Interaction Graphs for Object Importance Estimation in On-road Driving Videos

Zehua Zhang[1]*, Ashish Tawari[2], Sujitha Martin[2], and David Crandall[1]

Abstract— A vehicle driving along the road is surrounded by many objects, but only a small subset of them influence the driver's decisions and actions. Learning to estimate the importance of each object on the driver's real-time decision-making may help better understand human driving behavior and lead to more reliable autonomous driving systems. Solving this problem requires models that understand the interactions between the ego-vehicle and the surrounding objects. However, interactions among other objects in the scene can potentially also be very helpful, *e.g.*, a pedestrian beginning to cross the road between the ego-vehicle and the car in front will make the car in front less important. We propose a novel framework for object importance estimation using an interaction graph, in which the features of each object node are updated by interacting with others through graph convolution. Experiments show that our model outperforms state-of-the-art baselines with much less input and pre-processing.

I. INTRODUCTION

Driving is a complex task because it involves highly dynamic, complex environments in which many different autonomous agents (other drivers, pedestrians, etc.) are acting at the same time. Human drivers make real-time decisions by combining information from multiple sources, among which visual information often plays the most important part. Since humans have foveated vision systems that require controlling both head pose and eye gaze [1]–[6], people must identify and attend to the most task-relevant objects in their visual field at any given time.

Learning to predict drivers' attention has become a popular topic in recent years [7]–[11] due to potential application in advanced driver assistance systems (ADAS) and autonomous driving. However, much of this work [7]–[9], [11] focuses on predicting pixel-level human eye gaze, which has two main drawbacks. First, drivers will often look at objects irrelevant to the driving task — *e.g.*, beautiful scenery. Second, gaze is limited to a single, small region at any moment in time, whereas the human may be actively attending to multiple objects (using short-term memory, peripheral vision, or multiple saccades) — for example, if a group of people is crossing the road, a good driver (and hence an autonomous system) should pay attention to all of them instead of just a single person.

To overcome the above problems, in this paper we investigate how to directly estimate each object's importance

Fig. 1: *Given a video clip, our goal is to estimate, in an online fashion, which objects in the last frame are important for the driver to make real-time control decisions.* We also visualize different candidate object boxes, while the red box is the ground truth. Without considering the interaction between pedestrians and the front car, traditional methods make mistakes by predicting both pedestrians and the front car as important. Our proposed method can effectively model the interactions among other objects (visualized as dashed arrows) with a novel interaction graph and thus make the correct prediction. In this example, the front car prevents the ego vehicle from hitting the pedestrians and thus reduces the importance of them.

to the ego-vehicle for making decisions in on-road driving videos without using eye gaze as an intermediate step, as is shown in Fig. 1. We use the dataset collected by Gao *et al.* [10], in which each sample clip was viewed by experienced drivers and each object in the last frame of the clip was labeled as either important or not. These on-road videos were recorded by forward-facing cameras mounted on cars. This perspective is somewhat different from the drivers' actual field of view since the dashboard camera is fixed; it makes the video more stable than if it were from a head-mounted camera, but also makes the problem more challenging since we cannot use cues about where the driver was looking (e.g., human drivers tend to adjust their head pose to center an attended region within their visual field [12]).

We propose to leverage frequent interactions among objects in the scene other than the ego-vehicle. Such interactions are often overlooked by other methods but extremely helpful. For example, in Fig. 1, the front car will prevent the ego-vehicle from hitting the pedestrians and thus greatly reduces their importance: the ego-vehicle's driver only needs to avoid hitting the car in front at the moment. We model these interactions using a novel interaction graph, with features pooled from an I3D-based [13] feature extractor as the

nodes, and interaction scores learned by the network itself as the edges. Through stacked graph convolutional layers, object nodes interact with each other and their features are updated from those of the nodes they closely interact with. Our experiments show that the interaction graph greatly improves performance and our model outperforms state-of-the-art methods with less input information (RGB clips only) and pre-processing (object detection on the target frame only).

II. RELATED WORK

A. Driver Attention Prediction

As interest in (semi-)autonomous driving is growing, researchers have paid more attention to the problem of predicting driver attention. This is typically posed as predicting pixel-level eye gaze in terms of likelihood maps [7]–[9], [11]. Fully convolutional networks [14], which were original proposed to solve image segmentation, have been applied by Tawari *et al.* [8], [9] for similar dense spatial probability prediction tasks. Palazzi *et al.* [7] combine features from multiple branches with RGB frames, optical flow, and semantic segmentation to create the final prediction. Xia *et al.* propose to handle critical situations by paying more attention to the frames that are identified as crucial driving moments based on human driver eye gaze movements [11].

