DeepSignals: Predicting Intent of Drivers Through Visual Signals

Davi Frossard[1,2] Eric Kee[1] Raquel Urtasun[1,2]

***Abstract*— Detecting the intention of drivers is an essential task in self-driving, necessary to anticipate sudden events like lane changes and stops. Turn signals and emergency flashers communicate such intentions, providing seconds of potentially critical reaction time. In this paper, we propose to detect these signals in video sequences by using a deep neural network that reasons about both spatial and temporal information. Our experiments on more than a million frames show high per-frame accuracy in very challenging scenarios.**

I. Introduction

Autonomous driving has risen as one of the most impactful applications of Artificial Intelligence (AI), where it has the potential to change the way we live. Before self-driving cars are the norm however, humans and robots will have to share the roads. In this shared scenario, communications between vehicles are critical to alert others of maneuvers that would otherwise be sudden or dangerous. A social understanding of human intent is therefore essential to the progress of self-driving. This poses additional complexity for self-driving systems, as such interactions are generally difficult to learn.

Drivers communicate their intent to make unexpected maneuvers in order to give warning much further in advance than would otherwise be possible to infer from motion. Although driver movements communicate intent—for example when drivers slow down to indicate that they will allow a merge, or drive close to a lane boundary to indicate a desired merge position—motion cues are subtle, context dependent, and near-term. In contrast, visual signals, and in particular signal lights, are unambiguous and can be given far in advance to warn of unexpected maneuvers.

For example, without detecting a turn signal, a parked car may appear equally likely to remain parked as it is to pull into oncoming traffic. Analogously, when a driver plans to cut in front of another vehicle, they will generally signal in advance for safety. Buses also signal with flashers when making a stop to pick up and drop off passengers, allowing vehicles approaching from behind to change lanes, therefore reducing delays and congestion.

These everyday behaviors are safe when drivers understand the intentions of their peers, but are dangerous if visual signals are ignored. Humans expect self-driving vehicles to respond. We therefore consider in this work the problem of predicting driver intent through visual signals, and focus specifically on interpreting signal lights.

Estimating the state of turn signals is a difficult problem: The visual evidence is small (typically only a few pixels), particularly at range, and occlusions are frequent. In addition, intra-class variations can be large. While some regulation exists, many vehicles have stylized blinkers, such as light bars with sequential lights in the direction being signaled, and the regulated frequency of blinking (1.5 ± 0.5 Hz [1]) is not always followed. Furthermore, since we are interested in estimating intent, vehicle pose needs to be decoded. For instance, a left turn signal would correspond to a flashing light on the left side of a vehicle we are following, but on the other hand would correspond to a flashing light on the right side of an incoming vehicle. We refer the reader to Figure 2 for an illustration of some of the challenges of turn signal estimation.

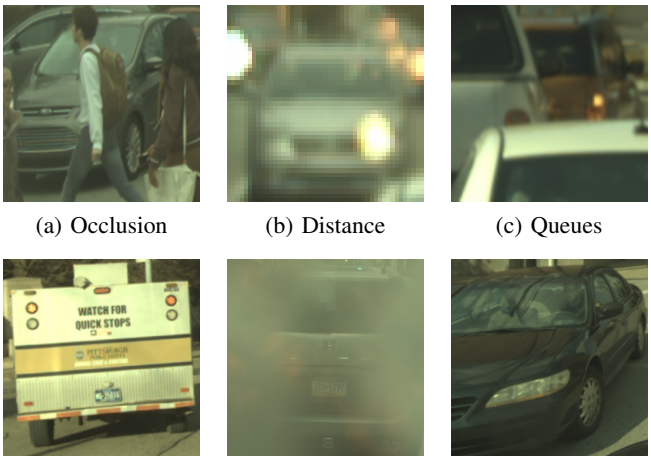
Fig. 1: A vehicle, signaling left, passes through occlusion. The actor's intent to turn left is correctly detected (left arrow), including the occlusion (question mark).

Surprisingly little work in the literature has considered this problem. Earlier published works [2], [3] use hand-engineered features, trained in-part on synthetic data, and are evaluated on limited datasets. Other approaches have considered only nighttime scenarios [4], [5]. Such methods are unlikely to generalize to the diversity of driving scenarios that are encountered every day.

In this paper, we identify visual signal detection as an important problem in self-driving. We introduce a large-scale dataset of vehicle signals, and propose a modern deep learning approach to directly estimate turn signal states from diverse, real-world video sequences. A principled network is designed to model the subproblems of turn signal detection: attention, scene understanding, and temporal signal detection. This results in a differentiable system that can be trained end-to-end using deep learning techniques, rather than relying upon hard coded premises of how turn signals should behave.

