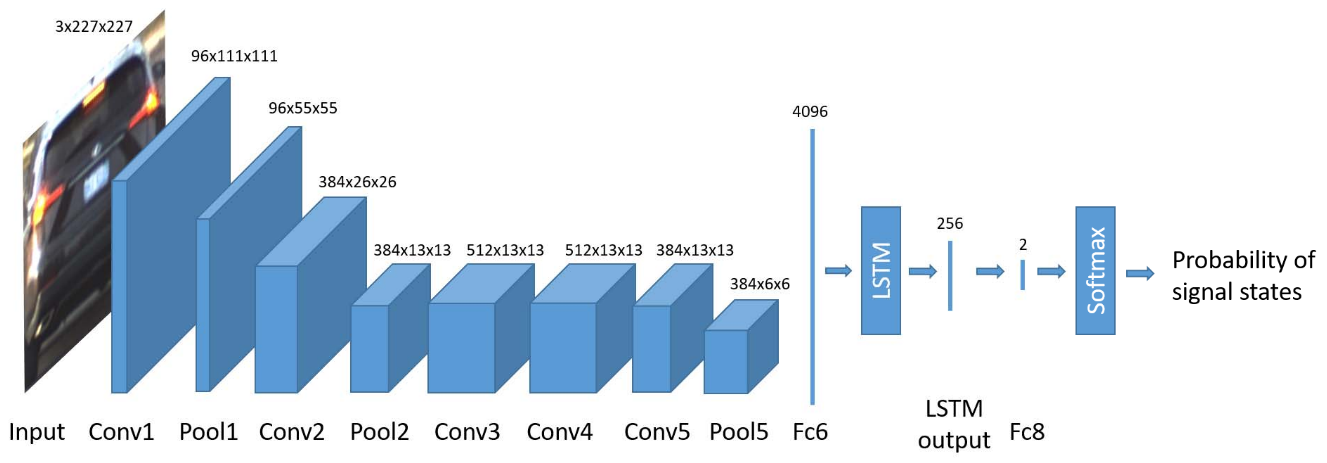


Fig. 5. Illustration of our network architecture. The layers before fc6 serve to extract spatial information from our input. Afterwards, we send the features to the LSTM module to utilize the temporal information and further classify the desired state in the last fully-connected layer.

learn the temporal information throughout the sequences to determine the dynamic flashing states.

The structure of the LSTM model in this work is shown in Fig. 6. One input is the feature x_t from the fc6 layer of the CNN described in Section III-B at time step t, and another input is the hidden unit h_{t-1} from the last time step. At every time step, the LSTM unit estimates the hidden unit h_t and sends it to the LSTM unit at the next time step. The LSTM unit also receives a memory cell c_{t-1} which holds the information from previous time steps. The memory cell is updated at every time step and then passed to the next LSTM unit. All the updates and outputs are processed by the different gates of the LSTM unit. First, the forget gate f_t determines what to discard from x_t and h_{t-1}. A forget gate is a sigmoid function (σ) that outputs values from 0 to 1 and performs element-wise product ($\odot$) with previous memory cell state c_{t-1} to determine what information to forget or remember. Next, the LSTM unit updates information to the memory cell through the input gate i_t and the hyperbolic tangent (*tanh*) layer g_t. These two gates, i_t and g_t, control what information to remember from x_t and h_{t-1} then add them to the memory cell. The activation gates are computed by

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \tag{1}$$
$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \tag{2}$$
$$g_t = tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{3}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \tag{4}$$

where W denotes the weight connecting the inputs and the given gate, and b is the corresponding bias term.

Aside from updating the memory cell, the LSTM unit also outputs a hidden state h_t at every time step. The output gate o_t is computed and weighted with the cell states through a *tanh* layer to determine the hidden state h_t by

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \tag{5}$$
$$h_t = o_t \odot tanh(c_t) \tag{6}$$

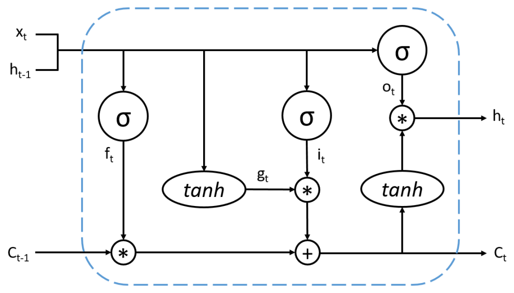


Fig. 6. The LSTM unit in the proposed algorithm.