However, using eye gaze prediction to estimate driver attention has limitations, such as that drivers can be attending to multiple objects at once (through short-term memory, frequent saccades or peripheral vision). To overcome these drawbacks, Gao *et al.* [10] collected an on-road driving video dataset with objects labeled as important or not by experienced drivers. They use an LSTM [15] leveraging goal information for object importance estimation. To achieve state-of-the-art performance, however, their technique requires multiple sources of input including maneuver information of the planned path, RGB clips, optical flow, and location information, as well as complex pre-processing of per-frame object detection and tracking.

B. Object-level Attention

While many papers study saliency [16]–[19] and eye gaze [12], [20]–[26] prediction, only a few focus on object-level attention [27]–[32]. Hand-designed features are applied for important object and people detection in Lee *et al.* [27]. Pirsiavash and Ramanan [28] and Ma *et al.* [29] detect objects in hands as a proxy for attended objects. Bertasius *et al.* [31] propose an unsupervised method for attended object prediction. The recent work of Zhang et al. [32] tries to jointly identify and locate the attended object. Inspired by the foveated nature of human vision system, the class and the location information are integrated through a self-validation module to further refine the prediction. But these techniques are for egocentric videos in which both head movement and ego hands are available for making inferences, while in our settings the camera is fixed and hand information is not applicable.

C. Graph Convolutional Networks

Graph convolutional networks (GCNs) [33], [34] have become popular in computer vision problems [35]–[38] for their ability to capture long-range relationships in non-Euclidean space. Compared with traditional convolutional networks which need to stack many layers for a large receptive field, GCNs can efficiently model long-range relations with an adjacency matrix. Inspired by these papers, we apply interaction graphs to solve the problem of object importance estimation in on-road driving videos. To the best of our knowledge, ours is the first work to perform per-node predictions in computer vision problems with GCNs. Also, our model learns to predict the graph edges themselves based on nodes' interactions, while existing work typically formulates the edges with hand-designed metrics such as spatial-temporal overlap [35].

III. OUR MODEL

Given videos taken by forward-facing (egocentric) vehicle cameras, our aim is to estimate the importance of each object in the scene to the driver. Following the same problem setting as [10], we do online estimation, predicting object importance scores in the last frame of each video clip. We propose a novel framework that leverages the interactions among objects using graph convolution. An overview of our model is shown in Fig. 2.

Object proposals. Since it is time-sensitive, we would like online prediction to require as little pre-processing as possible. In contrast to the state-of-the-art method [10] which requires object detection on each frame of the input video as well as object tracking, our model only needs to run object detection on the target frame (the last frame in the online detection setting). We apply Mask RCNN [39], using an off-the-shelf ResXNet-101 [40] 32x8d Mask RCNN model [41] trained on MSCOCO [42] without any further fine-tuning. Proposals of object classes other than person, bicycle, car, motorcycle, bus, train, truck, traffic light, and stop sign are removed since they are irrelevant to our task. For simplicity, a number of dummy boxes with coordinates of $(x1, y1, x2, y2) = (0.07, 0.91, 0.97, 1.0)$ (all box coordinates are rescaled to $[0, 1]$ with respect to the height and width of the input frame) are padded until each target frame contains the same number of proposals. We empirically set the number of object proposals per target frame after padding to be $N = 40$. The dummy box is the hood of the ego car, for which the image content almost remains unchanged for all of the samples. Also, object proposals were pre-generated to save time in our experiments, though during model design we tried our best to reduce pre-processing steps to make it meet the time requirements of online prediction.

Visual feature extractor. Since only RGB clips are used as input, making correct object importance predictions requires as strong a feature extractor as possible. We use Inception-V1 [43] I3D [13] because of its capacity for capturing both rich spatial and temporal features. Temporal motion information is important for reasoning about both the ego-vehicle's and other objects' intention and future movement,
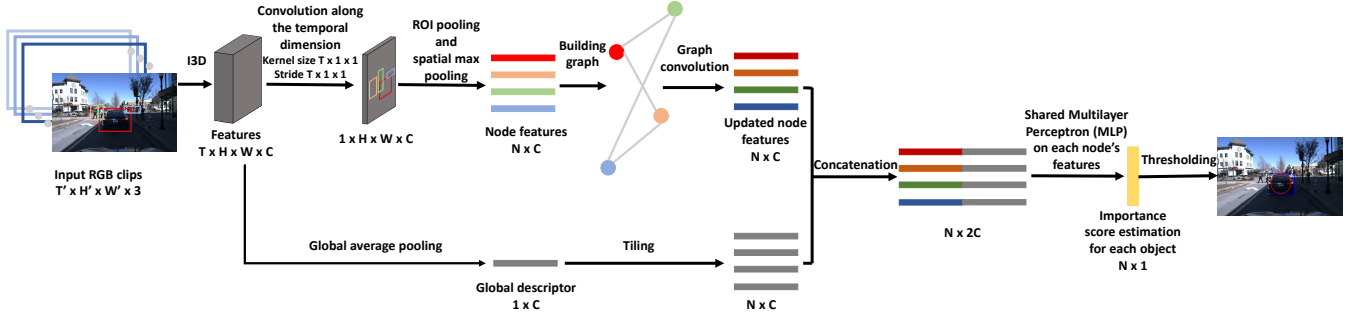
Fig. 2: *The architecture of our proposed model.* Features from the I3D-based feature extractor are first aggregated temporally by convolution along only the temporal dimension. Then ROI pooling and spatial max pooling are applied to obtain each object node's feature vector. These features are updated by interacting with other nodes through graph convolution based on the learned interaction graph edges. Finally, the updated features are concatenated with the global descriptor and fed into a shared Multilayer Perceptron (MLP) for object importance estimation. The step of object proposal generation is omitted for clarity.