We demonstrate the effectiveness of our approach on a new, challenging real-world dataset comprising 34 hours of

[1]Uber Advanced Technologies Group
[2]University of Toronto
{frossard, ekee, urtasun}@uber.com



(a) Occlusion (b) Distance (c) Queues

(d) Unusual (e) Weather (f) Bad Detections

Fig. 2: Challenging scenarios from the dataset of 1,257,591 labeled frames.

video from our self-driving platform. The dataset includes the adverse conditions found in real-world urban driving scenarios, including occlusion, distant and uncommon vehicles, bad weather, and night/daytime exposures (see Figure 2 for an illustration).

II. Related Work

The field of visual perception for autonomous vehicles has been intensely explored in the past few years. Deep learning, and in particular deep convolutional neural networks (CNNs), has shown great improvements for tasks such as detection [6], [7], [8], [9], tracking [10], [11], instance and semantic segmentation [12], [13], [14], [15], stereo estimation [16], [17], [18], and (more related to the problem of action prediction) traffic understanding [19], [20], [21].

Little published work has been devoted to estimate the state of turn signals. Frohlich et al. [2] proposed to detect light spots in vehicle bounding boxes and extract features using FFT over time, followed by an Adaboost classifier to predict the state of the turn signal. Another method [3] uses turn signals as a feature for vehicle detection and tracking. The state of the turn signals is also classified via a logic on the change of luminance on either side of the vehicle. Hsu et al. [22] propose the use of SIFT flow to align a sequence of images as a pre-processing step to compute a difference image, which is then used by a CNN-LSTM architecture to predict both turn signal state as well as brake lights. These methods however use manually engineered features that do not adapt to different viewpoints of the vehicles (front, left, right), and testing considers a limited set of vehicles.

Another line of work employs turn signals as a feature for vehicle detection. In [4], a Nakagami image is used to locate regions containing vehicular lights, which used as proposals for a Fast R-CNN detector. Other works use turn signals as a Haar-like feature for training Adaboost classifiers [5].

In contrast, our work focuses on a fully learned approach to classify the state of turn signals using spatial and temporal features. This is closely related to recent works in the field of action recognition in videos, which has shown great progress with the use of deep learning. Among recent works, deep 3D convolutional neural networks for human action recognition in videos [23] show the effectiveness of convolutions in the time dimension as a way to extract temporal features. Other works have considered decoupling spatial and temporal features [24], using two CNNs to perform action recognition: one processing single images for spatial features, and another using dense optical flow for temporal features.

Recurrent Neural Networks (RNNs) are an effective mechanism for temporal feature extraction. Recent works use LSTMs to learn social interactions between humans as a feature for tracking [25] and activity recognition [26] from videos. RNN-based agents have been trained with reinforcement learning in order to decide both how to tranverse the video and when to emit a prediction [27].

The union between CNNs and LSTMs has also been explored, with the use of CNNs to extract features from individual frames followed by LSTMs to incorporate temporal features [28], [29]. This approach can be extended by using 3D CNNs to extract local spatio-temporal features and a LSTM for long term temporal features [30]. Going a step further, CNNs and LSTMs can be fused into a single architecture by using convolutional gates, an approach first introduced in order to predict precipitation from radar intensity images [31], but that has been recently used to create online maps for self driving vehicles [32].

While initially introduced for machine translation [33], attention mechanisms have also been for neural captioning [34] and action recognition [35]. In [36], a Bidirectional LSTM is used to both detect events in videos and attend to the agents causing the event. ConvLSTMs with attention have also been used for action classification [37].

III. Predicting Actor Intent Through Visual Signals

In this paper, we tackle the problem of estimating actor intent from the state of turn signals and emergency lights. These visual signals are naturally captured in video on autonomous driving platforms. A typical video scene can contain multiple vehicles, which presents the problem of detection and tracking. Here it is assumed that such methods are available, and a preprocessing step will be applied to recover an axis-aligned region of interest (ROI) around each actor.

The ROI around an actor constitutes a streaming input of cropped video frames. The desired system output is also a stream, indicating the state of the actor's signals at each frame. Classifying turn signal states is a challenging problem, as the left turn signal of a vehicle can appear on either the image's left or right depending upon the vehicle orientation. The network therefore must also resolve the pose of the actor to correctly classify turn signal state regardless of view face.

