Deep Neural Network for Structural Prediction and Lane Detection in Traffic Scene

Jun Li, Xue Mei, *Senior Member, IEEE*, Danil Prokhorov, *Senior Member, IEEE*, and Dacheng Tao, *Fellow, IEEE*

***Abstract*— Hierarchical neural networks have been shown to be effective in learning representative image features and recognizing object classes. However, most existing networks combine the low/middle level cues for classification without accounting for any spatial structures. For applications such as understanding a scene, how the visual cues are spatially distributed in an image becomes essential for successful analysis. This paper extends the framework of deep neural networks by accounting for the structural cues in the visual signals. In particular, two kinds of neural networks have been proposed. First, we develop a multitask deep convolutional network, which simultaneously detects the presence of the target and the geometric attributes (location and orientation) of the target with respect to the region of interest. Second, a recurrent neuron layer is adopted for structured visual detection. The recurrent neurons can deal with the spatial distribution of visible cues belonging to an object whose shape or structure is difficult to explicitly define. Both the networks are demonstrated by the practical task of detecting lane boundaries in traffic scenes. The multitask convolutional neural network provides auxiliary geometric information to help the subsequent modeling of the given lane structures. The recurrent neural network automatically detects lane boundaries, including those areas containing no marks, without any explicit prior knowledge or secondary modeling.**

***Index Terms*— Image recognition, pattern analysis, recurrent neural networks.**

I. Introduction

DEEP neural networks are powerful tools for visual analytics [1] and have shown superior performance in various tasks [2]. Compared with the traditional models of shallow computational structures, one essential advantage of deep nets is that the data representations are constructed in the learning process automatically. Therefore, deep neural networks are often considered to be capable of end-to-end learning, emphasizing that manual feature construction is replaced by automatic representation learning. However, automatic data representation deals with the input end of processing. For the output end, most existing networks assume simplified output representation, such as one or a few variables standing for a binary or 1-in-N class labels.

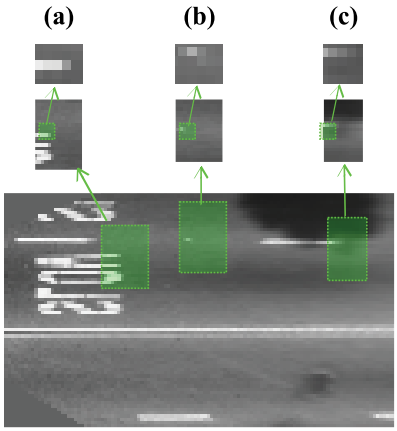


Fig. 1. Where are the lane boundaries? The importance of context when recognizing objects in images is demonstrated. (a)–(c) 7×7 image patches (scaled up for clear viewing). Middle: 30×20 surrounding areas of the small patches. Green rectangles: areas in the image from which the patches are taken. The area (a) contains paintings on road, not directly indicating traffic lanes. The area (b) is on a boundary between two lanes, without obvious marks. The area (c) is on the same boundary as (b) and contains a lane mark.

In practical vision tasks, however, extra processing steps are often needed to transform the simple outputs into the actual learning targets. For example, when an object is to be localized within an image (as opposed to merely perceiving its presence), a detector may employ a sliding window scheme to apply a neural network to examine every location within the image. This approach is effective when the outputs of the network can be independently evaluated for problem [3], [4]. In the above example, when the target is a monolithic object detectable from individual image patches, the scheme works well. However, in more challenging scenarios, such as detecting objects of varying sizes and shapes, making detections from the simple outputs requires sophisticated structuring and is nontrivial [5].

In this paper, we propose that for visual detection, deep-learning neural networks should not only automate input representation but also adopt task specific structures in the outputs. In particular, we consider the problem of detecting lanes in a traffic scene, which is important for driving automation. Two types of deep neural networks have been developed for recognizing the boundaries of the lanes from images. Before introducing our networks, it is helpful to inspect a typical example of lane detection. Fig. 1 shows a lane with a solid boundary and a broken boundary on each side. Since lanes and their boundaries usually span across large areas (the entire image in Fig. 1), the localization operates at least

Manuscript received April 30, 2015; accepted January 7, 2016. This work was supported in part by the Toyota Research Institute Collaborative Project, North America, and in part by the Australian Research Council under Grant FT-130101457.

J. Li and D. Tao are with the Centre for Quantum Computation Intelligent Systems, Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW 2007, Australia (e-mail: jun.li@uts.edu.au; dacheng.tao@uts.edu.au).

X. Mei and D. Prokhorov are with the Toyota Research Institute, Ann Arbor, MI 48105 USA (e-mail: nathanmei@gmail.com; dvprokhorov@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2016.2522428

two levels: locally, the lane model recognizes the patterns that mark a boundary with reasonable granularity (e.g., for every 7×7 image patch); globally, the model needs to revise the recognition with respect to related areas, including both the surrounding image context and structurally connected regions. Fig. 1(c) is recognized as boundary because of the visual evidence, while Fig. 1(b) is the boundary induced by the overall boundary structure. Fig. 1(a) is considered not to be a boundary mark due to its surrounding image context.

The two proposed networks are tailored to perform structured predictions to meet the need for lane detection. The first network is a deep convolutional neural network (CNN) that simultaneously performs multiple (two different) tasks. The output of the neural network consists of both classifier and regressor: the existence of visual cues (lane marks) is detected by the classifier, and if the detection returns a positive result, the regressor estimates the orientation and location of the lane mark within the region of interest (ROI). Compared with the traditional cue detectors based on binary classification, the proposed network can afford to work with relatively large ROIs, because the prediction about the target is further refined by the regressor. Large regions contain richer contextual information helping improve the detection accuracy.

The second network introduces a layer of recurrent neurons on top of the CNN. The resultant network has memory and is able to account for structures in the data. This is particularly useful to identify global targets (ones that persist over a sequence of local image areas) from local cues without explicitly specifying the structural knowledge of the global target. From a small number of labeled images, the network can be trained with a large set of augmented samples. In the test stage, the network predicts the boundaries of lanes from images. The detection framework is a complete end-to-end learning scheme. The proposed network is applied to the real-life traffic scene analysis showing promising results.

It is also worth noting that this paper is focused on the low-level detection, i.e., for each image location, determine whether it belongs to a lane boundary. The low-level detection can be integrated within any higher level models of traffic lanes, such as those in [6]–[8].

To summarize our contribution, for the problem of lane detecton from video, we develop a multitask deep convolutional network, which simultaneously detects the presence of the target and its geometric attributes (location and orientation) with respect to the ROI. In addition, for the same problem, we also develop a recurrent neural network (RNN) that uses its internal status memory to infer the presence or absence of a lane over a sequence of image areas. Fig. 2 shows the overall workflow of the proposed neural network systems.

This paper is organized as follows. We first review relevant research background in Section II. Sections III and IV introduce the multitask CNN and the adoption of recurrent neurons, respectively. Section V reports the empirical evaluation of the proposed networks. Section VI concludes this paper.

II. Background

Deep neural networks have witnessed significant progress in the past decade, since the increased computational capacity has led to breakthrough in effective learning schemes [9]–[12]. The deep computational structure has been proved useful in a wide range of application areas, including speech recognition [13], [14], natural language processing [15], and, particularly relevant to this paper, visual analytics [16], where the neurons are organized and connected to the input in a way that reproduces the convolution operation [2], [17]–[20]. For a comprehensive overview of the field, refer to [21], [22].

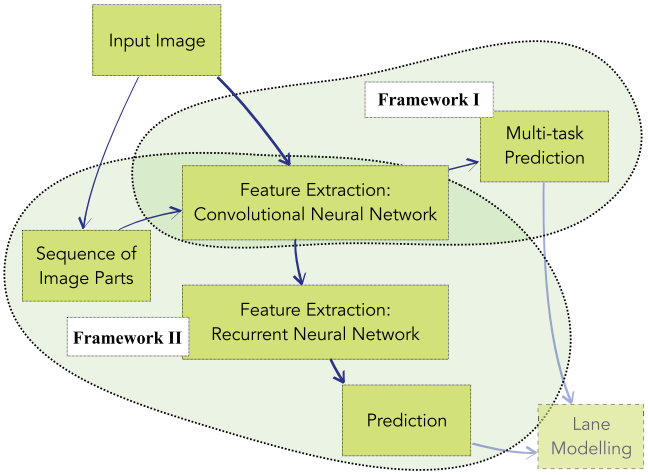


Fig. 2. Diagram of lane recognition using deep neural networks. The workflow of the two frameworks for lane recognition proposed in this paper. Both frameworks employ CNNs for feature extraction. Framework I predicts both the presence of the targets and the relevant geometric attributes. Framework II first processes an input image as a sequence of ROIs, and applies two steps of feature extraction on each ROI: by CNNs and by RNNs. The latter can automatically recognize global structures over multiple ROIs. Optionally, higher level models (such as those in [6]–[8]) of the lane structures can be constructed based on the predictions. This step is shown as a transparent box lane modeling. The details of the two frameworks are explained in Figs. 3 and 5, respectively.