and spatial appearance information helps determine the inherent characteristics of each object. Given T' contiguous RGB frames $B_{t,t+T'-1} \in R^{T' \times H' \times W' \times 3}$, we feed them through I3D and extract features $F \in R^{T \times H \times W \times C}$ from the last mixing layer of rgb_Mixed_5c. We have $T = \frac{T'}{8}$, $H = \frac{H'}{32}$, $W = \frac{W'}{32}$, and $C = 1024$ based on the architecture setting of Inception-V1 I3D.

Feature vectors of graph nodes. Extracted features F from above are further aggregated temporally through one-layer convolution along only the temporal dimension, with kernel size and stride set to $T \times 1 \times 1$. From the obtained feature maps $F' \in R^{1 \times H \times W \times C}$, the features $\{f_i | i = 1, 2, ..., N\}$ for each object are pooled using ROI Pooling [39], [44]. Following [39], we have $f_i \in R^{7 \times 7 \times C}$ (the temporal dimension is removed). These feature maps then go through a spatial max pooling layer, resulting in feature vectors $\{v_i \in R^C | i = 1, 2, ..., N\}$ for each object node.

Graph edge formulation. The strength of an edge E_{ij} in our interaction graph should reflect how closely two connected objects i and j interact with each other. We propose that the network itself learns to model the edge strength E_{ij} by estimating how closely two nodes i and j interact with each other. Given node features v_i and v_j, an interaction score IS_{ij} is first computed,

$$IS_{ij} = \Phi(\Gamma(v_i)||\Gamma'(v_j)), \qquad i,j \in \{1,2,...,N\}, \quad (1)$$

where $\Gamma(\cdot)$ and $\Gamma'(\cdot)$ are linear transformations with different learnable parameters w and w'. $\Phi(x) = \phi x$ is a linear transformation with ϕ as learnable parameters and $||$ denotes concatenation. With an interaction matrix IS obtained by computing interaction scores for each pair of nodes, we calculate E_{ij} by applying softmax on IS as well as adding an identity matrix $I_N \in R^{N \times N}$ to force self attention,

$$E_{ij} = \frac{e^{IS_{ij}}}{\sum_{k=1}^{N} e^{IS_{ik}}} + I_N, \qquad i,j \in \{1,2,...,N\}. \quad (2)$$

In this way, the model learns an interaction graph itself based on each node's features. The learned edge strength E_{ij} indicates how much node j will affect updating node

i's features through graph convolution and thus reflects how closely they are interacting.

We note that the interaction graph learned here is a directional graph, as $\Gamma(\cdot)$ and $\Gamma'(\cdot)$ are different transformations. This is reasonable since how much node i affect node j is not necessarily the same as how much j affects i. For example, in Fig. 1, while the front car will greatly reduce the importance of the pedestrians, the pedestrians almost have no influence on how important the front car is to the ego vehicle.

An alternative way of forming the graph is based on feature similarity between pairs of nodes following [35]. We ran experiments to compare our interaction graph with that similarity graph, and found that the model is not able to learn well with similarity graphs. Our hypothesis is that similarity graphs are not suitable for our problem, as objects sharing similar appearance or motion do not necessarily closely interact, and vice versa. For example, in Fig.1, the front car and the pedestrians are very different in terms of both appearance and motion, despite the close interaction between them. Since our proposed interaction graph yields better performance, we use it in our experiments here.

Graph convolution. With the graph E formulated, we perform graph convolution to allow nodes to interact and update each other. One layer of graph convolution can be represented as:

$$V' = \sigma(EVW) \quad (3)$$

where $E \in R^{N \times N}$ is the edge matrix, $V = [v_1, v_2, ..., v_N] \in R^{N \times C}$ is the input node feature matrix, and $W \in R^{C \in C'}$ is the weight matrix of the layer. $\sigma(\cdot)$ is a non-linear function; we use ReLU in our experiments. We stack 3 graph convolutional layers and simply set $C' = C$ for all of them. After the graph convolution, we obtain an updated feature matrix $U \in R^{N \times C}$.

