Journal of Ambient Intelligence and Humanized Computing
https://doi.org/10.1007/s12652-023-04550-8

ORIGINAL RESEARCH

Explaining autonomous driving with visual attention and end-to-end trainable region proposals

Luca Cultrera[1] · Federico Becattini[1] · Lorenzo Seidenari[1] · Pietro Pala[1] · Alberto Del Bimbo[1]

Received: 5 May 2022 / Accepted: 26 January 2023

Abstract
Autonomous driving is advancing at a fast pace, with driving algorithms becoming more and more accurate and reliable. Despite this, it is of utter importance to develop models that can offer a certain degree of explainability in order to be trusted, understood and accepted by researchers and, especially, society. In this work we present a conditional imitation learning agent based on a visual attention mechanism in order to provide visually explainable decisions by design. We propose different variations of the method, relying on end-to-end trainable regions proposal functions, generating regions of interest to be weighed by an attention module. We show that visual attention can improve driving capabilities and provide at the same time explainable decisions.

Keywords Autonomous driving · Explainability

1 Introduction

Although autonomous driving vehicles are starting to become a reality, their diffusion worldwide is still slowed down by how such advancements are perceived by society. To ensure the pervasivity of automotive in everyday life, it is fundamental that algorithms and learning models guiding the decisions of autonomous vehicles are trustworthy, transparent and fully understandable. In other words, it is of paramount importance that the technologies that the end user will rely on must be explainable. Explainability in autonomous driving has been largely studied in recent years, especially regarding machine learning and computer vision algorithms that make autonomous navigation possible (Omeiza et al. 2021; Zablocki et al. 2021; Cultrera et al. 2020). Explanations can be provided in different forms and styles, e.g. presenting factual, contrastive or counterfactual evidence to support cause effect relationships (Lim and Dey 2009) or showing the sensitivity of the decision with reference to parts of the input (Omeiza et al. 2021).

A simple yet effective way to interpret decisions, especially for computer vision based applications, is to provide a visual explanation of what the model is focusing on. Ex-post methods, such as Grad-cam (Selvaraju et al. 2017), attempt to demystify "black-box" models locating the most relevant pixels in the input image that lead to a decision. Such works, however, despite being largely used for explaining pre-trained classifiers, have been shown to be hard to adapt for regression tasks (Letzgus et al. 2021). A better alternative is to exploit a model specifically designed to be explainable. Visual attention has been largely used for this purpose, integrating in a model a mechanism to weight regions or parts of the input to establish their importance explicitly (Anderson et al. 2018).

In this work, we present a study on how different types of visual attention can be exploited to explain the decisions of a driving agent. We propose a conditional imitation learning approach capable of learning driving policies from RGB frames, trained with an attention block that weighs image

Special issue on ambient understanding for mobile autonomous robots.

✉ Federico Becattini
federico.becattini@unifi.it

Luca Cultrera
luca.cultrera@unifi.it

Lorenzo Seidenari
lorenzo.seidenari@unifi.it

Pietro Pala
pietro.pala@unifi.it

Alberto Del Bimbo
alberto.delbimbo@unifi.it

[1] University of Florence, Firenze, Italy

Published online: 03 February 2023

regions based on their importance for the task. We design different region proposals, trained end-to-end along with the driving agent. A preliminary version of this work was described in Cultrera et al. (2020), introducing the first visual attention based driving agent in the literature that learned to assign attention weights to a static grid of regions of interest in the input image. This work differs substantially from Cultrera et al. (2020) in several aspects: (i) we overcome the limitation of having static proposals by developing different dynamic region proposal functions based on either Region Proposal Networks (Girshick 2015) or Spatial Transformer Networks (Jaderberg et al. 2015); ii) we provide a comparison with ex-post explainability methods, showing the importance of explicitly modeling visual attention to obtain meaningful interpretations; (iii) we show that the learned attention maps can be used to retrieve hard examples framing the problem as an anomaly detection task.

2 Related works

Imitation learning is an approach for learning a policy that reflects a behaviour by analyzing demonstrations performed by an expert. Prior work has often exploited this paradigm for automotive, where a driving policy can be learned by attempting to replicate steering commands for urban navigation (Bojarski et al. 2016) or following high level commands such as *turn* or *go straight* (Codevilla et al. 2018). This type of task has also been addressed by Liang et al. (2018) with reinforcement learning. Such approaches learn a mapping between what is perceived by the ego-vehicle and the output controls. However, to foster generalization an intermediate representation can be used such as low-dimensional affordance as in Sauer et al. (2018) or perception indicators related to the surrounding environment as proposed by Chen et al. (2015).