The motivation behind this paper is to design neural networks that learn features from inputs and output predictions customized to the applications. Task-specific feature extraction has been proven helpful. The learning nets [2], [18], [23] have shown that given raw inputs, by adapting the outputs to the desired category labels, useful low-/middle-level features will emerge automatically during the training process, and the entire network routinely outperforms classifiers tuned to handcrafted features. However, as we have discussed above, in most existing neural networks for image analysis, the task is simplified to making choices from certain categories. But in real world, the targets are often beyond independent individual categorical labels. An effective way to encode structures in the network outputs is to have the internal status of the network preserved from one instance to another. RNNs employ a type of neuron units that have cyclical connections to themselves, where the output of a neuron is fed into its input. Therefore, the prediction on one instance will affect that on the subsequent ones. RNNs have been shown useful in sequence analysis [24]–[26], as well as many other applications in control, modeling, and signal processing [27]–[29]. Backpropagation through time and other algorithms have been developed to enable RNN training [30], [31]. Long-short-term-memory (LSTM) has been proposed to overcome the gradient dissipation or explosion during the

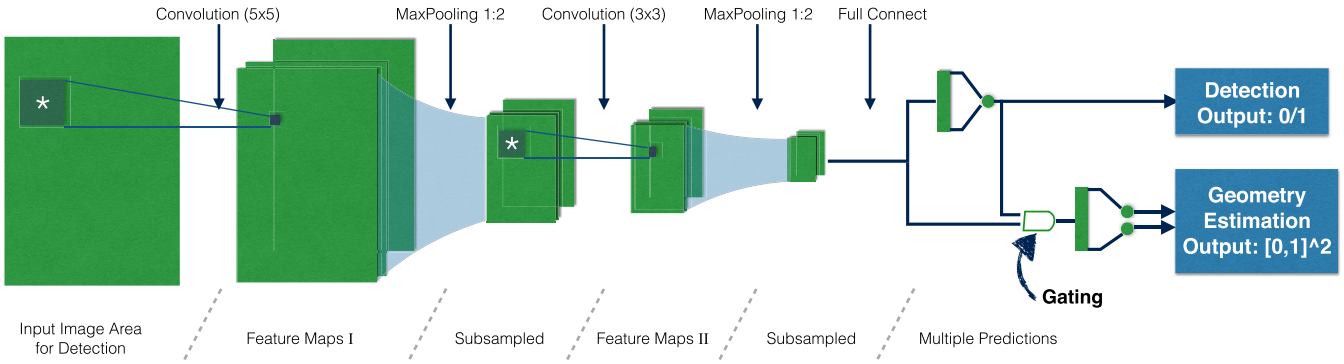


Fig. 3. Multitask object detector based on hierarchical convolutional network. Left to right: a signal flows through the operational steps of the neural network, which finally produces two kinds of outputs. The network accepts a rectangular area from an image (ROI). The early steps of the network are based on a LeNet [17], consisting of twice application of convolution (each convolutional kernel, e.g., a 2-D filter of 5×5, producing one feature map) and downsampling (shrink the size of the feature maps by pooling the maximum filter responses from local, e.g., 2×2, areas). At the output stage, the information flow branches and network simultaneously predicts: 1) whether the target is present in the input image area and 2) the geometry attributes of the target if 1) is true. It is because of these two outputs, or two tasks, that the network is called a multitask network.

backpropagation by introducing gates to conditionally regulate the nonlinearity in the nets [32], [33]. LSTM is successful in long sequence analysis [15]. For image analysis, LSTM was extended to multiple directions and dimensions and applied to handwritten digits recognition [34]. Recently, LSTM has been employed to learn the directing attention of perception in image recognition [35]. In this paper, we adopt LSTM as the structural learner of images.

Recognizing lanes is an important step toward understanding the traffic scene and ultimately toward autonomous driving [36]. Most existing efforts on lane recognition rely on handcrafted features to detect local cues, i.e., marks of lane boundaries [6], [37], [38]. Explicit lane modeling and corresponding robust selection methods (e.g., [39]) are necessary to infer global lane structures from local detections. Despite the appearing simple patterns of most lane marks, the task involves nontrivial tradeoff between several important aspects, including the speed, the accuracy and resolution of the detection, and modeling reliability [6], [40], [41].

A natural approach of recognizing objects is to employ part-based models and systematically compose the part models in a comprehensive classifier [42]. The so-called deformable model employs manually constructed features, which can be optimized for the task if the feature learning can be integrated in the model learning procedure. More importantly, existing deformable models rely on explicit model families, where the components and geometrical correlations of the target object are prescribed. Although such visual grammar can be designed for objects, such as pedestrians or bicycles, it is usually difficult to explicitly identify the components and the internal structure for object without certain constituent parts or shapes, such as traffic lanes or other elements of road. In this paper, the local-detection-and-global-modeling of the traffic lanes has been formulated as a sequence modeling problem. The structural learning problem has been tackled by probabilistic fields, such as Markov random fields [43] or conditional random fields [44] and, more broadly, by energy-based models [45]. The probabilistic fields have been successful for problems such as medical image analysis and segmentation [46]. For more complicated visual cues, it is usually not obvious how to design the appropriate features of the energy function. Moreover, it is difficult to represent objects with global support in an image by local structures specified by the features. In contrast, we propose to employ the LSTM RNN to capture such structures in this paper, where the long-term internal memory suits the need of the problem.

III. Multitask Convolutional Network

The specialized deep CNN simultaneously detects marked lane boundaries and extracts the geometry attributes of the boundary for positive detections. We adapted the hierarchical structure of the LeNet [17] to perform the two tasks, which share the low-level features.

A. Convolutional Network

The overall structure of the network is shown in Fig. 3. The input to the network is a ROI from an image. The feature extraction in a LeNet is through consecutively applying convolutional image filters and downsampling by pooling the maximum responses in neighborhoods on the image plane. A convolutional layer of P input feature maps (e.g., color channels) $\{X^p\}_{p=1}^{P}$ and Q output feature maps $\{H^q\}_{q=1}^{Q}$ is specified by a set of filters $\{W^{q,p}\}$ of size $K \times K$ and Q bias terms b^q. The output is calculated by

$$H^q := f\left(\sum_p X^p * W^{q,p} + b^q\right)$$

where the operator $*$ represents 2-D convolution and $f(\cdot)$ is an elementwise nonlinear activation function, e.g., $f(x) = 1/(1 + e^x)$. The resultant feature map H^q is then processed with spatial downsampling. For example, taking the maximum output from a local area of 2×2 of each feature map will shrink the signal by a factor of 4. Max-pooling improves the recognition robustness against the transformation of the pattern of interest within the input image. However, different from previous recognition-only tasks [2], we must make a tradeoff, because this spatial invariance reduces the

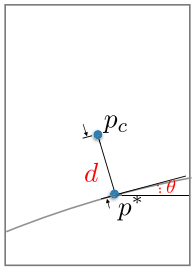


Fig. 4. How to represent geometric attributes? The attributes consist of (d, θ). The parameter d is a signed distance from the line segment to the center ("−" for above and "+" for below), which provides the location of the segment. The parameter θ is the angle between the segment and the horizontal line, which is the orientation of the segment.

information needed for estimating the geometric attributes of the lane border within the detection area. In practice, we employ 2-to-1 max-pooling twice, which provides robust detection, as well as effective geometric prediction. After the second step of max-pooling, all the output neurons are combined to feed in the next stage of the prediction network.

B. Branched Model Construction

The features given by the convolutional network are shared by two related prediction tasks. The detection task is essential to have a classifier to determine the presence of the target, e.g., segments of traffic lane boundaries, in the ROI. On the other hand, we also let the network estimate the geometric attributes of the target, when the detector returns positive results. In principle, the geometric attributes can refer to any quantities providing details about the target in the ROI. In our particular task, the geometric attributes are the position and orientation of a line segment within the ROI, representing a traffic lane boundary intersecting a rectangular area on road. Note that we consider inverse perspective mapped (IPM) images, where the perspective projection is removed from a camera image using known camera parameters, and each pixel corresponds to a point on the road surface (detailed discussion is found in Section V, and an example is shown in Fig. 8). The geometric attributes consist of two parameters, including: 1) the signed distance between the line segment and the center of the rectangle and 2) the angle between the segment and the bottom edge of the rectangle. The geometric model configuration is shown in Fig. 4, where d represents the signed distance and θ represents the angle. The sign of d indicates the relative position between the line segment and the center ("−" for above and "+" for below).

The detection and the geometric estimation are implemented by two individual feedforward networks. Both the networks consist of a layer of hidden units, which are fully connected to all the outputs of the final max-pooling stage. The hidden neurons then form two generalized linear models for the prediction tasks. In particular, a neuron of the hidden layer (either of the two) is activated by

$$z_i^{\{C,R\}} = h\left(\sum_j U_{i,j}^{\{C,R\}} y_j + c_i^{\{C,R\}}\right) \tag{1}$$

where C and R correspond to the classification and regression branch of the network, $U_{i,\cdot}$ and c_i represent the coefficients and bias term for the ith neuron z_i of the corresponding branch, j runs over all the neurons output by the final max-pooling, and $h(\cdot)$ is the activation function. The classification and regression targets are computed as follows:

$$p = \sigma\left(\sum_i \phi_i z_i^C + b^C\right) \tag{2}$$

$$(d, \theta)^T = \sum_i \boldsymbol{\psi}_i z_i^R + \boldsymbol{b}^R \tag{3}$$

where ϕ_i and ψ_i are the coefficients, and b^C and $\boldsymbol{b}^R$ are bias. Note that bold symbols represent 2-D vectors. The loss functions for the classification and regression tasks are defined by negative log-likelihood and squared errors, respectively. Given the ground-truth label $g \in \{0, 1\}$ and geometry $(\hat{d}, \hat{\theta})$, the losses are

$$L^C = -\log p^g (1-p)^{1-g} \tag{4}$$

$$L^R = [(d-\hat{d})^2 + (\theta-\hat{\theta})^2]g. \tag{5}$$

Compared with the standard squared errors, loss (5) has the detection label g as an additional switch. The switch has the following meaning. Of the two branches of the network, the classification is always performed. The estimation of the geometry of the border segment only makes sense if there is one within the detection area; otherwise, the geometry prediction will be skipped and no loss should be counted.