Within any particular video frame, a signal light may be illuminated or not, while its logical state is `ON`, `OFF`, or `UNKNOWN` (the latter representing when the light is occluded). Given the logical state of both signal lights, the

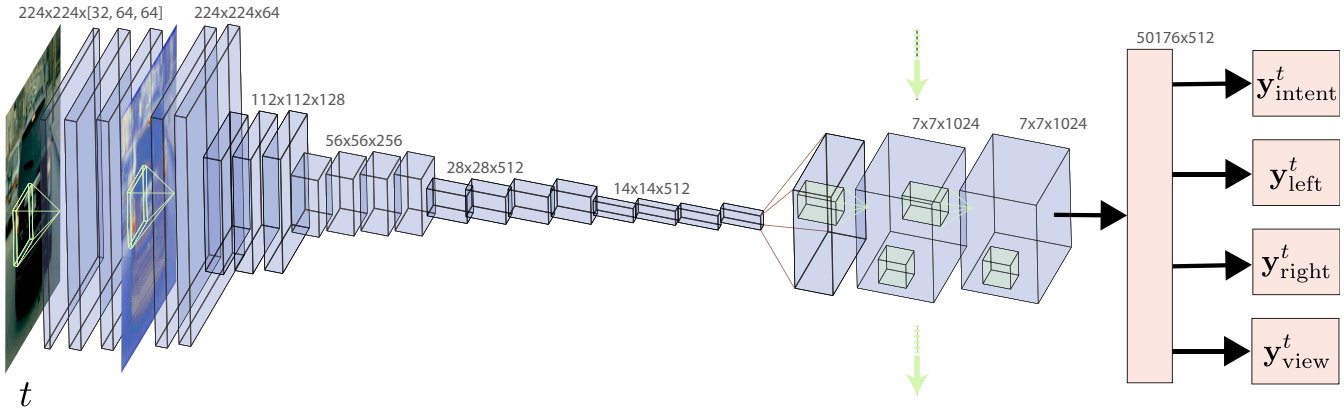


Fig. 3: In this work, we propose the use of a convolutional-recurrent architecture in order to classify the state of turn signals in vehicles. For each input frame, an attention mask is predicted using a fully convolutional network (a), the element-wise product is then taken with the original input image and spatial features are extracted using a VGG16-based CNN (b), temporal features are then incorporated using a Convolutional LSTM (c). From the final hidden state, probability distributions are predicted over the state of turn signals and the view face of the vehicle (d).

logical state of the turn signal can be determined: `LEFT TURN`, `RIGHT TURN`, `FLASHERS`, `OFF`, or `UNKNOWN`. For example, observing (`ON`, `OFF`) for the left and right signal lights indicates a `LEFT TURN`. Note that (`ON`, `UNKNOWN`) can be mapped to either `LEFT TURN` or `UNKNOWN` depending upon how much confidence is desired. In this section, we describe a learned model to predict such turn signal states.

A. Model formulation

The model is formulated to address three subproblems: *attention,* to identify the signal lights of the actor; *semantic understanding*, to identify occlusion and the direction from which the actor is being viewed; and *temporal reasoning*, to distinguish flashing lights and persistent lights from other specious patterns. A deep learning architecture is designed to address these problems. We refer the reader to Figure 3 for an illustration. Input frames are first processed by an attention module, which applies a spatial mask, and a deep convolutional network is used to recover spatial concepts. This per-frame information is then input to a convolutional LSTM to distinguish the temporal patterns of turn signals and emergency flashers from other content. The resulting spatial and temporal features are passed to fully connected layers for classification.

Attention processing begins by resizing the input ROI to a fixed 224×224 pixels. A 4-layer, fully convolutional network is used to compute a pixel-wise, scalar attention value. Kernels are 3×3 with dilations $(1, 2, 2, 1)$ and channel dimensions $(32, 64, 64, 1)$. The resulting scalar mask is point-wise multiplied with the original, resized input ROI. This allows the network to both add more saliency to relevant pixels and attenuate noisy spatial artifacts.

Spatial features are extracted using a CNN architecture based on VGG16 [38], as shown in Figure 3 (b). Weights are pre-trained on ImageNet, and fine tuned during training. This allows the network to model the vehicle of interest, its orientation, occluding objects, and other spatial concepts. The $7 \times 7 \times 512$ output retains a coarse spatial dimension for temporal processing by a Convolutional LSTM.

A Convolutional LSTM (ConvLSTM) module [31] is used to refine the spatial features by modeling temporal characteristics of the streaming input (now a stream of feature tensors), depicted in Figure 3 (c). Note that this design factors the spatial and temporal components of the model into separate modules. We show that factorization more efficiently uses the available compute resources, and leads to greater performance. The ConvLSTM allows the network to reason through time, and thus distinguish between flashing and persistent lights (i.e., turn signals and brake lights).