The output of each LSTM layer is sent to the last fully connected layer of our network to compute a class probability for each time step. In order to take the temporal dependence of an input sequence into account, we focus on the output prediction of the last frame, which contains sufficient information from all the previous frames. That is, we compute the loss of the last frame and backpropagate all frames in the sequence with the same loss. Given a test frame, instead of taking average among the output predictions, we take the prediction of the last frame as the label for the entire input sequence.

IV. Experimental results

In this work, we collect a data set of 649 videos including 63,637 frames. The sequences are recorded during the daytime under real-world driving conditions with various vehicle types. We crop out the image regions of car rears in each video and label them based on our definitions described in Section III-A. It is difficult to collect videos for emergency lights (OLR and BLR), since such cases are rarely observed in daily driving conditions. However, our network is able to handle this problem without having many samples of these two particular cases. The proposed algorithm separates each image into right and left signal light regions. With sufficient amount of images in all the other turn signal states (OOR, BOR, OLO and BLO), we are able to tell whether the turn signals are on for both regions. Table I summarizes the

TABLE I

DATA SET FOR TURN AND BRAKE LIGHTS.

	OOO	BOO	OLO	BLO	OOR	BOR	OLR	BLR	Total
Videos	188	211	78	63	58	33	9	9	649
Frames	21867	17874	6271	6380	4728	3527	1600	1390	63637

TABLE II

BRAKE RECOGNITION ACCURACY (%) COMPARISON WITH USING THE CNN AND OUR METHOD (CNN-LSTM). THE RESULTS ARE BASED ON FIVE-FOLD CROSS VALIDATION.

	Brake ON	Brake OFF
CNN	94.52 ± 0.19	95.60 ± 0.03
Ours (CNN-LSTM)	94.80 ± 0.21	96.22 ± 0.03

properties of the collected dataset.

A. Data pre-processing

Before training the network, we process the raw data such that we can have a sufficient amount of training samples for training the proposed CNN-LSTM network. In this work, we augment the raw data with random cropping, horizontal flipping, gamma correction, rotation, and shifting such that we increase the training samples by 300 folds.

For turn recognition, we compute the difference image of all the video frames. Additionally, we crop the ROI on the difference image, which is $2/5$ height and $2/5$ width of the right and left regions as shown in Fig. 4. Both regions are fed to the network as inputs and the training data is increased by two folds. We evaluate the proposed algorithm by five-fold cross validation.

B. Network settings

The proposed CNN-LSTM model is implemented using the Caffe toolbox [12]. We fine-tune the pre-trained model [8], which is developed based on the UCF101 data set [17]. The input of our network contains 10 batches each time, and each one contains 16 frames. A sequence of 16 frames allows us to obtain at least one cycle of the turn signal. Each frame is given the ground truth label based on the video sequence they belong to and resized to 227×227 pixels before feeding to the network. During the error backpropagation process, we take the ground truth and the output prediction of the last frame to compute the loss for every frame in the same sequence.

As we have an unbalanced data set for the brake and turn signals as shown in Table I, we randomly select the same amount of videos for the on and off of brake or turn signals in our 10 batches each time. As a result, we have sufficient amount of training samples for all classes.

C. Performance evaluation

For each test video, we select every unique sample of 16 frames sequence and feed it to the trained model (e.g., 2 sequences for a video of 17 frames). We compute the accuracy based on the percentage of correctly predicted sequences in each video and use the average as the metric for performance evaluation.

TABLE III

TURN RECOGNITION COMPONENT ANALYSIS OF THE PROPOSED ALGORITHM WITH DIFFERENT COMBINATIONS OF FEATURES AND MODULES. RGB/DIFFERENCE IMAGE (DIFF), ROI, AND LSTM ARE ADDED FOR EVALUATION BASED ON PREDICTION ACCURACY (%).