According to Fig. 3, the switch of the gate is connected to the signal at the binary output of the classification. It is worth noting that the switch operates in slightly different ways during the feedforward and backpropagation processes. In the feedforward process, the gate is controlled by the network prediction and a binary status determined by p in (2), which means that the geometry estimation is only valid when the target is detected. This applies to both the training and test samples. During the backpropagation process, the gate is controlled by the ground-truth status g, i.e., the objective of the detection. Backpropagation is only performed on the training samples, where the ground-truth g is provided.

IV. Recurrent Neural Network

In this section, we present a neural network that makes structured predictions and is able to detect local visual cues accounting for the global object structures. The multitask network introduced above can help build global object model by providing local geometric attributes. However, when it is inconvenient to explicitly prescribe a global model, it is desirable to enable the model to capture the global structures of the distribution of the cues and apply the knowledge in detection.

To achieve the global awareness by local observations, we allow the model to maintain internal memory. When the model takes inputs by spatially traversing the image, the memory enables the contents of the image in one part to affect the analysis of the other part. When trained properly, such a model is able to recognize the meaningful structures in the image.

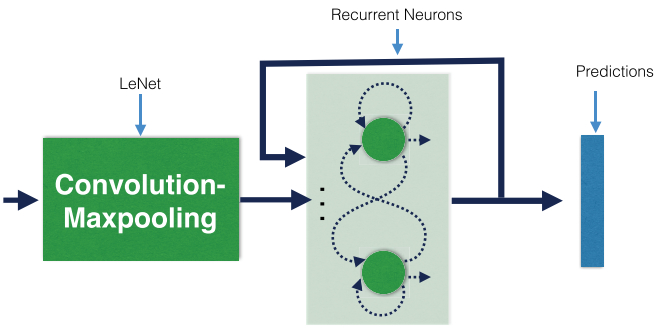


Fig. 5. Structure-aware detector based on RNN. The initial steps represent the input region using the convolution-maxpooling operations, as shown in Fig. 3. The predictions are made from the layer of recurrent neurons, which are computed from the extracted features of the current instance and their own current status. Since the status of the recurrent neurons forms the system state, at any moment, the network takes the previous observations into account to make the current predictions.

A. Network Structure of Recurrent Detector

Fig. 5 shows the chart of the main components of the recurrent network. From the input signals to the output predictions, there are three main stages. First, the inputs region passes a convolution/maxpooling step, which extracts representative features from the raw signals. The settings of the convolutional feature extraction net layers are similar to those used in Section III, with minor adjustments to adapt the overall network configuration. The second stage is implemented by a set of recurrent hidden units. In the third stage, the status of the recurrent neurons is transformed by a layer of classifiers to produce the outputs.

Similar to the LeNet in Section III, the convolution-maxpooling layer generates multiple feature maps for the input image and takes the maximum responses in local areas. There are two differences. First, there is only one, rather than two, pass of convolution-maxpooling in our construction of the recurrent detector network. Recurrent detector generally accepts smaller images as inputs and needs fewer downsampling steps. It is noteworthy that using smaller images is not a trivial pragmatic setting. The ability of drawing connections between local predictions allows the network to deal with missing cues in small-sized inputs. Second, the convolution-maxpooling layer does not include a nonlinear activation step, because subsequent hidden neurons are complex units with nonlinear activation functions for the inputs. The details of the hidden neurons are introduced in Section IV-C.

Following the convolution-maxpooling layer, the recurrent hidden layer further transforms the signal. The final predictions are made based on the status of the recurrent hidden units by using a standard feedforward layer.

B. Detection on Images

Given an image, the neural network is applied to multiple ROIs, which are fed to the network consecutively. In each processing step, the network produces predictions for the input ROI, as well as maintains the status of the recurrent units for the prediction of the next step.

The partition of an image and the order in which the recurrent net is applied should be designed to suit specific problems.

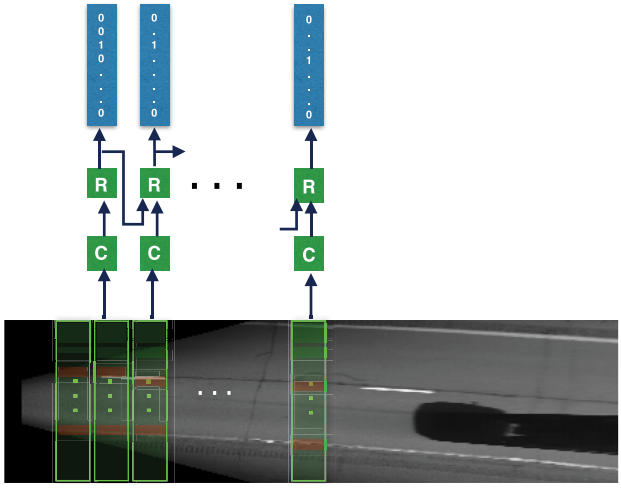


Fig. 6. Applying RNN detector on road surface image. The detection process of lane boundaries on road surface is shown. Each ROI is a strip (green rectangles). The neural network processes the strips in an image from left (near) to right (far). For each ROI, multiple binary decisions are made, corresponding to detecting the target (lane boundary) in small patches within the ROI. Each ROI consists of a stack of such small patches (best to be viewed in colors on a computer screen), and the red patches contain lane boundaries. In the charts showing the neural network layers, "C" represents convolution-maxpooling layer and "R" represents recurrent hidden layer. The CNN and the RNN are shown in Figs. 3 and 5.

For detecting lane boundaries in road surface images (see IPM discussed in Sections III-B and V), we let individual regions be narrow strips crossing the road, so that the sequence of the strips cover the road surface. The arrangement of the regions and the detection procedure are shown in Fig. 6. Because each ROI takes a thin slice of the road surface and the slice is roughly perpendicular to the road direction, we can expect the boundaries of the lanes occupy small segments within each ROI. The detection is, thus, formulated as multiple binary predictions for individual ROIs. In Fig. 6, some of the patches are recognized as boundaries not according to the presence of visual cues but because of the favorable context observed in previous ROIs in the input sequence.

There are several practical advantages of the detection problem as formulated above. An important one is that large numbers of training examples can be generated from only a few labeled images, which is discussed in Section V.

C. Recurrent Units

In the following, we briefly introduce the LSTM algorithm, where [47] can be referred to for more details. A layer of standard recurrent neurons can be formulated as

$$h_i^{t+1} = f\left(a_i^{t+1}\right) \tag{6}$$

$$a_i^{t+1} = \sum_j w_{ij} x_j^{t+1} + \sum_k u_{ik} h_k^t \tag{7}$$

where $h_i^{(t)}$ represents the status of the ith recurrent neuron at step t, x-variables are the neurons of previous layers in the network, and w and u are the connection weights. The $f(\cdot)$ function is a nonlinear activation. In an LSTM network, the individual recurrent neurons are equipped with several gates, regulating the flow of signals. Fig. 7 shows one LSTM cell.

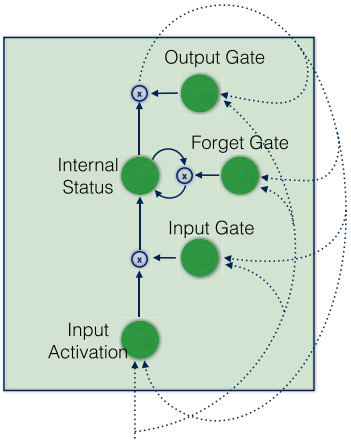


Fig. 7. Structure of an LSTM cell. Solid arrows: internal paths of the neuron activation signals. Dotted arrows: paths cross cells (from lower layers to the input and gates) or cross time steps (from the output to the input and gates). The internal status is the LSTM memory. The gates regulate information paths through multiplication operators (shown as $\otimes$).

There are three types of gates, as well as additional input and output activations. The cell status consists of modulated activated network input and a decayed memory of previous status. Therefore, by replacing (7), we have

$$a_i^{t+1} = c_i^{t+1} a_i^t + b_i^{t+1} g\left(\sum_j w_{ij} x_j^{t+1} + \sum_k u_{ik} h_k^t\right) \quad (8)$$

where b-variable represents the input gate, c-variable represents the forget/keeping gate, and $g(\cdot)$ is a nonlinear activation. LSTM network has also an output gate checking the cell's contribution to the rest of the network. So in LSTM, (6) is replaced by

$$h_i^{t+1} = d_i^{t+1} \sigma\left(a_i^{t+1}\right). \quad (9)$$

The d-variables are output gates and $\sigma(\cdot)$ is a nonlinear activation. The gates, i.e., the b-, c-, and d-variables, in (8) and (9) are themselves neurons. Their net input constitutes the signals from lower layer neurons x, the internal status of the hosting cell a_i, and the previous output of all cells h

$$\alpha_i^{t+1} = g\left(\sum_j w_{ij}^{\alpha} x_j^{t+1} + \sum_k u_{ik}^{\alpha} h_k^t + v_i^{\alpha} a_i^{t+i_\alpha}\right) \quad (10)$$

where $\alpha \in \{b, c, d\}$ represents the gates. Note that the order of execution is important: input gate (b), forget gate (c), cell status (8), output gate (d), and final output (9). Thus, in (10), the input and forget gates use old status $i_b = i_c = 0$, and the output gate uses updated status $i_d = +1$.

V. Experiments

The proposed models have been tested on the real-world traffic data. In this section, we discuss the performance of the proposed model compared with widely used detectors, and the characteristic behavior of the proposed model, as well as practical scheme of integrating the model in comprehensive traffic scene analysis. According to the attributes of the models proposed in Sections III and IV, two sets of tests have been conducted, regarding merging and normal driving scenarios, respectively.