Convolutional LSTMs learn temporal representations by maintaining an internal, hidden state, which is modified through a series of control gates. Let $\mathbf{X}_t$ be the feature tensor that is input at time t, and $\mathbf{W}$ and $\mathbf{B}$ be the learned weights and biases of the ConvLSTM. The hidden state is embodied by two tensors, $\mathbf{H}$ and $\mathbf{C}$, which are updated over time by the following expressions:

$$\mathbf{I}_t = \sigma\big(\mathbf{W}^{xi} * \mathbf{X}_t + \mathbf{W}^{hi} * \mathbf{H}_{t\text{-}1} + \mathbf{W}^{ci} * \mathbf{C}_{t\text{-}1} + \mathbf{B}^i\big) \quad (1)$$

$$\mathbf{F}_t = \sigma\big(\mathbf{W}^{xf} * \mathbf{X}_t + \mathbf{W}^{hf} * \mathbf{H}_{t\text{-}1} + \mathbf{W}^{cf} * \mathbf{C}_{t\text{-}1} + \mathbf{B}^f\big) \quad (2)$$

$$\mathbf{C}_t = \mathbf{F}_t \circ \mathbf{C}_{t\text{-}1} + \ldots \\ \mathbf{I}_t \circ \tanh\big(\mathbf{W}^{xc} * \mathbf{X}_t + \mathbf{W}^{hc} * \mathbf{H}_{t\text{-}1} + \mathbf{B}^c\big) \quad (3)$$

$$\mathbf{O}_t = \sigma\big(\mathbf{W}^{xo} * \mathbf{X}_t + \mathbf{W}^{ho} * \mathbf{H}_{t\text{-}1} + \mathbf{W}^{co} * \mathbf{C}_t + \mathbf{B}^o\big) \quad (4)$$

$$\mathbf{H}_t = \mathbf{O}_t \circ \tanh\big(\mathbf{C}_t\big) \quad . \quad (5)$$

The parameterized gates $\mathbf{I}$ (input), $\mathbf{F}$ (forget) and $\mathbf{O}$ (output) control the flow of information through the network, and how much of it should be propagated in time. Temporal information is maintained through the cell memory, which accumulates relevant latent representations, as shown in

Equation (3). Note that, to prevent overfitting, we apply dropout on the output of Equation (3) as a regularizer. Specifically, the input gate controls the use of new information from the input, Equation (1); the forget gate controls what information is discarded from the previous cell state, Equation (2); and the output gate further controls the propagation of information from the current cell state to the output, Equation (4) by element-wise multiplication, Equation (5).

The ConvLSTM module is constructed as a series of ConvLSTM layers, each following Equations (1)-(5). In this multi-layer architecture, each subsequent layer takes as input the hidden state, $\mathbf{H}_t$, from the preceding layer (the first layer takes $\mathbf{X}_t$ as input). In particular, we use two ConvLSTM layers, each with a $7 \times 7 \times 256$ hidden state.

Lastly, the ConvLSTM features are passed through a fully connected layer, depicted in Figure 3 (d). This produces the random variables of interest: $\mathbf{y}^t_{\text{intent}}$ over the states `LEFT TURN`, `RIGHT TURN`, `FLASHERS`, `OFF` and `UNKNOWN`; $\mathbf{y}^t_{\text{left}}$ and $\mathbf{y}^t_{\text{right}}$ over the states `ON`, `OFF`, `UNKNOWN` (for the individual lights on the left and right sides of the vehicle, respectively). We also define states from which an actor is being viewed `BEHIND`, `LEFT`, `FRONT`, `RIGHT`, and show that predicting these as $\mathbf{y}^t_{\text{view}}$ helps learning.

B. Learning

We train our model using a multi-task loss. Specifically, a weighted cross entropy loss is defined over the tasks. Given model inputs x, ground truth labels $\hat{y}$, model weights θ, task weights γ and the network function $\sigma(\cdot)$, the loss is

$$\begin{aligned}\mathcal{L}(\hat{y}, x|\theta) = \ell_{intent}(\hat{y}, x|\theta) \\ + \ell_{left}(\hat{y}, x|\theta) + \ell_{right}(\hat{y}, x|\theta) \\ + \ell_{view}(\hat{y}, x|\theta)\end{aligned} \tag{6}$$

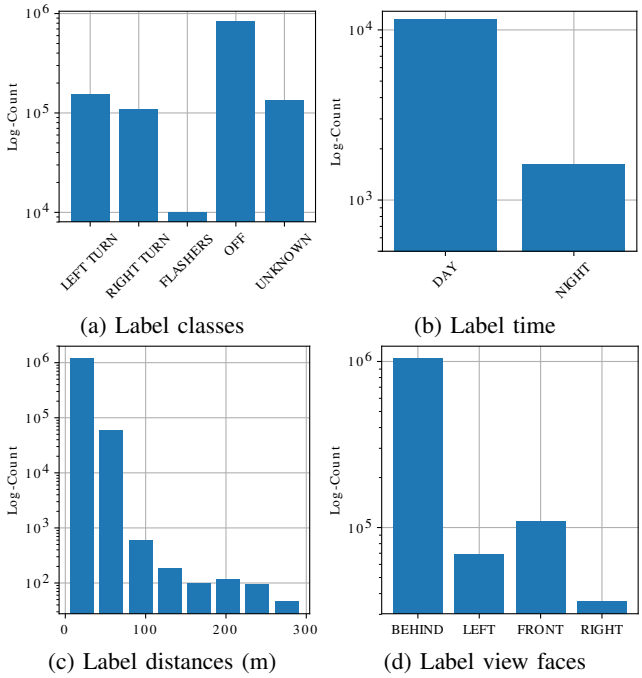
where each task loss uses cross-entropy defined as