Per-node object importance estimation. We now perform per-node object importance estimation. Although each node's features are updated through GCN to capture long-range relationships with other nodes, some global context may still be missing because object proposals cannot cover the whole image. Also, the object detector is not perfect and useful

objects may be missed (*e.g.,* small objects such as traffic lights). To circumvent this problem, we first apply global average pooling on the extracted features F from I3D to obtain a global descriptor $D \in R^{1 \times C}$, which are then tiled N times and concatenated with the updated node features U. Each row of the resulting features $Y \in R^{N \times 2C}$ is fed into a shared Multilayer Perceptron (MLP) for the final importance score estimation,

$$\widehat{score}_i = sigmoid(MLP(Y_i)), \qquad i \in 1, 2, ..., N. \quad (4)$$

Other implementation and training details

- **Missing detections.** Since we apply multi-fold cross validation (as in [10]) in our experiments, each sample will either serve as a training sample or a testing sample. For those samples with ground truth objects not detected by our object detector, we prepare two sets of proposals for training and testing respectively. The set for testing is the same as the 40 proposals obtained by padding dummy boxes to the results of Mask RCNN, while the set for training is slightly different as we replace dummy boxes with the ground truth object boxes which were not detected to avoid misleading the network. Note that missing detections of ground truth objects still happen when these samples serve as testing data, and thus our model can never reach $AP = 100\%$.

- **Hard negative mining and loss functions.** The dataset collected by Gao *et al.* [10] suffers from significant imbalance: of 8,611 samples (short video clips), only 4,268 objects are labeled as important. Considering our setting of 40 box nodes per sample, the ratio of the total number of positive boxes over negative boxes is almost 1:80. In contrast to [10] which applies weighted-cross-entropy based on the number of positive and negatives boxes, we do hard negative mining to address the problem. During each batch, we first compute the losses for each node importance estimation $\widehat{score}_i$ with binary cross entropy loss,

$$L_{node_i} = L_{be}(score_i, \widehat{score}_i), \qquad i \in 1, 2, ..., N, \quad (5)$$

where $score_i$ is the corresponding ground truth and,

$$L_{be}(x, \hat{x}) = -xlog(\hat{x}) - (1 - x)log(1 - \hat{x}). \quad (6)$$

Then the losses for negative nodes are sorted and we take only the N_{neg} greatest from them, along with the losses for all the positive nodes, to compute the total loss. Letting N_{pos} denote the total number of positive nodes, we empirically found that $N_{neg} = max(5 \cdot N_{pos}, 10)$ works well (after multiple experiments on different ratios). Supposing Ψ_{pos} and Ψ_{neg} denote the sets of indices for all the positive nodes and the selected negative nodes whose node losses are among the top N_{neg}, the total loss is then,

$$L_{total} = \frac{1}{N_{pos}} \left(\sum_{j \in \Psi_{pos}} L_{node_j} + \sum_{k \in \Psi_{neg}} L_{node_k} \right). \quad (7)$$

- **Other details.** We implemented the model with Keras [45] and Tensorflow [46]. A batch normalization

layer [47] was inserted after each layer of the feature extractor, with momentum 0.8. We used RGB clips with dimension $16 \times 200 \times 320 \times 3$ as the only input. During training, the I3D feature extractor was initialized with weights pretrained on Kinetics [48] and ImageNet [49], while other parts were randomly initialized. We trained the model with stochastic gradient descent with initial learning rate 0.0003, momentum 0.9, decay 0.0001, and L2 regularizer 0.0005. The loss function to be optimized was as Eq. 7. When making inference, the model predicts an importance score in the range of $[0, 1]$ for each of the object proposals, resulting in a 40×1 prediction per sample.

IV. EXPERIMENTS

A. Experiment Settings

Dataset. We evaluate our model on the the on-road driving video dataset used by Gao *et al.* [10]. This dataset consists of 8,611 annotated samples, each of which is a 30-frame RGB clip (during our experiments we only use the last 16 of them) recording real-world driving in highly-dynamic environments. The dataset focuses on real-time driving around road intersections, and were annotated by experienced drivers with object importance estimates. Please refer to [10] for detailed statistics about the dataset. In our experiments, we follow the same data split as in [10] and also perform 3-fold cross validation.

Metrics. We compute 11-point average precision (AP) [50] for each data split and then take the average across the 3 splits. A predicted box is considered as correct if its IOU with one of the unmatched ground truth boxes is over 0.5. We do not allow duplicated matching to the same ground truth box. Also, note that due to false negatives of the object detector, the upper bound performance of our model can never achieve 100%, as in [10].

Baselines. We compare our model with the state-of-the-art goal-oriented model as well as other competitive baselines in [10]. These baselines have similar network structures based on LSTMs [15], while the main difference between them is the input features. The strongest one, Goal-Visual Model, takes as input an RGB clip of 30 frames along with optical flow, goal information, and location information. Tab. I compares our model with these baselines from the perspective of the inputs and pre-processing required.