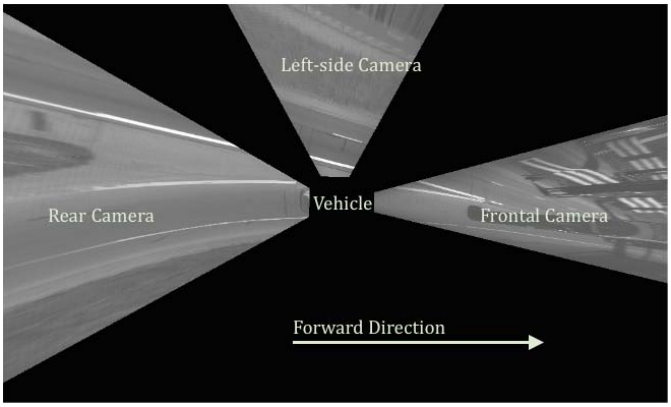


Fig. 8. Example of the IPM image. Each pixel corresponds to a 0.1×0.1 m^2 ground area (assuming that the road surface is flat). Depending on the application, an IPM image can aggregate one or multiple camera images to a unified map of the road surface. An IPM image integrating three camera observations is shown.

The multitask deep CNN has been applied to detect lane boundaries in a merging scenario, where the estimated geometric attributes can help further mapping of the scene. On the other hand, when driving on a multilane road, the number of lanes can vary and a general model can be cumbersome to specify. Thus, it is preferable to directly infer the lanes from the visual cues. Section V-A discusses the experiments on multitask deep CNN, and Section V-B discusses the flexible lane detection and modeling by RNN.

A. Lane Mark Detection and Geometry Estimation With Multitask Deep CNN

For the merging scenario, we use images taken by three cameras facing the front, left, and rear sides of the ego-vehicle. The observations are integrated in one IPM image using the calibrated camera parameters. An IPM image can be seen as a bird's view of the road surface recovered from the camera image by inverse perspective projection—mapping pixels in a camera image to a virtual horizontal plane approximately corresponding to the road surface. The inverse perspective is accurate only for the physical points on an assumed flat road, but is adequate as visual cues for a lane boundary. Fig. 8 shows an example of the IPM images we use for detection.[1]

For training and evaluation, the lane marks in the IPM images are labeled. Given a labeled IPM, two sets of training samples are extracted. For conventional local detectors, the samples consist of 7×7 image patches, which is corresponding to an area of 0.7 m $\times$ 0.7 m in the physical world. According to our discussion in Section III, the multitask CNN accepts samples of 28 $\times$ 20 patches.[2] Positive samples are

[1]We use this IPM image to illustrate the behavior of the detection algorithms. The image is about freeway merging from right, but the approach applies equally to the case of merging from left.

[2]Traditional detectors output a binary decision for a particular input. Thus, a patch must be reasonably small to reach an acceptable resolution of detection. In contrast, the proposed network additionally estimates the geometry, which makes further refinement possible. Relatively large patches are fed into the network to include additional contextual information.

Fig. 9. Intensity bump—a pattern of road markings to be detected in images.

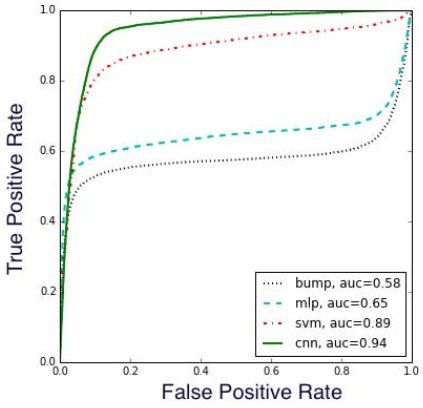


Fig. 10. ROC curve of detecting line segments on the IPM image. ROC curve shows the rates of correctly detected targets against those of incorrectly reported spurious detections: the higher the true positives achieved with lower false positives, the better the detector performs. Area under the curve (AUC) quantitatively measures detector accuracy. A large AUC is desirable.

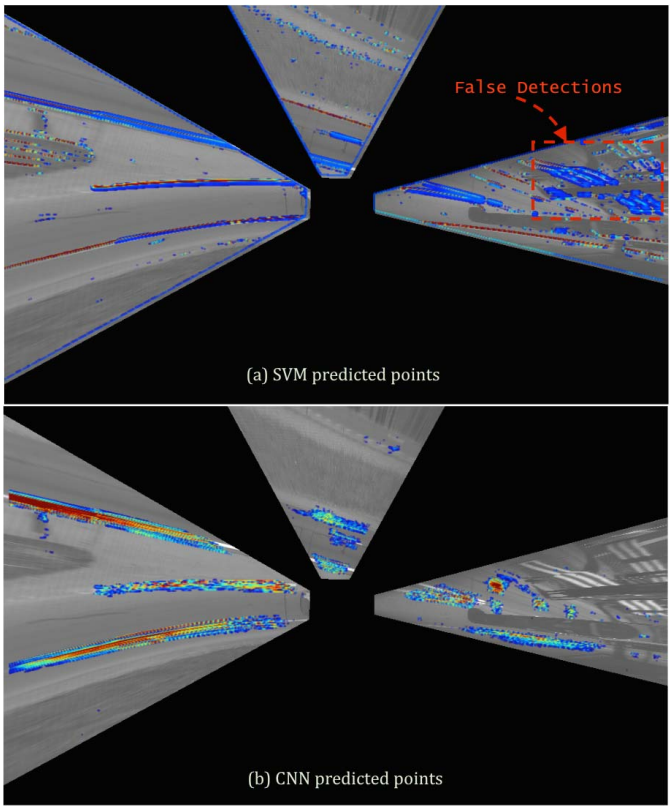


Fig. 11. Detection scores given by (a) SVM and (b) CNN. Both classifiers are applied to each image point to predict the presence of lane markings. The top 10% scored points are plotted with colors representing their respective scores. Blue regions: relatively low scores (among the top 10%). Red regions: relatively high ones. The locations of the top 10% scored points given by SVM and CNN are different. CNN tends to score points with more respect to how close the points are to the true lane markings than SVM does. SVM is affected by background clutter. (This figure is best viewed on a computer screen in color.)

extracted at the locations of labeled lane markings; geometric information is encoded, as shown in Fig. 4. Negative samples are random patches containing no marks.

We first examine how the proposed CNN model compares with widely used classifiers as a road marking detector. Three commonly used classifiers have been applied. The simplest detector checks the output of applying an intensity bump filter to the image patch [37], [38], [48]. The filter is constructed heuristically to represent a plausible pattern of lane markings, as shown in Fig. 9.

A more systematic method is to learn classifiers from example patterns. The proposed CNN and two widely used classifiers, support vector machine (SVM) and feedforward neural network (MLP), are tested [6]. The classifiers are applied at every pixel in the IPM image to predict whether a segment of lane marking is present. Fig. 10 shows the Receiver operating characteristic (ROC) curves of those classifiers. CNN has achieved superior performance in this test according to the ROC plot. The behaviors of the bump filter response, MLP, and SVM are as expected. Both the SVM and MLP performed better in the detection task than the simple filter, likely because the image characteristics learned from data are more representative than the handcrafted intensity bump. For generating the ROC curve with a range of varying false-versus-true positive rates, SVM is configured to output the probability of its prediction. The scores of the CNN model consolidates the prediction of both the classifier and the geometry predictor: $s = y - \alpha|d|$, where y is the log probability given by the classifier, $|d|$ is the predicted distance between the center of the region and the line segment, and $\alpha = (\text{PatchRadius}/2) = 10$.

In addition to the hit-or-miss criterion shown in the ROC curves, the decisions made by the CNN-based detector are also more accurate in spatial terms, i.e., the confidence of the detector is closely related to the distance between the sample and a true target. Fig. 11 shows an example by applying SVM and CNN at each location on an image and comparing the top 10% scored points.[3] The plots show that CNN has a different map of detection scores than SVM does, particularly for points that are not located exactly on the target curves. The CNN tends to give higher scores to the locations close to the target curves, and the score is linked to the distance. In contrast, SVM simply relies on local texture of the image to detect lane markings. If some extent of false detection is to be expected in practice, those of CNN are more benign to the subsequent task of modeling the road than those of SVM. This benefit of CNN is hardly surprising, because CNN considers a large context and extra geometric information for detection compared with standard classifiers.

CNN-Based Active Search Scheme: Both the efficiency and accuracy of CNN-based detection can be further improved by exploiting two attributes of the network: 1) the detection area is relatively large and 2) the location and orientation of the target can be estimated when detection is made. An active

[3]Note that such an exhaustive search is not how CNN is usually applied, see active search below.

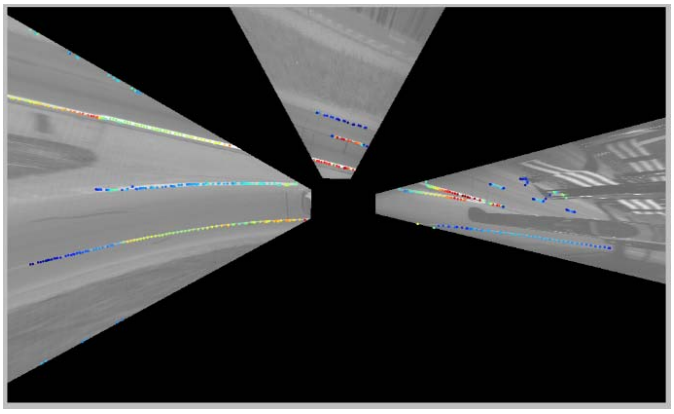


Fig. 12. Detection with active search and CNN. The result is presented similarly as in Fig. 11, and improvement is evident. Details of the active search are in the text. Here we show all detections, rather than the top 10% scored points as in Fig. 11.

search scheme can be designed, so that once a detection area is examined, we can employ lightweight weak detectors and avoid evaluating the entire network on overlapping and surrounding areas. This is different from most detectors, for which a sliding window searching must be performed on a dense grid of the image, and the detection areas are overlapping with each other heavily. For example, if SVM is applied to large image patches, one will have a binary prediction on a large patch, without any hint to refine the location of the target within the patch.