$$\ell(\hat{y}, x|\theta) = \gamma \sum_c \hat{y}_c \log(\sigma_c(x|\theta))$$

Note that the loss is defined in terms of a sum over the task space, which includes: ℓ_{intent}, the loss over the high level understanding of the actor; ℓ_{left} and ℓ_{right}, the losses over the left and right turn signals, respectively; and ℓ_{view}, the loss over the face of the actor that is seen. Furthermore, we also employ L2 weight decay on the fully connected layers to prevent overfitting.

IV. Experimental Evaluation

a) Dataset: Unfortunately, there is no public dataset for turn signal classification. Therefore, we label over 10,000 vehicle trajectories recorded in our autonomous driving platform at 10 Hz in terms of the state of turn signals, for a total of 1,257,591 labeled frames. Each frame is labeled for the left turn and right turn lights in terms of `ON`, `OFF` or `UNKNOWN`. Note that the label identifies the conceptual state of each light, with `ON` indicating that the signal is active even when the light bulb is not illuminated. These lower level labels are used to infer the high level action intents: `LEFT TURN`, `RIGHT TURN`, `FLASHERS`, `OFF` and `UNKNOWN`, which ultimately is what the model is trained to classify. The number of labels is shown in Figure 4a and it evidences a considerable bias towards the `OFF` class in the dataset. Also shown are the distributions over distance (Figure 4c) and viewpoint (Figure 4d).



(a) Label classes (b) Label time

(c) Label distances (m) (d) Label view faces

Fig. 4: Dataset distributions in log units. The distribution over labels (a) is imbalanced towards `OFF`, with `FLASHERS` being least common. Number of sequences during daytime (b) is almost an order of magnitude larger than those during nighttime. Distance to vehicles (c) place 90% within 30 meters. Lastly, the majority of vehicles are seen from behind (d), with right being relatively under-represented.

Model	Accuracy	Recall	F1	Fp	Fn
FC-LSTM	35.30%	30.47%	32.71%	27.70%	61.65%
ConvLSTM	37.32%	34.42%	35.81%	30.07%	63.95%
CNN-LSTM	60.52%	60.54%	60.53%	11.21%	39.17%
ours	**70.89%**	**72.11%**	**71.49%**	**5.63%**	**24.00%**

TABLE I: Comparison to baseline systems.

b) Experimental Setup: To train the models, we use Adam optimization [39] with a learning rate of 1×10^{-4}, $\beta_1 = 0.9$, and $\beta_2 = 0.999$. We also reduce the learning rate on plateau, multiplying it by a factor of 0.1 if 5 epochs go by without changing the loss by more than 1×10^{-3}. A weight decay of 1×10^{-4} and dropout with $p = 0.5$ is used in the fully connected layers for regularization. Training mini-batches are sampled using a stratified scheme that counteracts class imbalance. Training is limited to 50 epochs (models will generally converge around the 25th epoch) and selection is done according to the validation metrics. Lastly, data augmentation via random mirroring and color jittering is applied to the input sequences.

	LEFT	RIGHT	FLASHERS	OFF	UNKNOWN
LEFT	74%	1%	0%	22%	2%
RIGHT	1%	77%	0%	18%	4%
EMERGENCY	9%	17%	36%	26%	13%
OFF	3%	2%	0%	92%	3%
UNKNOWN	3%	1%	1%	23%	72%

TABLE II: Confusion matrix for our proposed model.

c) Baselines: We evaluate our proposed method against a series of baselines. In particular, it is compared against a fully connected LSTM (FC-LSTM), a Convolutional LSTM (ConvLSTM), and an LSTM using convolutional features (CNN-LSTM). In all cases, we use sequences of vehicle observations at size 224×224 pixels. For the FC-LSTM, the sequences are flattened and passed through an FC-LSTM with 3 hidden layers (with 256, 256 and 512 neurons, respectively). For the ConvLSTM, we also use a 3 layer network (with 8, 8 and 3 channels, respectively (deeper models could not be fit in the GPU). Lastly, for the CNN-LSTM, features are first extracted using VGG16, flattened and fed to a two layer LSTM with 256 and 128 neurons.

d) Metrics: To evaluate the model, we use accuracy, recall, and F1 metrics. We also define a False Positive (FP) metric as being the cases in which a ground-truth `OFF` or `UNKNOWN` is classified as any other state, and a False Negative (FN) metric when a ground-truth `ON` is classified as any other state.