Different types of sensor data are often exploited and additional synthetic data can be gathered from simulators to train driving models (Codevilla et al. 2018; Berlincioni et al. 2019; Lee et al. 2018; Berlincioni et al. 2021). Several approaches exploit additional data rather than RGB frames alone, e.g, considering depth (Xiao et al. 2019), semantic segmentation (Li et al. 2018) or LiDAR data (Haris and Glowacz 2022) as inputs or to perform multi-task learning (Xiao et al. 2019; Yang et al. 2018; Codevilla et al. 2019; Ishihara et al. 2021; Greco et al. 2022). The importance of model architecture and image features has also been stressed in the literature, benchmarking different convolutional networks (Orden and Visser 2021) or learning to generate features capable of generalizing across different environmental conditions (Guo et al. 2021). Temporal modeling is also used by Eraqi et al. (2017), George et al. (2018) and Xu et al. (2017). Zhang and Cho (2016), instead, developed an agent that can gracefully fallback to a safe policy when dangerous scenarios emerge.

One of the biggest problems of training imitation learning methods end-to-end is the inability to explain a model's behaviour. In fact, being a safety critical domain, explainability is becoming of prominent interest in automotive (Kim and Canny 2017; Xu et al. 2020; Marchetti et al. 2022; Bojarski et al. 2017). Xu et al. (2020) predict vehicle actions such as slowing down and provide a textual explanation. Marchetti et al. (2022) exploit memory augmented neural networks to forecast agent positions and reason about cause-effect relationships in motion patterns. In Kim et al. (2020), a model is proposed to generate advice (e.g., "wet road") that are then converted into actions (e.g. "slow down").

Qualitative explanations are also a common way of providing interpretable results by letting the model attend to portions of the input image. Examples can be found in Kim and Canny (2017) and Chen et al. (2017). In the former, salient regions are extracted from a saliency model to condition the output by weighing feature maps, whereas the latter exploit a biologically inspired cognitive model. However, both are not end-to-end trainable and require separate training steps to compute attention. Differently from these approaches, we learn attention end-to-end instead of using an external source of saliency to weigh intermediate network features. Dong et al. (2021) use a transformer's self attention mechanism to correlate frames to previously observed images and infer the action to be taken. In our work, instead, we learn to generate region proposals that are scored to highlight the most relevant regions of the observed scene. This indicates how well steering controls can be predicted based on the corresponding attended regions.

Proposals have been introduced in literature for object detection tasks by leveraging low-level image characteristics to localize salient regions (Uijlings et al. 2013; Zitnick and Dollár 2014; Cuffaro et al. 2016). Learning strategies have also been proposed, such as Region Proposal Networks (RPN) (Ren et al. 2015) where the network is trained as a class-agnostic detector. Similarly, Spatial Transformer Networks (STN) (Jaderberg et al. 2015) learn to focus on salient parts of the image by learning affine transformations instead of generating proposals. In this work we integrate both RPNs and STNs in our visual attention module to highlight regions relevant for the driving taks.

3 Problem statement

We address the autonomous driving problem in urban environments with a vision-based imitation learning strategy. In imitation learning an agent is trained to learn a policy π by attempting to replicate demonstrations D performed by an expert (Attia and Dayan 2018). Demonstrations are made of

the ith state observation z_i and the action performed at that instant a_i. Therefore a demonstration is an input–output pair $D = (z_i, a_i)$. The goal is to learn a policy function π capable of mapping the observations to output actions $\pi : Z \rightarrow A$. Here Z is the set of possible observations and A represents the possible actions (Argall et al. 2009). In an automotive context, the expert is a human driver and the policy to be learned is "*driving safely*". In general, the agent has access only to a representation of the surrounding environment, e.g., an RGB frame of the scene from an egocentric point of view. Actions, instead, are driving controls that allow the vehicle to follow the desired path, e.g. steering angle and throttle. In practice, the imitation learning framework is made possible by pre-recorded driving sessions performed by expert drivers, which yield a collection of *(frame, driving-controls)* pairs, acting as demonstrations.