In contrast, when the CNN reports a detection and predicts a line segment within a rectangle area, we then apply the bump filter (see Fig. 9) within ± 3 pixels of the predicted location of the segment [see Fig. 4 (point p^*)], and refine the location to where maximum response is. Then, the search moves to a new site following the predicted orientation of the lane. The process continues until the search encounters previous detections or track is lost (all filter responses in a local area are low). Fig. 12 shows an example of detection by searching, which further improves the detection quality.

B. Lane Boundary Detection via Recurrent Neural Network

Lane Detection by Recurrent Neural Networks: As shown by the above experiments, the multiple predictions by the CNN can be helpful to subsequent modeling. However, in many practical scenarios, a prescribed road or a lane model can be too restrictive. The RNN introduced in Section IV is a suitable learning model for tasks where the visual cues can only be partially observed. For example, when the lane boundaries are defined by broken lines, target (boundaries) can exist at locations where the local appearance is the same as nontarget (road surface). In such cases, the target object is best detected by accounting for both the appearance and the spatial structure of the visual cues. Moreover, the RNN learns structures implicitly from the local labels in the training data without heuristic knowledge about the structures.

The setting up of the prediction problem has been discussed in Section IV-B. An example is shown in Fig. 6. In particular, we use the strips of 10×80 pixels on the IPM (road surface image). The neural network generates 16 predictions for each strip, i.e., the predictions are about whether the 16 small patches (10×5, partitioning the strip along the 80-pixel elongated dimension) contain lane boundaries. In this test, we take totally 50 strips in an image, making an area of 500×80 in an IPM image. Note that, in practice, once a network has been trained, there is no limit on the length of the sequences the network can process in the test stage. We also include the data set in [49] in this test (Caltech data set). The data set consists of video clips taken during four sessions of urban driving with calibrated camera parameters and an IPM image generator. The original experimental configuration of the data set for temporal tracking of lanes differs from our testing objective of inferring partially marked lanes. Thus, we test the algorithms on individual frames. For data set, each sample is a sequence of 24 strips of 5×100 pixels, and on each of the strip, we make 20 predictions of the 5×5 areas.

For the networks on both the data sets, the feature extraction and recurrent neuron layers have the same structure. The convolution-maxpooling layers have 5×5 convolutional kernels and 2-to-1 pooling. We employ 64 LSTM cells in the recurrent layer. The RNN layer is specified by $64 \times 4 \times \#.\text{features}$ weights, because each LSTM complex cell contains four individual neurons, corresponding to the internal status, input, output, and forget gates.

The network is trained by fitting data with labeled lane boundaries. Labelling images is expensive and slow. Fortunately, fitting the model actually requires labeled strip sequences, which can be obtained in large quantities from a few labeled images. A distinctive sample of sequence can be extracted by shifting the position of each strip in a labeled image and adjusting the labels correspondingly. Intuitively, one can understand the procedure by considering the sequence as a stack of strips and producing multiple training samples through sliding each strip by a small distance.

The effect of the structural information is best shown by comparing the proposed recurrent network with detectors making independent predictions. First, we train an SVM classifier on the individual patches, i.e., the small areas within one strip (16 for our data set and 20 for the Caltech data set). The classifier treats each small patch as independent instances. As we have discussed above, the small patches contain insufficient context to allow the classifier to deal with ambiguities. What is worse for small-patch detectors in this test is that the positive labels are given to virtual boundaries, which means that for some positive samples, no local visual cues are present. Thus, for small-patch classifiers, the positive supervision is unreliable, and the confusion is very damaging to the training process. An example is shown in Fig. 14(a_1), where local appearance on the boundaries can have similar appearance as surrounding road surface [at locations between the marks along a boundary of broken line, as indicated by circle marks in Fig. 14(a_1)].

Second, we attempt to treat the individual strips as independent samples. This is done by removing the recurrent neurons and replacing the layer with nonrecurrent hidden neurons. In particular, the network has 64 fully connected hidden neurons between the convolution-maxpooling layer and the final output layer (referred to as CNN network).

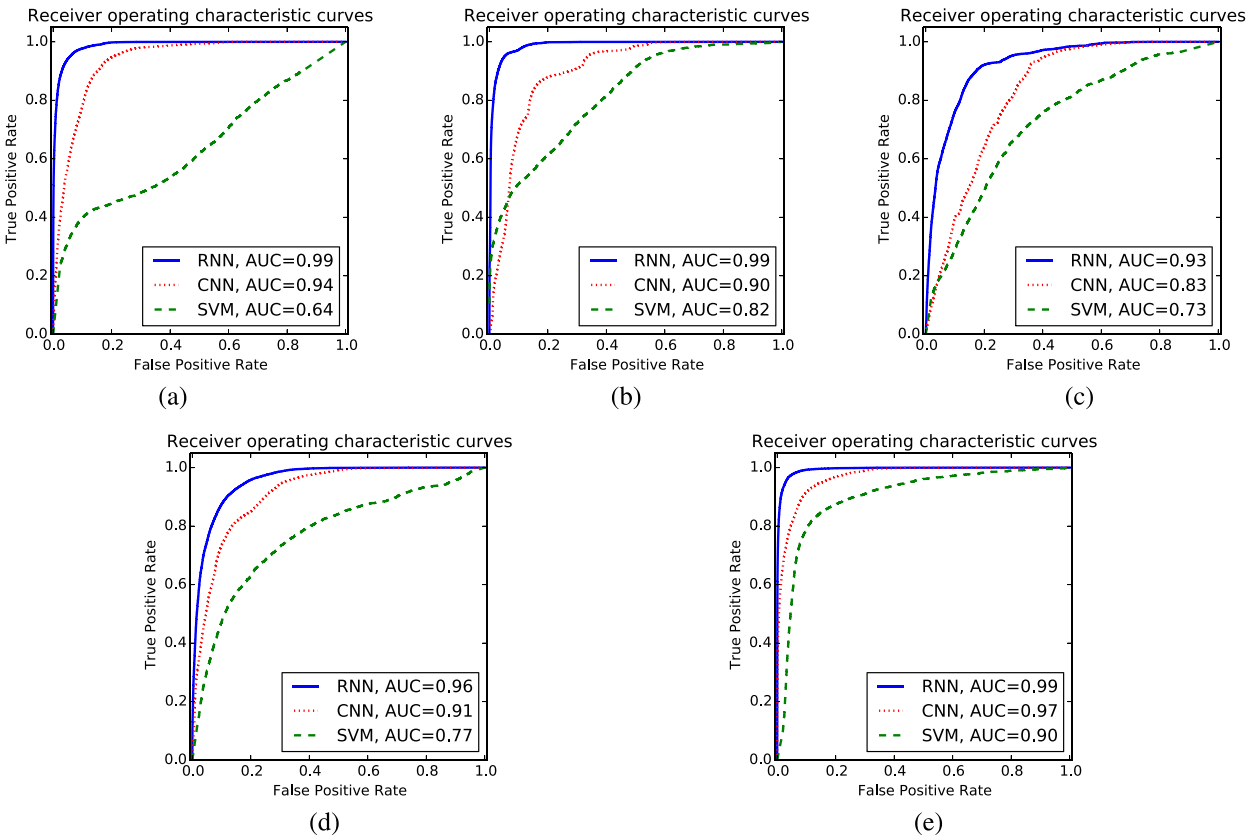


Fig. 13. ROC curves of lane boundary detectors. (a) Results on our data set. (b)–(e) Results on four video clips in the Caltect data set of lane marks [49]. The ROC curves and AUC areas compare the prediction of the detectors and the ground truth, which is explained in Fig. 10.

The ROC curve is shown in Fig. 13. Fig. 13(a) corresponds to the test result on our data set, and Fig. 13(b)–(e) represents the four video clips in the Caltech data. These curves shows that the performance of a detector is largely affected by how the detector uses the structural information in the data. The RNN network outperforms the CNN network, because the recurrent neurons allow inference about one strip to help that about subsequent ones. The CNN network accounts for contextual information within a strip, but also suffers from the lack of sequential structural information. By a large margin, the SVM classifier on the small patches is outperformed by the CNN.

Noticeably, the SVM detector has behaved poorly, not only compared with the neural network-based models but also compared with its own performance in Section V-A. We have trained and verified the SVM models using a reasonable range of settings, including both the linear and radial basis function (RBF) kernels, and on a parameter grid of $C = 10^{\{0,1,2,3,4\}}$ and $\gamma = 10^{\{-2,-3,-4,-5\}}$. Such settings have given good models in the experiments in Section IV-C. It is unlikely that it is an inadequate training or model selection procedure to blame. A possible reason for the poor SVM classification is discussed above: the training samples are labeled with hypothetical lane boundaries. If we consider the individual small patches, many positive samples contain no distinctive patterns. The lack of context and the confusing positive samples jointly prevent classifiers learned from independent small image patches (such as SVM) from reaching competitive performance.

Fig. 14 visually compares the detection results by the three models on two example frames from the video. The predictions on the road surface are drawn over the camera images. The blocks represent the predictions on the small patches in the strips, and the colors of those blocks correspond to the confidence of detecting lane boundaries (red for positive detections). Fig. 14(a)–(c) shows the results of the RNN, CNN, and SVM, respectively. The effectiveness of the detectors can be readily assessed by visual inspection, which is consistent with the quantitative comparison shown by the ROC curves.