B. Experiment Results

Qualitative results. We qualitatively compare our model with the Goal-Visual model in Fig. 3. While the Goal-Visual model usually fails in scenarios with multiple potentially-important objects (Fig. 3a, b, c, d, g and h), our model suppresses false positive predictions significantly by leveraging object interactions with our proposed interaction graph. Interestingly, when generating the visualization, our model uses a threshold of 0.3 and the Goal-Visual model uses 0.5, yet our model still makes fewer false positive predictions. Our hypothesis is that when multiple objects can possibly be important, a suppression procedure is performed among

TABLE I: Comparison of our model with other baselines in terms of required input, required pre-processing, and average precision on different splits. Our model outperforms the state-of-the-art with the least input and the easiest pre-processing.

Models	RGB clips	Input length	Optical flows	Goal	Location	Object detection	Tracking	$AP_1 \uparrow$	$AP_2 \uparrow$	$AP_3 \uparrow$	$avgAP \uparrow$
Our model	✓	16	✗	✗	✗	On the target frame	✗	68.5	**73.8**	**71.9**	**71.4**
Goal-Visual Model [10]	✓	30	✓	✓	✓	On each frame	✓	**70.2**	70.3	**72.0**	70.8
Visual Model [10]	✓	30	✓	✗	✓	On each frame	✓	68.1	68.1	70.9	69.0
Goal-Geometry Model [10]	✗	0	✓	✓	✓	N/A	✓	32.1	40.6	41.8	38.2
Visual Model-Image [10]	✓	1	✗	✓	✓	On the target frame	✓	35.5	42.1	32.6	36.7

(a)   **(b)**   **(c)**   **(d)**
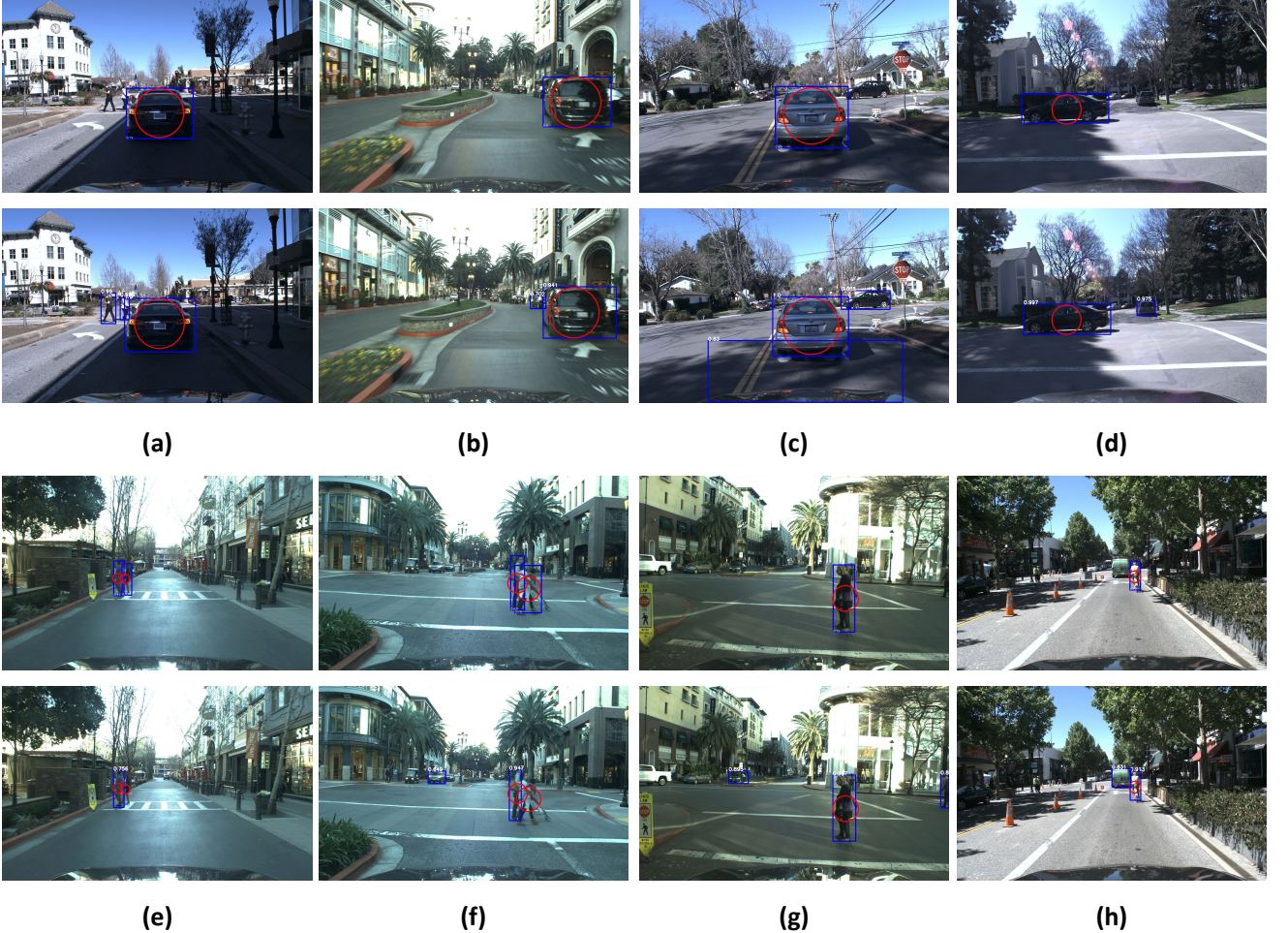
(e)   **(f)**   **(g)**   **(h)**