The aforementioned metrics are shown in Table I for the FC-LSTM, ConvLSTM, CNN-CLSTM and our proposed method. The FC-LSTM results in the weakest performance; this can be explained by the ineffectiveness of fully connected layers in extracting spatial features, counterbalanced only by the large capacity of the network, which allows it to learn more complex functions. The ConvLSTM achieves slightly better results by leveraging convolutions in the gates, which makes it more suitable for spatial feature extraction. Its memory inefficiency, however, prevents us from using deeper architectures and therefore limits the capacity of the model. Combining the two approaches we arrive at a CNN-LSTM, which is both able to leverage the rich spatial feature extraction from CNNs and temporal feature representation of LSTMs, achieving better results than the previous baselines. Our proposed method further adds convolutions inside the LSTM and attention mechanisms, giving the best results.

The confusion matrix of our approach is shown in Table II. Note that detection accuracy is distinctly high for `OFF` and low for `FLASHERS` because the two classes are over- and under-represented (respectively) in the dataset, Figure 4d (a). This imbalance naturally affects test performance, even when using a stratified sampling scheme, because the label distributions are similar between train and test sets.

Lastly, we show several outputs from our model in Figure 1 and Figure 5. These examples illustrate that turn signals can be classified from various viewpoints, and that occlusion can be detected. Furthermore, the attention network produces masks with highest value around bright spots, which makes turn signal regions more salient, improving performance (see Section IV-A). In Figure 6, we show examples of frames in which the network fails, and provide hypotheses for the cause. Failures can result from distracting lights (or reflections), and unusual vehicle shapes or features. Examples are also included as video in the supplementary material.

A. Ablation Studies

Here we evaluate the effects of diverse changes to the model: using real detections, changing the attention architectures, output parameterization and normalization schemes. Results for the experiments are shown in Tables III-IV.

INPUT	ACCURACY	RECALL	F1	FP	FN
Detections	66.85%	62.96%	64.85%	7.42%	26.25%
Labels	**70.89%**	**72.11%**	**71.49%**	**5.63%**	**24.00%**

TABLE III: Comparison between labels and detections as input for the model (numbers reported on held-out test set).

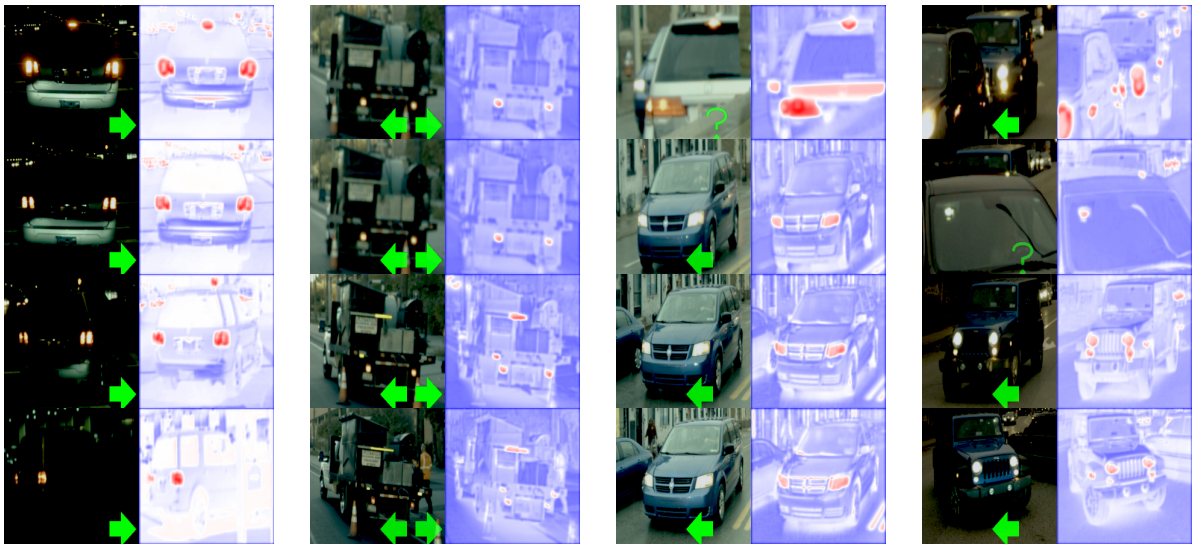
a) Real Detections: We first compare how performance changes when using input crops coming from a detector as opposed to labeled bounding boxes. In particular, a lidar-based CNN is used to detect vehicles, and project the boxes into the image to crop the input for the network. Results are shown in Table III, and indicate that while performance levels are lower when using detections, the model is able to cope with imperfect ROIs and produce comparable results.

b) Attention: The impact of different attention mechanisms is considered in Table IV rows 1-2. Specifically, we consider a model using no attention, and a u-Net [40] based CNN for attention (marked with a $*$).[1] Results show that attention increases recall by 5.48 percentage points. The u-Net architecture did not outperform the fully convolutional approach, which can be explained by the fine positioning that the latter can achieve, emphasizing only important pixels.