This end-to-end approach is particularly effective for its ability to learn a safe driving policy without the need to provide explicit safe driving rules, such as '*to turn right follow the right edge of the roadway*'. Yet this hinders a true understanding of the reasons why a certain driving action is adopted and this contrasts with the increasing demand for explainability that is rapidly emerging in the autonomous driving domain.

3.1 Model overview

The goal of our proposed model is to learn a driving policy capable of imitating the expert by producing the steering angle values required to comply with a given high level command. Following prior works (Codevilla et al. 2018; Sauer et al. 2018), to ensure system scalability on high-level input commands, we divide our architecture into multiple branches, with a separate head for each type of command.

We equip each branch with a visual attention mechanism to make the model interpretable so to explain the estimated maneuvers. In particular, we rely on a region proposal function $\mathcal{R}$ that generates Regions of Interest (RoI) in the input image. The visual attention module then scores each RoI, assigning an importance to each region, thus highlighting portions of the image that are relevant for addressing the driving task. The model, which is end-to-end trainable, first extracts a global feature map f from the input image with a convolutional neural network backbone. Based on f, the region proposal function $\mathcal{R}$ outputs a set of R relevant RoIs $\{BB_i\}_{i=1}^{R}$ in the form of bounding boxes $BB_i = [x_i, y_i, h_i, w_i]$, where x_i and y_i denote the upper-left coordinates and h_i and w_i its height and width.

At this point, we obtain a region descriptor r_i for each RoI in the image by applying RoI Pooling (Girshick 2015) on the feature map generated by the convolutional backbone. The pooling is applied only on the portion of the convolutional feature map identified by the ith bounding box. Pooled features are then weighed using an attention layer and concatenated together, yielding a global descriptor which is condensed into a lower dimensional space with a dense block. The block is followed by a dense regressor that generates steering angle predictions as outputs.

The multi-head architecture that we propose is depicted in Fig. 1.

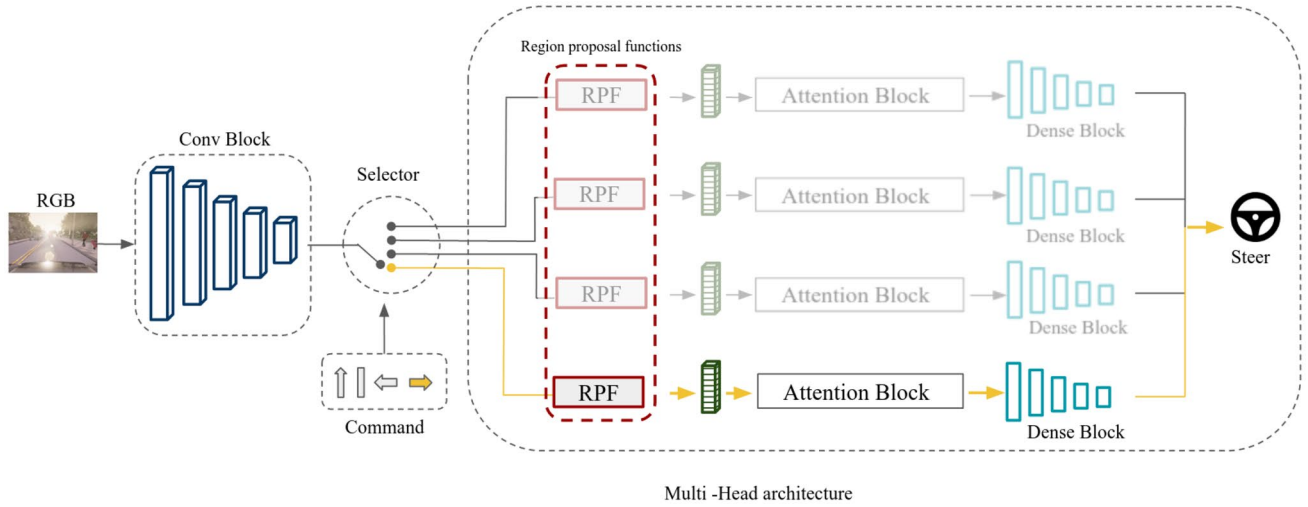

Fig. 1 A convolutional backbone generates a feature map. Then, a region proposal function extracts RoIs that are pooled and weighed by an attention layer. Separate region proposal and attention modules are trained for each high level command in order to focus on different regions and output the appropriate steering angle

