Reliable Multilane Detection and Classification Using a Compact Encoder-Decoder CNN
Reliable multilane detection and classification using a compact encoder-decoder CNN

Shriyash V. Chougule$^{1}$, Asad Ismail$^{2}$, Ganesh Adam$^{3}$, Vikram Narayan$^{2}$ and Matthias Schulze$^{2}$

$^{1}$ Visteon Corporation, Pune, India
$^{2}$ Visteon Corporation, Karlsruhe, Germany

Abstract Reliable lane detection is crucial functionality for autonomous driving. Additionally positional information of ego lanes and side lanes is pivotal for critical tasks like overtaking assistants and path planning. In this work we present a CNN based segmentation approach for detecting multiple lanes as well as positionally classifying them. Wherein we optimize an encoder-decoder architecture for speed and accuracy. We evaluate our network against ENet and SegNet, and found it performing notably better in terms of accuracy. Our network achieves a promising 23 FPS performance on Nvidia's PX2 platform and it has been validated on our test vehicle in highway driving conditions.

Keywords Multilane detection, CNN, Segmentation

1 Introduction

Advanced Driver Assistance Systems (ADAS) is experiencing rapid adoption and growth in automotive industry [1]. A recent study [2] estimated that ADAS functions are effective in reducing head-on and single-vehicle crashes, and minimizing the driver injury risks. Among these safety-critical functionality, capability of autonomous navigation is most sought after function in ADAS. The ability to accurately and reliably detect ego lane and side lanes lie at the core of autonomous driving, and it also serves other critical driving assistance tasks like lane keeping, lane departure warning, and path planning [3].

A vision based lane detection approach provides a low cost solution, but is required to perform robustly in complex driving scenarios. Extracting lane signature from road surface becomes challenging due to
Figure 2.1: Example of multilane detection and classification by our network, where the lane boundaries are classified as 
left side, 
left ego, right ego and right side lane boundaries and are shown by orange, green, red and blue colors respectively.

varying road appearance. Low light situations like sunset, dawn and night add to the challenge. Varying road texture due to shadows casted by trees, vehicles and buildings demands extreme robustness. Other inevitable situations like worn-out lane markings, weakly marked lane boundaries (using periodically placed reflectors) and occlusions due to vehicles is very common driving scenario. Also, due to the road geometry, the side lane signatures are relatively feeble than ego lane in a given image. Thus estimating side lane boundaries is relatively difficult and rarely explored problem. Model based approaches use traditional computer vision techniques to devise specialized hand-crafted features. Such solutions usually works under a controlled environment and are likely to fail in complex driving scenarios. Convolutional neural networks (CNN) have demonstrated a superior performance in pattern recognition. A computer vision approach, devised around a CNN, has the potential to bring about a robust solution to autonomous driving.

In this work, we intent to exploit the CNN's ability of recognizing complex patterns for gaining robustness in challenging driving situations. Towards that direction we investigate architectures of renowned semantic segmentation networks (ENet [4] and SegNet [5]) to assess their potential in segmenting lane boundaries. We observed that these networks have inherent low sensitivity to objects that occupy relatively few pixels in an image (like lane markings and poles). With these acquired insights, an encoder-decoder CNN architecture was chosen for optimization to segment-out lane boundaries as well as classify them positionaly. The network and the dataset used for training is described in Section 3. In Section 4, we compare our network against the ENet and SegNet where we demonstrate that our network is notably more reliable than the rest. Examples of multilane detection and classification by our network are shown in Fig. 2.1
2 Related Work

Recent approaches incorporating CNNs for the lane marker detection [6] have proven to be more robust than model based methods. A CNN used as a preprocessing step in a lane detection system [7], helps in enhancing edge information by noise suppression. DeepLanes [8] detects immediate sideward lane markings with laterally mounted camera system. Although DeepLanes achieve real time performance with high accuracy, it cannot detect road turning ahead. Multilane detection method in [9] makes use of CNN and regression to identify line segments that approximate a lane boundary effectively, it requires high resolution images to work with which hampers the speed. SegNet [5] based multilane detection network [10] segment out lanes, though promising the segmented mask are not accurate at road turnings. VPGNet [11] detects and classify lane markings along with road informative markings, it is inspired by the human intuition of identifying lane layout from global context like road structure and traffic flow. VPGNet trains a baseline network for task of vanishing point detection and further fine tune it for lane and road marking detection task, which helps to improve the overall accuracy. To improve segmentation accuracy of thin and elongated objects like poles and lanes boundaries, Spatial CNN (SCNN) [12] replaces the conventional layer-by-layer convolution with slice-by-slice convolution within feature maps. Such an arrangement provides information flow between pixels across rows and columns, which is hypothesized to be effective for summarizing the global context. A GAN framework is utilized in [13] for lane boundary segmentation, where the discriminator network is trained using an "embedding loss" function which iteratively helps the generator network in learning of higher structural semantics of lanes. LaneNet [14] is a instance segmentation network which makes use of a branched structure to output binary lane segmentation mask and pixel localization mask, which is further used to infer lane instance by clustering process. Apart from LaneNet, a separate network is trained for obtaining parametric lane information from lane instance segmentation mask.