Fig. 3: *Qualitative comparison of our model (the 1st and the 3rd rows) with the state-of-the-art method of Goal-Visual Model [10] (the 2nd and the 4th rows).* Blue rectangles are the predicted important object boxes and red circles represent the ground truth. In the upper half (a, b, c and d) we visualize samples with vehicles as the important object, while samples with important pedestrians are visualized in the bottom half. In a,b,c,d,g and h, our model outperforms [10] by making fewer false positive predictions, while in e and f it yields more true positive predictions.

nodes of these objects inside the interaction graph. These nodes interact with each other and make inferences based on node features to suppress the false positives. With the interaction graph, our model becomes more cautious about predicting multiple objects as important. However, it does not lose the ability to predict multiple true positives. As shown in Fig. 3 (e) and (f) where the pedestrians are crossing the road together with each other, our model with an interaction graph can effectively capture the relation and assign similar importance scores to them.

Quantitative results. Our model and other baselines are

quantitatively compared in Tab. I. Despite requiring the least input and the easiest pre-processing, our model still outperforms the state-of-the-art model in terms of average AP across the 3 splits. We also observed that our model significantly outperforms the Goal-Visual model on split 2, achieves comparable AP on split 3, but performs worse on split 1. The reason may be that the number of samples on which goal information can significantly help the final estimation varies across the 3 splits. As we will show in the next section, in some cases it is almost impossible for the network to make correct estimates without knowing the goal

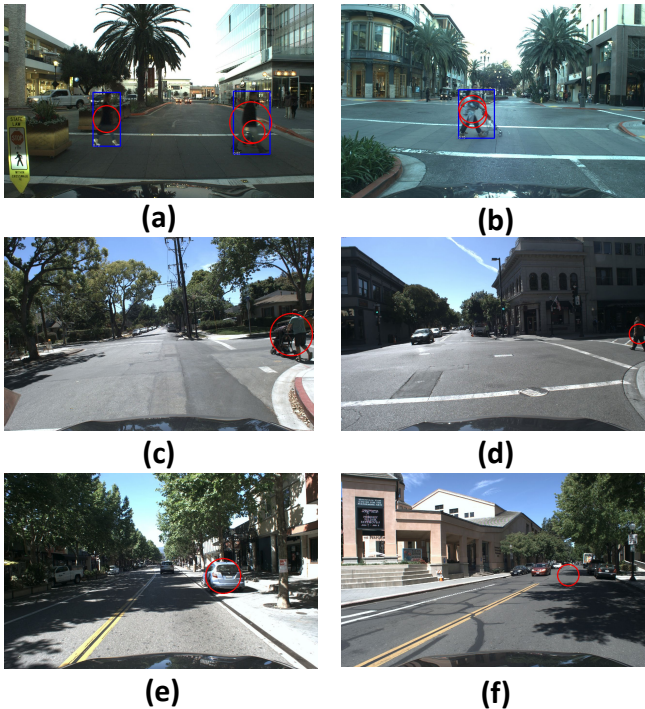**(a)**   **(b)**

(c)   **(d)**

(e)   **(f)**

Fig. 4: *Sample failure cases of our model.* Blue rectangles are the predicted important object boxes and red circles represent the ground truth. (a) and (b) are caused by missing detections. (c) and (d) are due to lack of goal information. (e) and (f) are samples with confusing ground truth.

of the ego-vehicle. In the future, we plan to have human annotators further analyze the 3 splits to investigate this.

Failure cases. Sample failure cases are visualized in Fig. 4. The first rows show failures caused by missing detections, which are not the fault of our model as the proposals are generated by off-the-shelf third-party object detectors. Hopefully in the future, this kind of failure can be solved with better object detection models. The failures in the second row reflect the difficulty of online prediction as the future is unknown. In both of Fig. 4 (c) and (d), the ego-vehicles have been going straight and stop right at the intersection in the frames. Without goal information, the model can never know that the driver plans to turn right, and thus fails to predict the pedestrians crossing the road at the right as important. To solve this problem requires further incorporating goal information to our model. The last row shows samples with confusing ground truth. Fig. 4 (e) shows a common case in which the annotator labeled parked cars along the road as important even though no one is starting or driving them, while Fig. 4 (f) contains incorrect ground truth of part of the road region as an important object.