3.2 Visual attention

An attention scheme allows a model to attend only on relevant parts of the input. When the input is an image, this translates into a task driven saliency over pixels or image regions. In automotive, explicitly attending to specific parts of the observed scene aids the decision making process by taking into account environmental cues or surrounding objects, such as turns, intersections or other agents. Thus, a visual attention must weigh RoIs by generating a probability distribution according to their importance for the driving task. To make the attention task-driven, we learn it directly from the data with an end-to-end training, that is using an attention layer inside a model that generates driving commands from the input image. To foster the model's interpretability, we use a branched architecture with multiple prediction heads. Each head generates driving steering angles for different high level commands. As a consequence, by integrating a separate attention layer in each head, we obtain different ways of attending to elements in the scene, depending on the high level command.

To perform attention over image regions, we first flatten all RoI-pooled region features r_i and we stack them in a single vector r, describing the whole scene. We feed r to a dense layer that generates a different logit for each image region. Logit are then normalized using a softmax activation function, yielding a set of RoI weights $\alpha = \alpha_1, \ldots, \alpha_R$, where R is the number of regions: $\alpha = Softmax(r \cdot W_a + b_a)$. Here W_a and b_a are respectively the weights and biases of the fully connected attention layer. We use the softmax function since it dampens the logits while sharpening the most relevant ones. This means that the model is able to concentrate only on a restricted subset of regions. We obtain a final feature r_a, where the importance of each region is weighed by the respective attention value, by concatenating RoI features scaled with α: $r_a = concat(r_i \cdot \alpha_i)$ *for* $i \in [1, \ldots, R]$. Our attention block architecture is shown in Fig. 2.

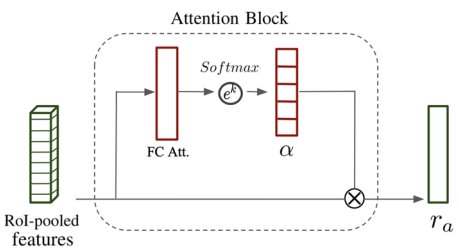

Fig. 2 Attention block. A weight vector α is generated by a linear layer with softmax activation. The final descriptor is a concatenation of RoI features scaled with attention weights

3.3 Multi-head architecture

To comply with different high level commands, the autonomous vehicle must exhibit different behaviors. For example, when reaching an intersection the agent can turn left or right or can keep going straight, depending on the route it has to follow. To keep our architecture as flexible and extensible as possible, we structure our model as a branched multi-head regressor where each head predicts steering values for different high level commands. This allows new heads to be easily plugged in to address additional high level commands. This is a common trend which has been shown to outperform single-headed models (Codevilla et al. 2018; Sauer et al. 2018). We use an attention layer in each head, so that the model can learn to addend to different cues, depending on the command. This solution has the advantage of improving the explainability of the model, since the attention maps are conditioned on the commands, highlighting what is important for different tasks.

To decide which head to use, we feed the command as input to the model as we use it as a selector to pick the correct branch. This has the effect of guiding backpropagation only through the part of the model that is actually used to generate the output while training the network. At the same time, the convolutional backbone is shared among commands, so it will get updated for ecah sample, regardless of the high-level command. This enables an end-to-end training of the model.

4 Region proposals

The shared convolutional backbone, after extracting features from input images, is followed by a RoI pooling layer (Girshick 2015). Each Region of Interest generated by the region proposal function $\mathcal{R}$ can exhibit different sizes and aspect ratios. The RoI pooling layer extracts a fixed-size descriptor r_i for each proposal by dividing the region into a number of cells on which a pooling function is applied. Here we adopt the max-pooling operator over 4×4 cells.

RoI generation is a fundamental step in our pipeline since extracting good RoIs allows the attention mechanism, explained in Sect. 3.2, to correctly select salient regions of the image. We propose different formulations for $\mathcal{R}$. First, we use a static grid of fixed boxes at different scales. We then propose a fully learnable formulation, making $\mathcal{R}$ a neural network capable of generating task-driven dynamic proposals, depending on image content. In this case, we follow two alternative approaches to build the region proposal function, relying on Region Proposal Networks (RPN) (Ren et al. 2015) and Spatial Transformer Networks (STN) (Jaderberg et al. 2015). In the following we provide an overview

of the static and dynamic formulations to build the region proposal function $\mathcal{R}$.