c) Loss Parameterization: We show the effects of using different loss parameterizations on the model, Table IV rows 3-4, which affects the output produced by the model. Specifically, we consider a single task loss over the intent states, (in which case the loss function is just the ℓ_{intent} component), and removing supervision for the left and right sides (therefore using ℓ_{intent} and ℓ_{view} as the loss components). Results show multi-task loss helps learning, with extra supervision improving as much as 5.23 percentage points in F1 between our model and its single-task counterpart.

d) Normalization: Lastly, we study the effects of adding normalization schemes on the model, Table IV rows 5-6. In particular, we experiment with using layer normalization (LN) and batch normalization (BN). These methods differ in terms of the dimensions in which normalization happens: layer normalization normalizes across the channel dimensions only, while batch normalization normalizes across the batch and spatial dimensions [41]. The results show that these methods are not beneficial for the model, which makes intuitive sense as BatchNorm is not suitable for the small batch sizes we use. One hypothesis as to why

[1]The network consists of two, 2-layer convolutional blocks: 32 and 64 channels respectively, both using kernel size 3×3, and no dilation.



(a) (b) (c) (d)

Fig. 5: Several sequences from the test set. Input image is shown on the left with the network output illustrated on the bottom right, the corresponding attention mask is shown on the right. Column (a), right turn signal is correct before and throughout the maneuver. Columns (b), vehicle with flashers stopped on the sidewalk is correctly classified. Columns (c) and (d) show challenging sequences with correct classification of incoming vehicles signaling left turns (including occlusion).

(a) (b) (c) (d) (e)

Fig. 6: Failure modes of the network. (a) Bright lights at night are misclassified as a left turn. (b) Bright reflection on the right side of a distant vehicle is misclassified as a right turn. (c) Right turn signal is missed on an unusual vehicle. (d) Actor pose is incorrectly decoded and output is flipped. (e) False positive left turn on a vehicle carrying a bike.

	ATTENTION	LOSS			BN	LN	ACCURACY	RECALL	F1
		ℓ_{intent}	ℓ_{left}/ℓ_{right}	ℓ_{view}					
1	—	✓	✓	✓	—	—	66.03%	66.63%	66.33%
2	✓*	✓	✓	✓	—	—	66.82%	70.84%	68.77%
3	✓	✓	—	—	—	—	63.77%	68.95%	66.26%
4	✓	✓	—	✓	—	—	65.60%	70.89%	68.14%
5	✓	✓	✓	✓	✓	—	61.01%	67.29%	65.61%
6	✓	✓	✓	✓	—	✓	64.90%	70.07%	67.39%
ours	✓	✓	✓	✓	—	—	**70.89%**	**72.11%**	**71.49%**

TABLE IV: Ablation studies of the model.

LayerNorm did not help is that, in changing the activations for each frame separately, the saliency of bright spots (turn signal lights) is diminished, which degrades performance.

V. CONCLUSION

In this paper, we have tackled the important and unexplored problem of turn signal classification. We proposed a method that can be trained end-to-end and is able to handle different viewpoints of the vehicles. The proposed network is designed to reason about both spatial and temporal features through attention, convolutions, and recurrence to classify turn signal states at the frame level for a sequence of observations. We train and evaluate our method using a dataset containing over 1.2 million real-word images. Future works in this problem include the extension to signals from emergency vehicles and the use of more features from classification (such as images from underexposed cameras).

REFERENCES

[1] "Side turn signal lamps for vehicles less than 12 m in length," SAE International, Warrendale, PA, Standard, 2014.

[2] B. Frohlich, M. Enzweiler, and U. Franke, "Will this car change the lane?-turn signal recognition in the frequency domain," in *Intelligent Vehicles Symposium Proceedings*. IEEE, 2014.

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

[4] L. Chen, X. Hu, T. Xu, H. Kuang, and Q. Li, "Turn signal detection during nighttime by cnn detector and perceptual hashing tracking," in *Transactions on Intelligent Transportation Systems*. IEEE, 2017.

[5] R. K. Satzoda and M. M. Trivedi, "Looking at vehicles in the night: Detection and dynamics of rear lights," in *Transactions on Intelligent Transportation Systems*. IEEE, 2016.

[6] Y. Xiang, W. Choi, Y. Lin, and S. Savarese, "Subcategory-aware convolutional neural networks for object proposals and detection," in *Winter Conference on Applications of Computer Vision*. IEEE, 2017.

[7] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2017.

[8] J. QiongYan and Y. LiXu, "Accurate single stage detector using recurrent rolling convolution," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2017.

[9] F. Chabot, M. Chaouch, J. Rabarisoa, C. Teulière, and T. Chateau, "Deep manta: A coarse-to-fine many-task network for joint 2d and 3d vehicle analysis from monocular image," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2017.