C. Ablation Studies

We performed ablation studies to evaluate the effectiveness of the proposed interaction graph, as well as some of our model design choices.

Interaction graph. We remove the interaction graph along with the graph convolution layers and directly concatenate

TABLE II: *Results of ablation studies.*

Model	$AP_1 \uparrow$	$AP_2 \uparrow$	$AP_3 \uparrow$	$avgAP \uparrow$
Our full model	**68.5**	**73.8**	**71.9**	**71.4**
Remove interaction graph	65.0	70.6	69.3	68.3
Remove global descriptor	66.0	71.7	70.3	69.3
Remove self attention	4.2	5.3	5.8	5.1

the pooled features of each object proposals with the global descriptor. The concatenated features are fed into the shared MLP for final estimation. The results are in Tab. II. The AP on each split drops by 3.5%, 3.2%, and 2.6%, respectively, and the average AP falls below the score of the Visual model in [10], indicating that the interaction graph is important for performance improvement.

Global descriptor. We remove the global descriptor and let the model make importance estimation based only on the updated node features through GCN. Performance drops on all three of the splits, as shown in Tab. II. This implies that the global descriptor is helpful as it provides useful global context. Also, we observed that AP drops less by removing the global descriptor than removing the interaction graph, suggesting that the interaction graph is more important in improving the performance.

Self attention. The identity matrix I_N in Eq. 2 is removed and we trained the model with the new graph. We found that the model is not able to learn without the forced self attention, as is shown in Tab. II. Self attention is crucial as it ensures that each node can retain its own characteristics while interacting with others in graph convolution.

V. CONCLUSION

We propose a novel framework for online object importance estimation in on-road driving videos with interaction graphs. The graph edges are learned by the network itself based on the nodes' features and reflect how closely the connected nodes interact with each other. Through graph convolutional layers, object nodes are able to interact with each other in the graph and update each other's node features. Experiments show that our model outperforms the state-of-the-art with much less input and much easier pre-processing, and ablation studies demonstrate the effectiveness of the interaction graph as well as our other model design choices.

REFERENCES

[1] M. C. Mozer and M. Sitton, "Computational modeling of spatial attention," *Attention*, vol. 9, pp. 341–393, 1998.

[2] S. Lazzari, D. Mottet, and J.-L. Vercher, "Eye-hand coordination in rhythmical pointing," *Journal of motor behavior*, vol. 41, no. 4, pp. 294–304, 2009.

[3] M. C. Bowman, R. S. Johannson, and J. R. Flanagan, "Eye–hand coordination in a sequential target contact task," *Experimental brain research*, vol. 195, no. 2, pp. 273–283, 2009.

[4] M. Hayhoe and D. Ballard, "Eye movements in natural behavior," *Trends in cognitive sciences*, vol. 9, no. 4, pp. 188–194, 2005.

[5] E. D. Vidoni, J. S. McCarley, J. D. Edwards, and L. A. Boyd, "Manual and oculomotor performance develop contemporaneously but independently during continuous tracking," *Experimental brain research*, vol. 195, no. 4, pp. 611–620, 2009.

[6] S. Perone, K. L. Madole, S. Ross-Sheehy, M. Carey, and L. M. Oakes, "The relation between infants' activity with objects and attention to object appearance." *Developmental psychology*, vol. 44, no. 5, p. 1242, 2008.

[7] A. Palazzi, D. Abati, F. Solera, R. Cucchiara, *et al.*, "Predicting the driver's focus of attention: the dr (eye) ve project," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 41, no. 7, pp. 1720–1733, 2018.

[8] A. Tawari and B. Kang, "A computational framework for driver's visual attention using a fully convolutional architecture," in *IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2017, pp. 887–894.

[9] A. Tawari, P. Mallela, and S. Martin, "Learning to attend to salient targets in driving videos using fully convolutional rnn," in *International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 3225–3232.

[10] M. Gao, A. Tawari, and S. Martin, "Goal-oriented object importance estimation in on-road driving videos," *arXiv preprint arXiv:1905.02848*, 2019.

[11] Y. Xia, D. Zhang, J. Kim, K. Nakayama, K. Zipser, and D. Whitney, "Predicting driver attention in critical situations," in *Asian Conference on Computer Vision*. Springer, 2018, pp. 658–674.

[12] Y. Li, A. Fathi, and J. M. Rehg, "Learning to predict gaze in egocentric video," in *IEEE International Conference on Computer Vision (ICCV)*, 2013.