4.1 Static proposals

The simplest formulation to obtain a set of region proposals, is to let $\mathcal{R}$ yield a static grid of fixed handcrafted RoIs. To obtain meaningful RoIs we use a multi-scale regular grid spanning across the image. Here, we assume the image to have height H and width W. We generate the grid by sliding variable size boxes on the input image with a given stride. For this purpose we used four types of windows as explained in Fig. 3:

- BIG^H—horizontal boxes with height $H/2$ and width W; these boxes cover the whole width of the image, spanning from top to bottom with a 60px vertical stride;
- BIG^V—vertical boxes with height H, width $W/2$ and an horizontal stride equal to $W/2$; it yields two regions dividing the image into a left and right side;
- MEDIUM—boxes with height $H/2$ and width $W/2$, with an area equal to a quarter of the image; we slide the box on the two horizontal halves of the image with an horizontal stride of 60px;
- SMALL—boxes with height $H/2$ and width $W/4$, with a 3px stride in the horizontal and vertical directions.

The shape of the four boxes is designed to consider different aspects of the scene. The BIG scale is coarse and is intended to focus on structural elements in the scene (e.g. vertical for traffic signs or buildings and horizontal for forthcoming intersections). The MEDIUM and SMALL scales instead focus on smaller details such as approaching vehicles or distant turns. Overall, we use a grid of 48 image regions: 2 BIG^V, 6 BIG^H, 8 MEDIUM and 32 SMALL.

4.2 Dynamic proposals

Casting $\mathcal{R}$ as a static proposal generator has evident limitations since we are posing a strong prior on the regions that the model can attend to. However, dynamically generating region proposals is not trivial. In fact, once proposals are cropped, all the spatial information is lost. Indeed, when the proposals are static, the network learns to correlate the relative position of each feature with its underlying semantics. This is possible since the ith feature to be attended will always correspond to the same spatial coordinates. The fact that the model is learning positional cues based on the order in which RoI features are presented to the attention model can be easily demonstrated by shuffling the boxes during inference. In Sect. 9, we show that by doing so, the model is unable to generate meaningful steering commands.

Fig. 3 Four sliding windows are used to generate a multi-scale grid of RoIs. Colors indicate the box type: BIG^V (red), BIG^H (green), MEDIUM (yellow) SMALL (blue)

To overcome this limitation, we simply concatenate a spatial cue to the input image by adding two additional channels containing x and y normalized frame coordinates. This allows the model to take into account absolute RoI positions and be invariant to proposal ordering. We adopt this technique to enable the generation of dynamic proposals that vary depending on the image content. We propose two different, alternative, formulations for $\mathcal{R}$: Region Proposal Networks (Girshick 2015) and Spatial Transformer Networks (Jaderberg et al. 2015).

Region Proposal Networks—Region Proposal Network (RPN) (Ren et al. 2015) is a convolutional architecture for generating proposals given a convolutional feature map of an image. An RPN has a regression and a classification layer. The regression layer modifies some anchors with predefined sizes and aspect ratios to generate bounding boxes coordinates. The classification head instead is used to assign objectness scores to proposals.

Although RPN represents an effective method to generate RoIs, its original formulation appears to have limitations: (i) a strong supervision signal is needed, namely ground truth bounding-boxes; (ii) it relies on RoI Pooling to extract features, which is non differentiabile. The authors originally overcame these limitations for an object detection task by adopting a two step-training, i.e. pretraining the RPN with ground truth class-agnostic boxes. Since we do not know a-priori which regions might be considered useful by the model, for the purpose of our work it is essential that the model can be trained end-to-end and discover relevant proposals in a task driven fashion.

As a solution, instead of standard RoI Pooling, we use a differentiable RoI pooling layer called *Precise RoI Pooling* (Jiang et al. 2018). Precise RoI Pooling is an integration-based pooling strategy based on bilinear interpolation and allows the gradient to be backpropagated through the bounding-box coordinates. This makes the regression head differentiable and allows the proposal generation to be fully task-driven.

In addition, unlike Faster R-CNN (Girshick 2015), which generates a large number of boxes and then thresholds them based on the predicted objectness, we completely remove the classification head and retain all the generated proposals.