[10] D. Frossard and R. Urtasun, "End-to-end learning of multi-sensor 3d tracking by detection," in *International Conference on Robotics and Automation*. IEEE, 2018.

[11] B. Lee, E. Erdenee, S. Jin, M. Y. Nam, Y. G. Jung, and P. K. Rhee, "Multi-class multi-object tracking using changing point detection," in *European Conference on Computer Vision*. Springer, 2016.

[12] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2017.

[13] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *International Conference on Computer Vision*. IEEE, 2017.

[14] S. Liu, J. Jia, S. Fidler, and R. Urtasun, "Sgn: Sequential grouping networks for instance segmentation," in *International Conference on Computer Vision*. IEEE, 2017.

[15] M. Bai and R. Urtasun, "Deep watershed transform for instance segmentation," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2017.

[16] J. Pang, W. Sun, J. Ren, C. Yang, and Q. Yan, "Cascade residual learning: A two-stage convolutional neural network for stereo matching," in *International Conference on Computer Vision Workshops*. IEEE, 2017.

[17] A. Kendall, H. Martirosyan, S. Dasgupta, P. Henry, R. Kennedy, A. Bachrach, and A. Bry, "End-to-end learning of geometry and context for deep stereo regression," in *International Conference on Computer Vision*. IEEE, 2017.

[18] A. Seki and M. Pollefeys, "Sgm-nets: Semi-global matching with neural networks," in *Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, 2017.

[19] A. Geiger, M. Lauer, C. Wojek, C. Stiller, and R. Urtasun, "3d traffic scene understanding from movable platforms," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 5, pp. 1012–1025, 2014.

[20] T. Scharwächter, M. Enzweiler, U. Franke, and S. Roth, "Stixmantics: A medium-level model for real-time semantic scene understanding," in *European Conference on Computer Vision*. Springer, 2014.

[21] C. Chen, A. Seff, A. Kornhauser, and J. Xiao, "Deepdriving: Learning affordance for direct perception in autonomous driving," in *International Conference on Computer Vision*. IEEE, 2015.

[22] H.-K. Hsu, Y.-H. Tsai, X. Mei, K.-H. Lee, N. Nagasaka, D. Prokhorov, and M.-H. Yang, "Learning to tell brake and turn signals in videos using cnn-lstm structure," in *International Conference on Intelligent Transportation Systems*. IEEE, 2017.

[23] S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, 2013.

[24] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Neural Information Processing Systems*, 2014.

[25] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese, "Social lstm: Human trajectory prediction in crowded spaces," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2016.

[26] Z. Deng, A. Vahdat, H. Hu, and G. Mori, "Structure inference machines: Recurrent neural networks for analyzing relations in group activity recognition," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2016.

[27] S. Yeung, O. Russakovsky, G. Mori, and L. Fei-Fei, "End-to-end learning of action detection from frame glimpses in videos," in *International Conference on Robotics and Automation*. IEEE, 2016.

[28] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2015.

[29] 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 *International Conference on Robotics and Automation*. IEEE, 2015.

[30] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt, "Sequential deep learning for human action recognition," in *International Workshop on Human Behavior Understanding*. Springer, 2011.

[31] S. Xingjian, Z. Chen, H. Wang, D.-Y. Yeung, W.-K. Wong, and W.-c. Woo, "Convolutional lstm network: A machine learning approach for precipitation nowcasting," in *Neural Information Processing Systems*, 2015.

[32] N. Homayounfar, W.-C. Ma, S. Kowshika Lakshmikanth, and R. Urtasun, "Hierarchical recurrent attention networks for structured online maps," in *Conference on Computer Vision and Pattern Recognition*. IEEE, 2018.

[33] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[34] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International Conference on Machine Learning*, 2015.

[35] S. Sharma, R. Kiros, and R. Salakhutdinov, "Action recognition using visual attention," *arXiv preprint arXiv:1511.04119*, 2015.

[36] V. Ramanathan, J. Huang, S. Abu-El-Haija, A. Gorban, K. Murphy, and L. Fei-Fei, "Detecting events and key actors in multi-person videos," in *International Conference on Robotics and Automation*. IEEE, 2016.

[37] Z. Li, K. Gavrilyuk, E. Gavves, M. Jain, and C. G. Snoek, "Videolstm convolves, attends and flows for action recognition," *Computer Vision and Image Understanding*, vol. 166, pp. 41–50, 2018.

[38] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv:1409.1556*, 2014.

[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv:1412.6980*, 2014.

[40] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015.

[41] M. Ren, R. Liao, R. Urtasun, F. H. Sinz, and R. S. Zemel, "Normalizing the normalizers: Comparing and extending network normalization schemes," in *arXiv:1611.04520*, 2016.