Testing the Rule-Learning of RNN: To further clarify how the recurrent network helps recognize the structures, we have generated a data set consisting of sequences of graphical patterns with different meaning. The goal is to let a learning model automatically identify the cues by the graphics, and infer the rules by which the cues influence the subsequent predictions. Example images are shown in Fig. 15(a) and (b), where the patterns and rules of interpreting the images are also explained. Two neural networks are constructed in the similar way as in the previous experiments. The RNN has 16 kernels sized 5×5, a 2-to-1 maxpooling layer, 64 recurrent cells, and 4 output predictions. The number of output predictions, four, is different from the previous RNN, because each sample

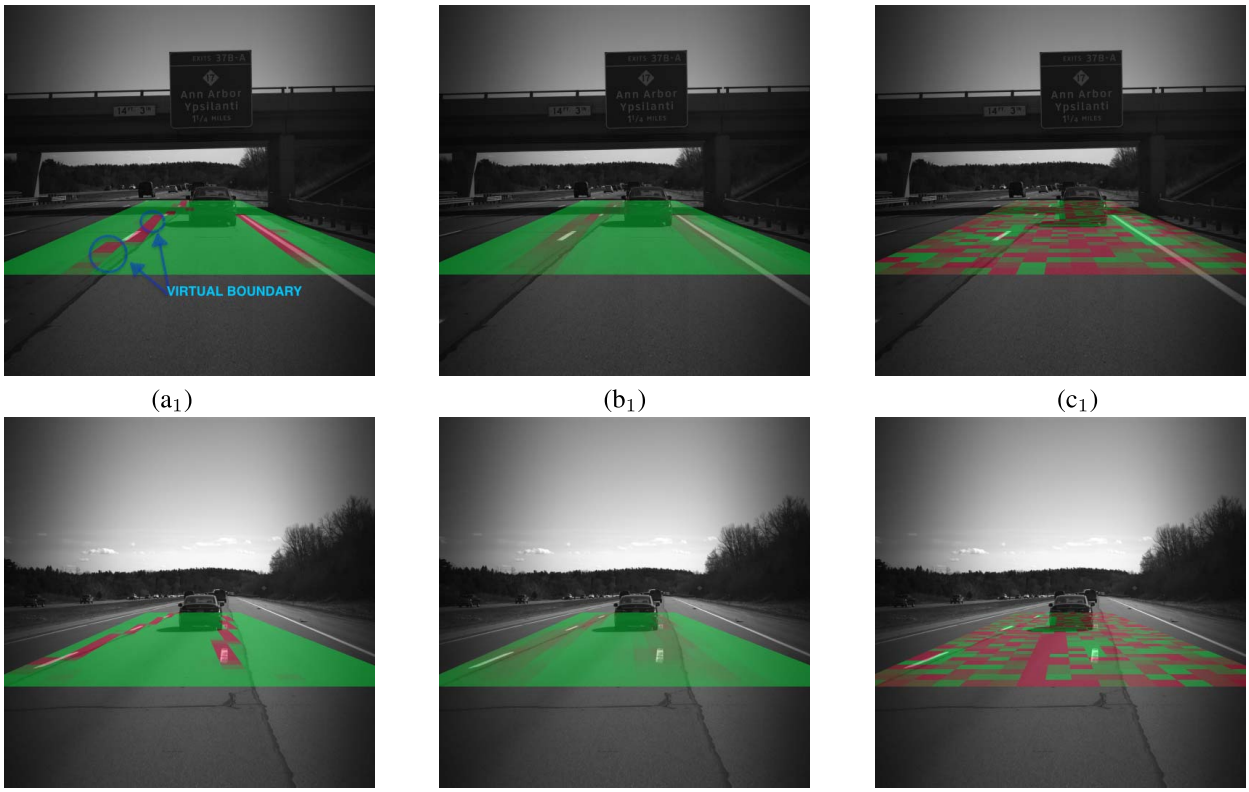


Fig. 14. Detecting lane boundaries with and without structural information. Results of the (a) RNN network, (b) CNN network (similar structure as the RNN network, but replacing recurrent neurons with standard feedforward ones), and (c) SVM models. Colors in the small blocks indicate a detector's confidence that the block belongs to a lane boundary. Red: high confidence. Green: low confidence. (This figure should be viewed in color on a computer screen.) The amount of structural information considered in the models has the following order: (c) SVM < (b) CNN < (a) RNN. (c) SVM predicts on individual blocks and is largely affected by the lack of context. A row of blocks makes a strip. (b) CNN predicts on the individual strips, and its confidence is affected by virtual boundaries containing little visual cues. (a) RNN accounts for the structure within a sequence of strips, and captures boundaries more successfully. In (a_1), we use circle marks to indicate areas of virtual boundary: boundary without apparent visual cues.

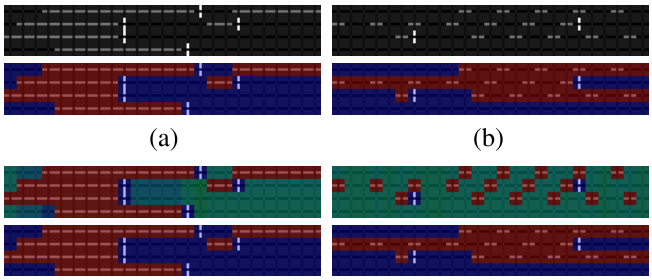


Fig. 15. Synthetic data: different graphical cues and the meaning. Each image contains four rows. Each row is a sequence of 25 small boxes (10×10 pixels). A sequence is interpreted from left to right and one box after another. Initially, an empty box means a negative status. The pattern "–" in a box turns ON positive status. The positive status keeps on until a pattern "|" is encountered. (a) and (b) Examples of the data. Top: original image. Bottom: ground-truth status overlain on the image. Dark red region: positive prediction. Dark blue: negative prediction. Green region: less confident predictions (spectrum in-between). The images are best seen on a computer screen in colors. (c) and (d) Predictions made by the CNN and the RNN networks on the two data samples. The results of CNN and RNN are on the top and the bottom, respectively.

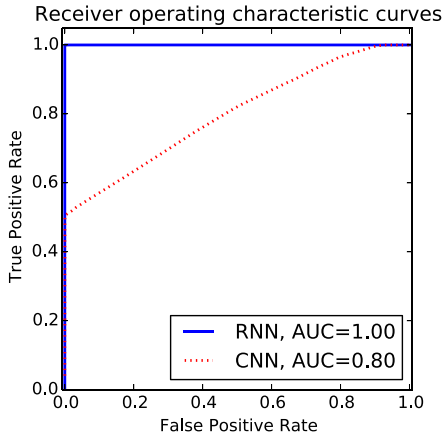


Fig. 16. ROC curves of two neural networks (CNN and RNN) on the synthetic data.

in the data set has four sequences (see Fig. 15). As above, the CNN is constructed similarly as the RNN, with the recurrent cells being replaced by feedforward neurons.

The performance of the two neural networks is compared by the ROC curve in Fig. 16. Two examples of the predictions made by the networks are displayed in Fig. 15(c) and (d). It can be interpreted from the plots that the RNN has made accurate predictions, which shows that the network has learned the rules about how the sequences of the graphical pattern determine the status. On the other hand, the CNN predicts based on independent strips (a column of four blocks in the images). When the visual cues are present, the CNN can

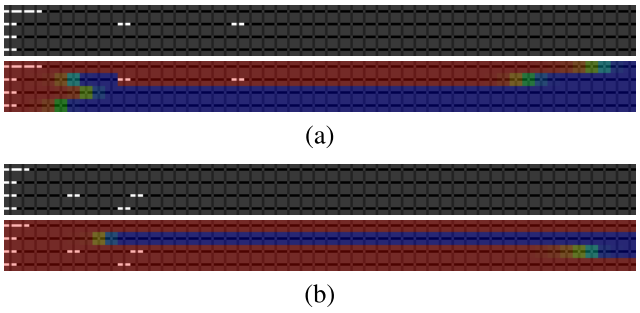


Fig. 17. Example RNN predictions on unbounded sequences. (a) and (b) Examples of the sequences of length 50. As in the training data, four parallel sequences make one sample. Without a stop pattern "|," the desirable result predicts the entire sequences as positive status. However, only a few (1–3) "–" patterns are given in the beginning of each sequence as cues. The purpose of this test is to check whether and how well RNN has learned to use its internal memory to realize and generalize the rule, and continue producing positive predictions after it has seen the cues.

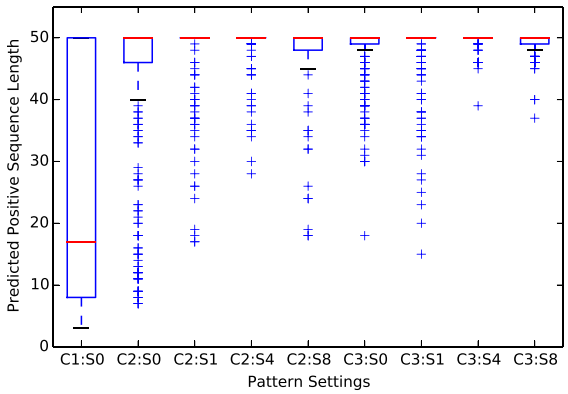


Fig. 18. Statistics of RNN predictions on unbounded sequences. Box plots: positive predictions for length-50 sequences, where a sequence is initiated with 1–3 "–" patterns, but without the stopping "|" pattern (unlike Fig. 15). Each box plot corresponds to an experiment setting. "C" (for cue) means the number the "–" pattern repeats. "S" (for skip) means the empty boxes between consequtive "–" patterns. Since the "–" pattern opens a string of positive status when training the network, more "–" patterns gives the network stronger evidence to continue outputing positive predictions for more steps. The trend can be seen by comparing the box-plots of "C{1, 2, 3}:S0." With a few empty boxes between the "–" patterns, most sequences are completely predicted all positive, e.g., C{2, 3}:S{1, 4}.

make correct predictions. But the network is confused by the empty patterns, because the empty patterns have no consistent labels if seen separately. In this experiment, we do not include the classifiers on individual small blocks, such as SVM. The described deficiency of the CNN applies to such models as well.