	ROI	LSTM	No turn	Left turn	Right turn	Emergency
RGB		✓	88.90	59.82	43.63	0.00
Diff		✓	88.45	89.79	64.10	100.00
Diff	✓		90.85	80.50	64.47	100.00
Diff	✓	✓	97.94	94.57	87.72	100.00

TABLE IV

TURN RECOGNITION COMPARISON WITH USING THE CNN AND OUR METHOD (CNN-LSTM). BOTH METHODS USE DIFFERENCE IMAGE AND ROI FOR INPUTS, AND EVALUATED USING FIVE-FOLD CROSS VALIDATION.

	No turn	Left turn	Right turn	Emergency
CNN	91.32 ± 0.17	66.42 ± 0.82	64.46 ± 1.19	91.61 ± 1.17
Ours	95.92 ± 0.09	91.12 ± 0.23	93.46 ± 0.22	99.16 ± 0.04

Table II shows the brake recognition results using the proposed algorithm and the CNN only method, which is implemented based on the AlexNet in the Caffe toolbox. The CNN method is evaluated using the average accuracy from the predictions of all frames in each video. The results show that although brake lights are static and depend less on temporal features, our method is able to achieve better accuracy with additional temporal information.

We evaluate the turn signal classification results to analyze how the proposed model performs. Table III shows the experimental results with different features and modules including RGB/difference image, ROI, and LSTM. Experimental results show that we can increase the recognition accuracy of turn signals using all the modules and features in the proposed algorithm. In Table IV, we show the results using five-fold cross validation on the proposed algorithm in comparison to the method only using the CNN. Both networks use difference image and ROI as inputs for classification. The results show the effectiveness of the proposed algorithm using the LSTM model.

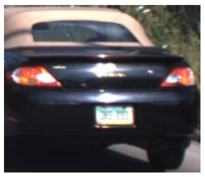
Table V shows the five-fold cross validation results for 8 turn signal recognition using the proposed algorithm. Overall, the proposed algorithm is able to tell the turn light signals effectively.

In the experiments, we observe some failure cases as shown in Fig. 7. For brakes, there are cases that are difficult to recognize due to the sunlight. For example, Fig. 7(a) is labeled as OOO but it looks like the brake signal is turned on. On the other hand, Fig. 7(b) is labeled as BOO but it is difficult for humans to tell whether it is the color of the light mask or it is actually turned on. Turn signals have similar issues with additional factors such as shades. In Fig. 7(c), tail lights of the middle two frames appear to be on due to the sunlight, and the lights in the other frames have cast shadows. As a result, the image differences between the first two frames (and last two frames) appear to be similar to flashing signals. In such cases, the proposed CNN-LSTM model does not perform well.

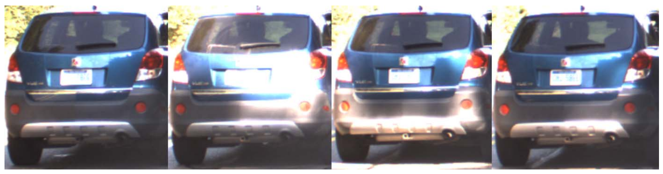
TABLE V

FINAL RECOGNITION ACCURACY (%) EVALUATED USING FIVE-FOLD CROSS VALIDATION.

	OOO	BOO	OLO	BLO	OOR	BOR	OLR	BLR
Accuracy	93.52 ± 0.25	90.30 ± 0.38	86.12 ± 0.89	88.02 ± 0.44	94.72 ± 0.20	84.00 ± 1.23	97.90 ± 0.18	100.00



(a) Brake failure (OOO) (b) Brake failure (BOO)



(c) 4 continuous frames of the turn failure case (OOO)

Fig. 7. Brake and turn failure cases. (a) and (b) are brake recognition failures affected by strong sunlight and is difficult for human to recognize the signal state. (c) is a turn failure case caused by sunlight and shades which in result acts similar to flashing signals.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we present a method that learns to recognize 8 different signal states. Our method utilizes a deep network to learn the temporal information through a LSTM network in addition to exploiting the spatial features from the CNN. We train the brake signal classifier using RGB frames and use a two-step localization method to learn the dynamic patterns of the turn signals. First, we compute the difference between two adjacent frames. Second, we segment out the ROI for the rear light regions in the image as our input to the network. The two classifiers are learned separately and is then integrated to our recognition system. We show that our proposed system is able to effectively predict each signal states in the real-world traffic scenes. Instead of training two separate classifiers, our future work will focus on learning to tell the brake and turn signals altogether in an end-to-end network.