In this work we optimize an encoder-decoder architecture [15], along with borrowing design choices from SCNN and ENet to improve accuracy and speed respectively.
3 Multilane Detection and Classification Network

Lane markings and poles have peculiar shape of being an elongated and thin objects, usually occupying less than 5% of the total pixels in a driving scene image. We analyze two prominent semantic segmentation networks like SegNet [5] and ENet [4], to evaluate their potential for segmenting lane boundaries. We summarize their performance on Camvid dataset, which contain examples of an urban driving scenario. As can be seen in Fig. 2.2 (b), the segmentation accuracy for lane markings (road markings) class and pole class is low in comparison with remaining classes. As a consequence of poor segmentation accuracy, the segmented lane markings appear fragmented due to many false positive and false negative as shown in Fig. 2.2 (a). Inferring lane boundaries from these fragmented lane markings demands designing of a tedious post-processing stage, followed by the use of a tracking framework. Fragmentation also hinders the prospect of estimating turns and curvature of the road ahead, which is a key information for critical driving assistance tasks (such as path planning and lane keeping). In the following subsections we elaborate our network design decisions to boost the segmentation accuracy of lane boundaries and minimize their fragmentation.
3.1 Use of encoder-decoder architecture

Use of gradually graded deconvolution layers instead of a single deconvolution layer is known to construct denser and much precise segmentation mask [16]. Thus the segmentation accuracy of lane boundaries can greatly benefit by these graded deconvolutional layers. So we choose to optimize a symmetrical encoder-decoder architecture where the encoder section (or convolutional network) is a sequence of convolutional layers, followed by the decoder section (or deconvolutional network) involving deconvolutional layers. The encoder section behaves as a feature extractor whereas the decoder section constructs a segmentation map from the extracted features. We reduce the original 39 layers encoder-decoder network [15] to 10 layers network i.e. 5 convolutional and 5 deconvolution layers (see Fig. 2.3). Such compact structure of encoder-decoder network significantly cut down the computations, and yet is observed to be sufficiently accurate for segmenting lane boundaries as demonstrated in Section 4. Also we add interpolation layers [17] at appropriate positions, which helps in resizing the feature map to match input dimensions of a successive layer. We also implement the SCNN_D and SCNN_L layer of Spatial-CNN [12] as explained in Section 3.3, in order to improve segmentation accuracy.

3.2 No pooling operations

It is well known in the computer vision domain that for any given method, lowering the input image resolution provides execution speed
up at the cost of accuracy. ENet leverages this fact and achieves impressive speedup by employing a strategy of early down sampling, done by its first two bottleneck layers. Instead of going for early down sampling strategy, we choose to work directly with a low resolution input image to uplift the execution speed. And to counter the accuracy loss because of lowering the input image resolution, we choose to remove the max pooling layers and corresponding unpooling layers from the original deconvolution network. Max-pooling is ubiquitous procedure used to retain most active neurons, which translates in decreased features dimensionality as well as introduces some degree of translation invariance. However, in our problem we found that loosing spatial information by using max pooling layer is indeed not desirable. Without pooling layers our network produces much finer segmentation masks, and saves the computation time as well.

3.3 Spatial information flow

We implement the SCNN_D and SCNN_I, spatial information flow modules/layers of SCNN to further enhance the segmentation accuracy. As described in the original work, the SCNN_D layer establishes an information flow from top to bottom within a feature map, whereas SCNN_I establishes information flow from left to right. Other two spatial information modules/layers (SCNN_U and SCNN_R) were ignored in order to save computations. All the modules use a general strategy to slice the 3D feature map along the information flow directions under consideration. These feature slices are used to generate a new set of feature slices, using successive convolution and element-wise addition operations given by below equation:

\[
\text{slice}_{i}^{\text{new}} = \begin{cases} 
\text{slice}_{i}^{\text{old}}, & \text{if } i = 1 \\
\text{slice}_{i}^{\text{old}} + f(\text{convolution}(\text{slice}_{i-1}^{\text{new}})) & \text{otherwise}
\end{cases}
\]

(2.1)

where \( f \) is nonlinear activation function like ReLU, \( i \) indicates the slice sequence observed w.r.t. the information flow direction. The new slices are concatenated along the information flow direction to create a new 3D feature map. Network implementation details along with the convolution operation in above equation is given in the Table 2.1.
### Table 2.1: Network description (Caffe deep learning framework context).