As shown in Fig. 17, when a "–" pattern is observed, positive status is to be assigned to the block and subsequent blocks, until a "|" pattern is encountered. We examine how the RNN utilizes its internal memory to realize this rule. The testing conditions are different from those of the training time. Sequences of 50 empty blocks are generated, then one to three blocks in the beginning are filled with "–" patterns ([ith zero, one, or more empty blocks between the "–" blocks (see Fig. 17)]. The sequences are unbounded (i.e., without the stop signs "|"), and the entire sequence should be predicted as positive status. Fig. 17 shows two examples of such sequences and the predicted probabilities by the RNN. We have tested RNN repeatedly on 1000 examples, and the statistics is shown in Fig. 18. The result shows that a string of positive predictions is produced when a signal "–" is observed. When more than one "–" block is present, in most cases, the network predicts positive for the entire sequence (50 blocks, while the net is trained using only 25 block sequences). In the training data, there can be zero to three empty blocks between the "–" patterns. The network may indeed mispredict for some steps when significantly more, e.g., eight, empty blocks are present between the "–" patterns. Nonetheless, the RNN shows its capability of generalization, which indicates that the rule we designed for this test has been learned effectively. The observed behavior is consistent with what to expect from the well-trained RNN on challenging sequence prediction tasks (see [28], [29]).

Working With Higher Level Lane Modeling: We have shown that the proposed detection framework can learn and detect the visual structure of the lanes, without resorting to explicit lane modeling [6]–[8]. However, although lane modeling is no longer mandatory, prior knowledge about the lanes can still be useful when available. The proposed network can be integrated as the base detector in an explicit lane modeling framework. As an example, we report our implementation and test results of the lane detection system in [6] in this section.

The lane modeling system relies on the positions of the potential lane boundaries returned by the base detector. Robust geometric modeling is then carried out to sort out the boundaries and correspondingly the lanes. The geometric models are constructed similarly as those in [6]: random sample consensus (RANSAC [39]) is employed to select two or three boundaries in an image, and each boundary is a straight line with two control points or a quadratic curve with three control points. All the model parameters are generated by a stochastic program. The number of control points in each boundary is randomly selected, and the control points are sampled from the candidate positions provided by the based detector. The proposals are assessed by the number of candidate positions that can be taken as inliers of a boundary. After a certain amount of RANSAC operations, the best proposal is accepted as the lane model of the image. In our experiment, we evaluate the accepted lane models by comparing the boundaries with those of the labeled lanes. The distances from the positions on the labeled boundaries to the nearest boundaries in the lane model are measured in pixels.

We tested SVM, CNN, and RNN as the base detectors and performed the experiment on the real-life traffic images. For each frame, we: 1) take all the pixels that are annotated as lane boundaries; 2) compute the distances from the pixels to the curves of the resultant model (the distance from a pixel to the nearest curve); and 3) record the average distance from boundary pixels to the curves as the model error of this frame. Fig. 19 compares the statistics of the model errors obtained from individual frames belonging to five video sequences.[4]

[4]It is noteworthy that the criterion for performance discussed here for Fig. 19 is different from that is shown by Fig. 13. The comparison is between the annotation and the pointwise detection, as shown in Fig. 13, and between the annotation and modeled lane boundary curves, as shown in Fig. 19.

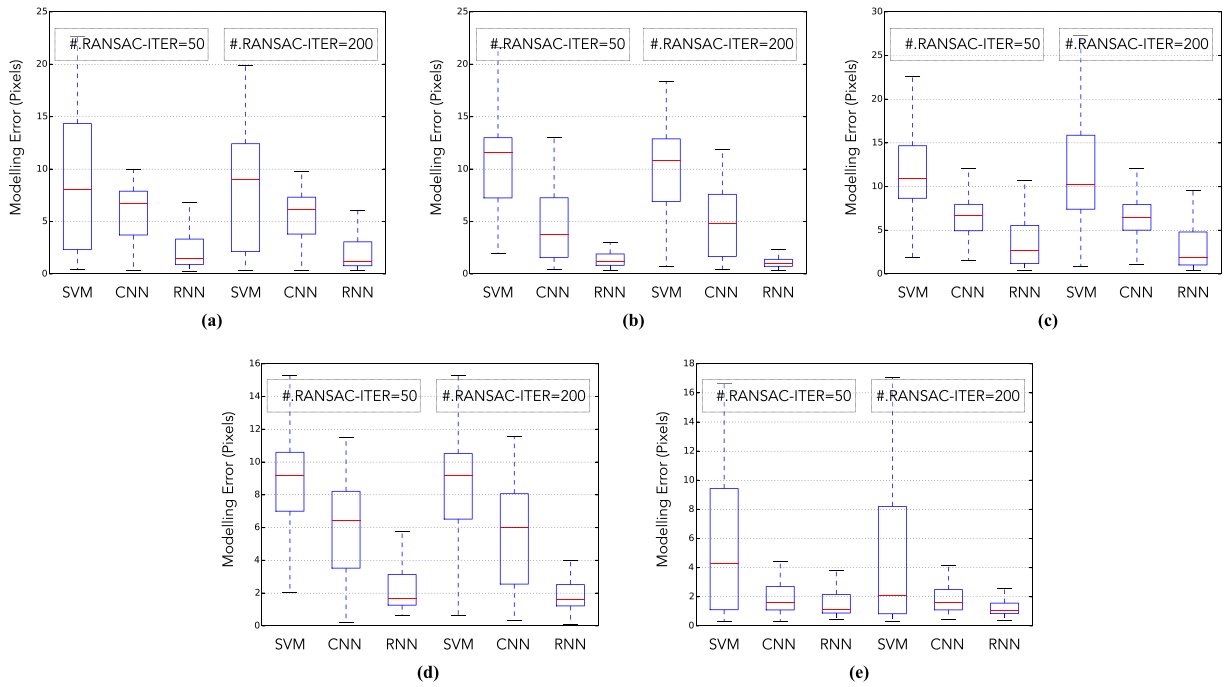


Fig. 19. Lane modeling [6] using different base detectors. The effects of the base detectors on the resultant lane models are shown. The proposals accepted by RANSAC are compared with labeled lane boundaries. (a)–(e) Statistics of the mean deviations obtained from individual frames belonging to the same five traffic video sequences as in Fig. 10 (see the text and the footnote for details on how the deviation is computed). Each subfigure displays two sets of results produced by 50 and 200 RANSAC iterations, respectively. While sometimes SVM results can be as good as or even better than those of RNN, the range of SVM results is much wider than that of RNN results.

The results show that employing better base detectors leads to superior recognition performance.

VI. Conclusion

We propose to adapt the framework of deep neural networks to learn the structures for visual analytics. Two new types of deep neural networks have been developed accounting for structures in images, and have been applied to recognizing lanes in traffic scenes.

A multitask deep CNN has been constructed, which allows sharing features learning between multiple prediction tasks. We demonstrate that the model serves satisfactorily as traffic lane mark detector. Furthermore, a recurrent neuron layer has been adopted on top of the convolutional feature extraction. The recurrent neurons serve as memory cells for the network and enable the network to learn structures in a sequence of predictions. Based on the RNN, we have designed a lane boundary detector, which can work with or without higher level models of traffic lanes. When being integrated within a model-based lane detection system, the RNN base detector improves the overall performance of the system.

Both the CNN and RNN detectors have been shown to be effective in detecting lanes in practical traffic scenes outperforming conventional detectors. In practical cases, the induction made by the RNN detector may link lane boundaries across obstacles such as vehicles. Such detections can be superseded by vehicle and other obstacle detection modules in a system, and the system can be aware of both the obstacle and the hidden lane boundary.

It is worth noting that the framework proposed in this paper is not limited to the detection of lanes. It can be adapted to other visual perception tasks.

References

[1] Y. Bengio, "Learning deep architectures for AI," *Found. Trends Mach. Learn.*, vol. 2, no. 1, pp. 1–127, 2009.

[2] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. NIPS*, 2012, pp. 1–9.

[3] B. Alexe, T. Deselaers, and V. Ferrari, "Measuring the objectness of image windows," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2189–2202, Nov. 2012.

[4] M.-M. Cheng, Z. Zhang, W.-Y. Lin, and P. Torr, "BING: Binarized normed gradients for objectness estimation at 300 fps," in *Proc. IEEE Conf. CVPR*, Jun. 2014, pp. 3286–3293.

[5] G. H. Bakir, T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar, and S. V. N. Vishwanathan, *Predicting Structured Data* (Neural Information Processing). Cambridge, MA, USA: MIT Press, 2007.

[6] Z. Kim, "Robust lane detection and tracking in challenging scenarios," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 1, pp. 16–26, Mar. 2008.

[7] A. H. S. Lai and N. H. C. Yung, "Lane detection by orientation and length discrimination," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 30, no. 4, pp. 539–548, Aug. 2000.

[8] Y. U. Yim and S.-Y. Oh, "Three-feature based automatic lane detection algorithm (TFALDA) for autonomous driving," *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 4, pp. 219–225, Dec. 2003.

[9] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[10] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," in *Proc. Adv. NIPS*, 2006, pp. 1–8.

[11] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proc. 26th Annu. ICML*, 2009, pp. 609–616.

[12] L. Shao, D. Wu, and X. Li, "Learning deep and wide: A spectral method for learning deep networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 12, pp. 2303–2308, Dec. 2014.

[13] G. Hinton *et al.*, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *IEEE Signal Process. Mag.*, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[14] A.-R. Mohamed, G. Dahl, and G. Hinton, "Deep belief networks for phone recognition," in *Proc. NIPS Workshop Deep Learn. Speech Recognit. Rel. Appl.*, 2009, pp. 1–9.