REFERENCES

[1] A. Almagambetov, M. Casares, and S. Velipasalar. Autonomous tracking of vehicle rear lights and detection of brakes and turn signals. In *IEEE Symposium on Computational Intelligence for Security and Defense Applications*, pages 1–7, 2012.

[2] A. Almagambetov, S. Velipasalar, and M. Casares. Robust and computationally lightweight autonomous tracking of vehicle taillights and signal detection by embedded smart cameras. *IEEE Transactions on Industrial Electronics*, 62(6):3732–3741, 2015.

[3] M. Casares, A. Almagambetov, and S. Velipasalar. A robust algorithm for the detection of vehicle turn signals and brake lights. In *IEEE Ninth International Conference on Advanced Video and Signal-Based Surveillance*, pages 386–391, 2012.

[4] D.-Y. Chen, Y.-J. Peng, L.-C. Chen, and J.-W. Hsieh. Nighttime turn signal detection by scatter modeling and reflectance-based direction recognition. *IEEE Sensors Journal*, 14(7):2317–2326, 2014.

[5] H.-T. Chen, Y.-C. Wu, and C.-C. Hsu. Daytime preceding vehicle brake light detection using monocular vision. *IEEE Sensors Journal*, 16(1):120–131, 2016.

[6] L.-C. Chen, J.-W. Hsieh, S.-C. Cheng, and Z.-R. Yang. Robust rear light status recognition using symmetrical surfs. In *IEEE International Conference on Intelligent Transportation Systems*, pages 2053–2058, 2015.

[7] Z. Cui, S.-W. Yang, and H.-M. Tsai. A vision-based hierarchical framework for autonomous front-vehicle taillights detection and signal recognition. In *IEEE International Conference on Intelligent Transportation Systems*, pages 931–937, 2015.

[8] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2625–2634, 2015.

[9] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[10] B. Frohlich, M. Enzweiler, and U. Franke. Will this car change the lane?-turn signal recognition in the frequency domain. In *IEEE Intelligent Vehicles Symposium*, pages 37–42, 2014.

[11] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[12] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM international conference on Multimedia*, pages 675–678, 2014.

[13] Y. Li, Z.-x. Cai, and J. Tang. Recognition algorithm for turn light of front vehicle. *Journal of Central South University*, 19(2):522–526, 2012.

[14] C. Liu, J. Yuen, and A. Torralba. Sift flow: Dense correspondence across scenes and its applications. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):978–994, 2011.

[15] W. Liu, H. Bao, J. Zhang, and C. Xu. Vision-based method for forward vehicle brake lights recognition. *International Journal of Signal Processing, Image Processing and Pattern Recognition*, 8(6):167–180, 2015.

[16] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[17] K. Soomro, A. R. Zamir, and M. Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[18] P. Thammakaroon and P. Tangamchit. Predictive brake warning at night using taillight characteristic. In *IEEE International Symposium on Industrial Electronics*, pages 217–221, 2009.

[19] C. Tomasi and T. K. Detection. Tracking of point features. Technical report, Tech. Rep. CMU-CS-91-132, Carnegie Mellon University, 1991.

[20] J.-G. Wang, L. Zhou, Y. Pan, S. Lee, Z. Song, B. S. Han, and V. B. Saputra. Appearance-based brake-lights recognition using deep learning and vehicle detection. In *IEEE Intelligent Vehicles Symposium*, pages 815–820, 2016.

[21] W. Zaremba and I. Sutskever. Learning to execute. *arXiv preprint arXiv:1410.4615*, 2014.

[22] G. Zhong, Y.-H. Tsai, Y.-T. Chen, X. Mei, D. Prokhorov, M. James, and M.-H. Yang. Learning to tell brake lights with convolutional features. In *IEEE International Conference on Intelligent Transportation Systems*, pages 1558–1563, 2016.