<table>
<thead>
<tr>
<th>Layer parameters</th>
<th>Encoder</th>
</tr>
</thead>
<tbody>
<tr>
<td>conv{output, padding, kernel, group, stride}</td>
<td></td>
</tr>
<tr>
<td>deconv{output, padding, kernel, group, stride}</td>
<td></td>
</tr>
<tr>
<td>interpolation{height, width}</td>
<td></td>
</tr>
<tr>
<td>scan{direction, conv{output, padding, kernel, group, stride}}</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Decoder</td>
</tr>
<tr>
<td>deconv{64, 1, 3, 2, 1}</td>
<td></td>
</tr>
<tr>
<td>scan{top to bottom, conv{96, 1, 3, 1, 1}}</td>
<td></td>
</tr>
<tr>
<td>scan{left to right, conv{96, 1, 3, 1, 1}}</td>
<td></td>
</tr>
<tr>
<td>deconv{128, 1, 3, 2, 2}</td>
<td></td>
</tr>
<tr>
<td>interpolation{23, 40}</td>
<td></td>
</tr>
<tr>
<td>deconv{96, 1, 3, 2, 1}</td>
<td></td>
</tr>
<tr>
<td>interpolation{89, 157}</td>
<td></td>
</tr>
<tr>
<td>deconv{16, 1, 3, 1, 1}</td>
<td></td>
</tr>
<tr>
<td>interpolation{184, 320}</td>
<td></td>
</tr>
<tr>
<td>deconv{5, 1, 3, 1, 1}</td>
<td></td>
</tr>
</tbody>
</table>

### 3.4 Dataset generation

To train our network for lane segmentation task, we gathered a dataset containing 33000 images of highway driving scenarios and manually annotated the lane boundaries. Images from TuSimple lane detection challenge dataset were included as well, after adapting its ground truth by fitting curves on the list of lane boundary points. An image in the dataset may have 1 to 4 annotated lane boundaries depending upon the scene. The ground truth information for each lane boundary is represented by a gray scale image where the lane boundary pixels are given distinct integer values which act as labels to the four lane types i.e. rightego, leftego, rightside and leftside and the background. All images are of
Figure 2.4: Illustration of MIoU metric components, where the background, leftego boundary and rightego boundary have class id as 0, 1 and 2 respectively.

184 x 320 resolution which were recorded in normal weather conditions, during day and evening time.

3.5 Network training

Caffe deeplearning framework was used for training our network as well as for ENet and SegNet training. The dataset was split into 32000 training and 1000 testing images. On an average, out of the total pixels in a training image, the background label accounted for 97.25% pixels whereas leftside, leftego, rightego and rightside labels accounted for 0.59%, 0.91%, 0.82% and 0.43% pixels respectively. Because of the uneven sample counts, a weighted softmax loss function was used in training all the networks where the class loss weights were based on label percentage i.e. (1 – percentage label). All the networks were trained till convergence to 92% training accuracy using ADAM optimizer.

<table>
<thead>
<tr>
<th>Network</th>
<th>MIoU</th>
</tr>
</thead>
<tbody>
<tr>
<td>SegNet</td>
<td>52.48</td>
</tr>
<tr>
<td>ENet</td>
<td>64.23</td>
</tr>
<tr>
<td><strong>Ours</strong></td>
<td><strong>76.39</strong></td>
</tr>
</tbody>
</table>

Table 2.2: Lane detection accuracy measured using MIoU metric.
4 Evaluation

Most common metrics for evaluating segmentation accuracy are based on similarity measures between predicted and ground truth pixels. The predictions of all the networks were obtained on test dataset and then compared to the ground truth (grayscale images) using the MIoU (mean intersection over union) metric. The MIoU metric for a single test image is defined as follows:

\[
MIoU = \frac{1}{1 + k} \sum_{i=0}^{k} \frac{TP_{ii}}{\sum_{j=0}^{k} FN_{ij} + \sum_{j=0}^{k} FP_{ji} - TP_{ii}}
\]  \hspace{1cm} (2.2)

where \(k\) is number of classes (\(k=5\) in our case i.e right\_ego, left\_ego, rights\_ide, left\_side and background). \(TP, FN\) and \(FP\) are pixel counts in the true positive, false negative and false positive regions respectively as illustrated in Fig. 2.4. Performance of networks is summarized in Table 2.2.
Figure 2.6: Scenarios where network performance dropped.

The predictions on challenging scenarios is shown in Fig. 2.5 for visual comparison, where it can be observed that our network predictions are relatively more accurate than that of ENet and SegNet. Particularly our network is more reliable in capturing the road curvature at turnings ahead, during occlusions due to vehicles, and at approaching road split where ENet tends to confuse. Fig. 2.6 shows cases where our network performance dropped. It includes case where over-bridge shadow is covering more than 50% of scene, multiple lane markings, and when the vehicle is changing lane. Our network has been validated on our test vehicle using Nvidia’s PX2 platform where a consistent and stable detection was observed, with a promising performance of 23 FPS.

5 Conclusions

We presented a CNN based segmentation approach for detecting multiple lane boundaries and classifying them based on position. Our network demonstrated reliable performance in challenging driving scenarios like lane occlusions due to vehicle, poorly lane marked roads and in low evening light, when compared to ENet and SegNet. Also it accurately captured the road turnings compared to the rest. A performance drop was observed during lane changing, which we intend to address either by tracking or by use of feedback framework analogous to RNN. We also intend to modify our network to directly output parametric lane boundaries, which is a more convenient format to work with.

References