This stems from the fact that without a direct supervision, the classification head is unable to provide effective objectness scores. To control the number of boxes we act on the stride of the convolutions and on the number of anchors used to generate the proposals. Further details are provided in Sect. 5.

Spatial Transformer Networks—Spatial Transformer Networks (STN) (Jaderberg et al. 2015) allow a model to learn spatial transformations on input feature maps. The effectiveness of STN derives from the fact that transformations are learned without a direct supervision, in a task-driven fashion. In detail STN is made up of three main blocks. A *Localization Network* is responsible for predicting the parameters θ of the transformation matrix T_θ. It takes a feature map as input and is formed by a convolutional or fully connected block followed by a regressor. A *Grid Generator* then uses the affine transformation matrix T_θ to output a parameterized sampling grid $T_\theta(G)$, where G is a regular grid corresponding to image coordinates. The final output transformation is obtained using a *Grid Sampler* which applies the sampling grid $T_\theta(G)$ on the input feature map. This operation is achieved through bilinear interpolation. Overall, the transformation performed by the STN is an affine transformation that maps points in the input feature map into warped positions. Therefore, using an STN as region proposal function, does not require a RoI Pooling step as the transformation directly outputs the features of the region of interest.

We constrain the transformation to avoid skew and rotation, making it of the form:

$$T_\theta = \begin{bmatrix} s & 0 & T_x \\ 0 & s & T_y \end{bmatrix} \tag{1}$$

where s is the scale factor, and T_x, T_y are the translation parameters. In our model, each branch dedicated to a high-level command is equipped with an STN generating R spatial transformations (e.g. proposals).

5 Training details

The shared backbone is composed of five convolutional layers with ELU activations. The first three layers have respectively 24, 36 and 48 5×5 kernels with stride 2 and are followed by two other layers with 64 3×3 filters with stride 1. All input images are resized to a resolution of 200×88 pixels. In the RPN model we control the number of boxes by changing the stride in the convolutional block. In particular we use stride 2 to generate 108 boxes. Anchor size and aspect ratio parameters used for training are respectively $\{64, 80, 100\}$ and $\{0.5, 1\}$. RPN produces a feature map of size 1024 for each box. A linear reduction layer is used to reduce its size to 512. Unlike the other models, STN does not use RoI pooling but a localization network instead: a convolutional layer with 64 channels with stride 1 and ReLU activation function, followed by a linear layer with dimension 64 and *tanh* activation function. Finally a linear layer predicts the transformation parameters for the desired number of proposals. STN produces a feature map of size 4608 for each proposal. A linear reduction layer is used to reduce its size to 512. As for the attention block, it is composed of fully connected layers of decreasing size, i.e. 1024, 512, 128, 10. A final fully connected layer outputs the steering angle necessary to comply with the given high level command. As a loss function we use Mean Squared Error (MSE) for all architectures. To train the model, we use Adam as optimizer with a learning rate of 0.0001 for 50 epochs and a batch size of 64.

6 Dataset

To train and evaluate our autonomous agent, we use the CARLA Simulator (Dosovitskiy et al. 2017). Carla is an open-source platform conceived and designed for research in autonomous driving. It provides a realistic reconstruction of an urban and suburban environment that includes two Towns. It also offers the possibility of setting different weather conditions and daytimes. We use data from Codevilla et al. (2018), in which Town01 is used as training and Town02 as validation. The dataset was recorded using four different weather conditions. For each example in the train set, measurements concerning the value of steering, throttle, brake, and high-level command are provided. To test the abilities of an autonomous agent, Codevilla et al. (2018) also released a test benchmark composed of separate driving episodes. The benchmark is goal-oriented: for each episode the autonomous agent is asked to reach a goal point on the map, given a starting point, within a certain time limit. Both Towns are included. For each Town the benchmark is divided into four tasks: (i) Go Straight—driving along a straight road; (ii) One Turn—the destination is reached by making either a right or left turn; (iii) Navigation—to reach the destination an agent has to drive along a longer route, in which there might be several turns; (iv) Navigation dynamic—the same as Navigation but with other vehicles and pedestrians. For each task, there are 25 episodes, replicated in six different weather conditions, four of which already seen in training, and two used only for testing. In total, the benchmark consists of 600 episodes for Town01 and 600 for Town02 for a total of 1200 episodes.