[13] J. Carreira and A. Zisserman, "Quo vadis, action recognition? a new model and the kinetics dataset," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6299–6308.

[14] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 3431–3440.

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

[16] Q. Hou, M.-M. Cheng, X. Hu, A. Borji, Z. Tu, and P. H. Torr, "Deeply supervised salient object detection with short connections," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 3203–3212.

[17] N. Liu and J. Han, "Dhsnet: Deep hierarchical saliency network for salient object detection," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 678–686.

[18] P. Zhang, D. Wang, H. Lu, H. Wang, and X. Ruan, "Amulet: Aggregating multi-level convolutional features for salient object detection," in *IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 202–211.

[19] T. Judd, K. Ehinger, F. Durand, and A. Torralba, "Learning to predict where humans look," in *IEEE International Conference on Computer Vision (ICCV)*. IEEE, 2009, pp. 2106–2113.

[20] X. Huang, C. Shen, X. Boix, and Q. Zhao, "Salicon: Reducing the semantic gap in saliency prediction by adapting deep neural networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 262–270.

[21] Z. Zhang, S. Bambach, C. Yu, and D. J. Crandall, "From coarse attention to fine-grained gaze: A two-stage 3d fully convolutional network for predicting eye gaze in first person video," in *British Machine Vision Conference (BMVC)*, 2018.

[22] M. Zhang, K. T. Ma, J. H. Lim, Q. Zhao, and J. Feng, "Deep future gaze: Gaze anticipation on egocentric videos using adversarial networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[23] Y. Huang, M. Cai, Z. Li, and Y. Sato, "Predicting gaze in egocentric video by learning task-dependent attention transition," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 754–769.

[24] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 20, no. 11, pp. 1254–1259, 1998.

[25] A. Borji, D. N. Sihite, and L. Itti, "Probabilistic learning of task-specific visual attention," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[26] K. Yamada, Y. Sugano, T. Okabe, Y. Sato, A. Sugimoto, and K. Hiraki, "Attention prediction in egocentric video using motion and visual saliency," in *Advances in Image and Video Technology*, Y.-S. Ho, Ed., 2012, pp. 277–288.

[27] Y. J. Lee, J. Ghosh, and K. Grauman, "Discovering important people and objects for egocentric video summarization," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2012, pp. 1346–1353.

[28] H. Pirsiavash and D. Ramanan, "Detecting activities of daily living in first-person camera views." in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012, pp. 2847–2854.

[29] M. Ma, H. Fan, and K. M. Kitani, "Going deeper into first-person activity recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1894–1903.

[30] G. Bertasius, H. S. Park, S. X. Yu, and J. Shi, "First person action-object detection with egonet," *arXiv preprint arXiv:1603.04908*, 2016.

[31] G. Bertasius, H. Soo Park, S. X. Yu, and J. Shi, "Unsupervised learning of important objects from first-person videos," in *IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 1956–1964.

[32] Z. Zhang, C. Yu, and D. Crandall, "A self validation network for object-level human attention estimation," in *Advances in Neural Information Processing Systems*, 2019, pp. 14 702–14 713.

[33] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[34] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2016, pp. 3844–3852.

[35] X. Wang and A. Gupta, "Videos as space-time region graphs," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 399–417.

[36] Y. Chen, M. Rohrbach, Z. Yan, Y. Shuicheng, J. Feng, and Y. Kalantidis, "Graph-based global reasoning networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 433–442.

[37] Y. Li and A. Gupta, "Beyond grids: Learning graph representations for visual recognition," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018, pp. 9225–9235.

[38] J. Yang, J. Lu, S. Lee, D. Batra, and D. Parikh, "Graph r-cnn for scene graph generation," in *European Conference on Computer Vision (ECCV)*, 2018, pp. 670–685.

[39] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-cnn," in *IEEE International Conference on Computer Vision (ICCV)*, 2017, pp. 2961–2969.

[40] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[41] R. Girshick, I. Radosavovic, G. Gkioxari, P. Dollár, and K. He, "Detectron," https://github.com/facebookresearch/detectron, 2018.

[42] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *European Conference on Computer Vision (ECCV)*. Springer, 2014, pp. 740–755.

[43] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 1–9.

[44] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2015, pp. 91–99.

[45] F. Chollet, J. Allaire, *et al.*, "R interface to keras," https://github.com/rstudio/keras, 2017.

[46] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "Tensorflow: A system for large-scale machine learning," in *USENIX Conference on Operating Systems Design and Implementation*, 2016, pp. 265–283.

[47] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning (ICML)*, 2015.

[48] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijaya-narasimhan, F. Viola, T. Green, T. Back, P. Natsev, *et al.*, "The kinetics human action video dataset," *arXiv preprint arXiv:1705.06950*, 2017.

[49] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Ieee, 2009, pp. 248–255.

[50] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results," http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.