[15] I. Sutskever, J. Martens, and G. E. Hinton, "Generating text with recurrent neural networks," in *Proc. 28th ICML*, 2011, pp. 1017–1024.

[16] H. Goh, N. Thome, M. Cord, and J.-H. Lim, "Learning deep hierarchical visual feature coding," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 12, pp. 2212–2225, Dec. 2014.

[17] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[18] K. He, X. Zhang, S. Ren, and J. Sun. (2015). "Delving deep into rectifiers: Surpassing human-level performance on ImageNet classification." [Online]. Available: http://arxiv.org/abs/1502.01852

[19] A. Stuhlsatz, J. Lippel, and T. Zielke, "Feature extraction with deep neural networks by a generalized discriminant analysis," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 23, no. 4, pp. 596–608, Apr. 2012.

[20] M. Bianchini and F. Scarselli, "On the complexity of neural network classifiers: A comparison between shallow and deep architectures," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 8, pp. 1553–1565, Aug. 2014.

[21] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[22] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, no. 1, pp. 85–117, Jan. 2015.

[23] C. Szegedy *et al.* (2014). "Going deeper with convolutions." [Online]. Available: http://arxiv.org/abs/1409.4842

[24] J. L. Elman, "Finding structure in time," *Cognit. Sci.*, vol. 14, no. 2, pp. 179–211, Mar. 1990.

[25] K. J. Lang, A. H. Waibel, and G. E. Hinton, "A time-delay neural network architecture for isolated word recognition," *Neural Netw.*, vol. 3, no. 1, pp. 23–43, 1990.

[26] L. Szymanski and B. McCane, "Deep networks are effective encoders of periodicity," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 10, pp. 1816–1827, Oct. 2014.

[27] D. Prokhorov, G. Puskorius, and L. Feldkamp, "Dynamical neural networks for control," in *A Field Guide to Dynamical Recurrent Networks*, J. F. Kolen and S. C. Kremer, Eds. New York, NY, USA: Wiley, 2001, pp. 23–78.

[28] D. V. Prokhorov, L. A. Feldkarnp, and I. Y. Tyukin, "Adaptive behavior with fixed weights in RNN: An overview," in *Proc. IJCNN*, 2002, pp. 2018–2022.

[29] L. A. Feldkamp, D. V. Prokhorov, and T. M. Feldkamp, "Simple and conditioned adaptive behavior from Kalman filter trained recurrent networks," *Neural Netw.*, vol. 16, pp. 683–689, Jun./Jul. 2003.

[30] R. J. Williams and D. Zipser, "Gradient-based learning algorithms for recurrent networks and their computational complexity," in *Backpropagation: Theory, Architectures, and Applications*. U.K.: Psychology Press, 1995, pp. 433–486.

[31] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, Oct. 1990.

[32] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997.

[33] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber, "Gradient flow in recurrent nets: The difficulty of learning long-term dependencies," in *A Field Guide to Dynamical Recurrent Neural Networks*, S. C. Kremer and J. F. Kolen, Eds. USA: Wiley, 2001.

[34] A. Graves, S. Fernández, and J. Schmidhuber, "Multi-dimensional recurrent neural networks," in *Proc. 17th Int. Conf. Artif. Neural Netw.*, 2007, pp. 549–558.

[35] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu, "Recurrent models of visual attention," in *Proc. Adv. NIPS*, 2014, pp. 1–9.

[36] A. Geiger, M. Lauer, C. Wojek, C. Stiller, and R. Urtasun, "3D traffic scene understanding from movable platforms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 5, pp. 1012–1025, May 2014.

[37] S. Sivaraman and M. M. Trivedi, "Integrated lane and vehicle detection, localization, and tracking: A synergistic approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 2, pp. 906–917, Jun. 2013.

[38] C. Siagian, C.-K. Chang, and L. Itti, "Mobile robot navigation system in outdoor pedestrian environment using vision-based road recognition," in *Proc. IEEE ICRA*, May 2013, pp. 564–571.

[39] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[40] M. Nieto, A. Cortés, O. Otaegui, J. Arróspide, and L. Salgado, "Real-time lane tracking using Rao–Blackwellized particle filter," *J. Real-Time Image Process.*, vol. 11, no. 1, pp. 179–191, 2012.

[41] A. B. Hillel, R. Lerner, D. Levi, and G. Raz, "Recent progress in road and lane detection: A survey," *Mach. Vis. Appl.*, vol. 25, no. 3, pp. 727–745, 2014.

[42] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.

[43] B. Taskar, V. Chatalbashev, and D. Koller, "Learning associative Markov networks," in *Proc. 21st ICML*, 2004, p. 102.

[44] J. Lafferty, A. McCallum, and F. C. N. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," in *Proc. 18th ICML*, 2001, pp. 1–10.

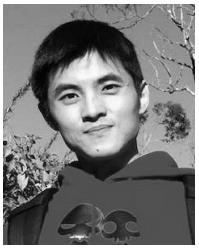
[45] Y. LeCun, S. Chopra, R. M. Hadsell, M. Ranzato, and F. J. Huang, "A tutorial on energy-based learning," in *Predicting Structured Data*, G. Bakir, T. Hofman, B. Scholkopf, A. Smola, and B. Taskar, Eds. USA: MIT Press, 2006, pp. 191–246.

[46] T.-L. Chen, "A Markov random field model for medical image denoising," in *Proc. 2nd Int. Conf. Biomed. Eng. Inform.*, Oct. 2009, pp. 1–6.

[47] A. Graves, "Supervised sequence labelling with recurrent neural networks," Ph.D. dissertation, Univ. Toronto, Canada, 2009.

[48] J. C. McCall and M. M. Trivedi, "Video-based lane estimation and tracking for driver assistance: Survey, system, and evaluation," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 1, pp. 20–37, Mar. 2006.

[49] M. Aly, "Real time detection of lane markers in urban streets," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2008, pp. 7–12.



Jun Li received the B.S. degree in computer science and technology from Shandong University, Jinan, China, in 2003, the M.Sc. degree in information and signal processing from Peking University, Beijing, China, in 2006, and the Ph.D. degree in computer science from the Queen Mary University of London, London, U.K., in 2009.

He is currently a Lecturer with the Centre for Quantum Computation and Information Systems, and the Faculty of Engineering and Information Technology, University of Technology Sydney, Ultimo, NSW, Australia.

Xue Mei (SM'14) received the B.S. degree in electrical engineering from the University of Science and Technology of China, Hefei, China, and the Ph.D. degree in electrical engineering from the University of Maryland, College Park, MD, USA.

He was with the Automation Path-Finding Group in Assembly and Test Technology Development and the Visual Computing Group, Intel Corporation, USA, from 2008 to 2012. He is currently a Senior Research Scientist with the Future Mobility Research Department, Toyota Research Institute, Ann Arbor, MI, USA, a Toyota Technical Center division. He serves as an Adjunct Professor with Anhui University, Hefei. His current research interests include computer vision, machine learning, and robotics with a focus on intelligent vehicles research.

Dr. Mei was an Area Chair of the Winter Conference on Computer Vision in 2015 and 2016, and a Lead Organizer of the My Car Has Eyes: Intelligent Vehicle With Vision Technology Workshop at the Asian Conference on Computer Vision in 2014. He serves as a Lead Guest Editor of the Special Issue on Visual Tracking of *Computer Vision and Image Understanding*.

Danil Prokhorov (SM'02) was a Research Engineer with the St. Petersburg Institute for Informatics and Automation, Russian Academy of Sciences, Saint Petersburg, Russia. He has been involved in automotive research since 1995. He was an Intern with the Scientific Research Laboratory, Ford Motor Company, Dearborn, MI, USA, in 1995. In 1997, he became a Research Staff Member with Ford Motor Company, where he was involved in application-driven research on neural networks and other machine learning methods. Since 2005, he has been with the Toyota Technical Center, Ann Arbor, MI, USA. He is currently in charge of the Department of Future Mobility Research, Toyota Research Institute, Ann Arbor. He has authored over 100 papers in various journals and conference proceedings and holds many patents in a variety of areas.

Dr. Prokhorov served as the International Neural Network Society President from 2013 to 2014, and was a member of the IEEE Intelligent Transportation Systems Society Board of Governors, a U.S. National Science Foundation Expert, and an Associate Editor/Program Committee Member of many international journals and conferences.

Dacheng Tao (F'15) is currently a Professor of Computer Science with the Centre for Quantum Computation and Intelligent Systems, and the Faculty of Engineering and Information Technology, University of Technology Sydney (UTS), Ultimo, NSW, Australia. He mainly applies statistics and mathematics to data analytics problems. His research results have been expounded in one monograph and over 200 publications in prestigious journals and prominent conferences, such as the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, the IEEE TRANSACTIONS ON IMAGE PROCESSING, the *Journal of Machine Learning Research*, the *International Journal of Computer Vision*, the Conference on Neural Information Processing Systems, the International Conference on Machine Learning, the Conference on Computer Vision and Pattern Recognition, the International Conference on Computer Vision, the European Conference on Computer Vision, the International Conference on Artificial Intelligence and Statistics, the International Conference on Data Mining (ICDM), and the ACM Special Interest Group on Knowledge Discovery and Data Mining. His current research interests include computer vision, data science, image processing, machine learning, and video surveillance.

Dr. Tao is a fellow of the Optical Society of America, the International Association for Pattern Recognition, and the International Society for Optics and Photonics. He received several best paper awards, such as the Best Theory/Algorithm Paper Runner Up Award in the IEEE ICDM in 2007, the Best Student Paper Award in the IEEE ICDM in 2013, and the ICDM 10-Year Highest-Impact Paper Award in 2014. He also received the 2015 Australian Scopus-Eureka Prize, the 2015 ACS Gold Disruptor Award, and the UTS Vice Chancellor's Medal for Exceptional Research.