7 Experimental results

In this section we discuss experimental results obtained by our proposed models. First of all, we compare the driving success rates on the CARLA Benchmark using different

types of attention. In Table 1 we compare results of a vanilla model without attention against models with static and dynamic proposals. The model without attention does not generate any proposals and directly feeds the global feature map to the final multi-head architecture. Explicitly modeling attention leads to significantly increased driving performance. Interestingly, the model with the static proposal function obtains good results, improving by a large margin compared to the attention-less baseline. As for the dynamic proposal functions, the STN proposal obtains the best overall success rate, with the notable exception of *New weather and new Town* where static proposals yield slightly better results. Surprisingly, RPN perform worse than all the other proposal-based methods. We impute this to multiple factors: (i) proposals are generated based on local features corresponding to anchors, whereas STN performs global reasoning; (ii) anchors which must be handcrafted, thus diminishing the expressiveness of the model; (iii) training an RPN without direct supervision on box positions may not be enough, especially since there is no specific foreground object the box coordinate regressor can adhere to.

It must be noted that in this work we are not interested in obtaining the best results on a driving benchmark, but rather to offer a comprehensive analysis on how attention mechanisms can be integrated into a driving model and which benefits this can provide. Nonetheless, in Table 2, we provide a comparison with other state of the art methods. Since our focus is on explainability and attention, our model does not come equipped with bells and whistles like data augmentation or exploiting high-level input representations (e.g., depth, semantic segmentation).

An additional characterization of the results, further motivating the success of STN over RPN, can be given looking at the proposals generated by the models along with their attention weights. This is of particular interest since it provides a degree of explainability with respect to the outputs of the model. Figure 4 shows attention heatmaps obtained by cumulating the top 5 proposals over the entire validation set consisting of 74,000 frames. STN offers spatially fine grained explanations compared to the other methods. Attention is focused on the horizon and sidewalks for the follow command, the center line for the straight command and the bottom left centerline for the left and right commands. For the turn left command the model looks at the centerline when the road is still straight, yet also focusing ahead on the left side to anticipate the curve. On the contrary, for the turn right command, the model keeps the lane by following the bottom left part of the centerline and looks on the right side when the curve is visible. Interestingly, often in Carla the centerline interrupts at intersections. The model is therefore exploiting this cue to perform the driving task. This bimodal distribution of attention is even more visible in Fig. 5, which shows the distribution of all the proposals generated for each model. For RPN, instead, the distribution of the boxes is mostly focused on the left and right edge of the road at the horizon, using bigger and coarser RoIs. The static model, on the other hand, yields an attention map that has a regular, axis aligned distribution, with the right side of the road getting higher attention for all the high level commands. Samples of attentions in single frames are shown in Fig. 6. A frame-by-frame breakdown of a right turn is also provided to show how the attention changes when approaching an intersection in Fig. 7.

Table 1 Percentage of completed tasks using static proposals, RPN and STN

	Training conditions					New weather				
Model	Straight	Turn	Nav	Nav. D	Mean	Straight	Turn	Nav	Nav. D	Mean
Ours no att	**100**	91	80	79	87.50	**100**	96	76	72	86.00
Ours static	**100**	95	91	89	93.75	**100**	**100**	92	92	96.00
Ours RPN	**100**	93	84	82	89.75	**100**	84	82	80	86.50
Ours STN	**100**	**100**	**95**	**90**	**96.25**	100	**100**	**94**	**94**	**97.00**
	New Town					New weather and new town				
Model	Straight	Turn	Nav	Nav. D	Mean	Straight	Turn	Nav	Nav. D	Mean
Ours no att	94	37	25	18	43.50	92	52	52	36	58.00
Ours static	**99**	**79**	53	40	67.75	**100**	**88**	67	**56**	**77.75**
Ours RPN	90	60	50	48	62.00	98	64	50	48	65.00
Ours STN	95	77	**67**	**51**	**72.50**	90	80	**70**	54	73.50

Bold values indicate the best results

We also show a baseline without attention

Table 2 Comparison with the state of the art, measured in percentage of completed tasks

	Training conditions					New weather				
Model	Straight	Turn	Nav	Nav. D	Mean	Straight	Turn	Nav	Nav. D	Mean
MP$^{\diamond}$ Dosovitskiy et al. (2017)	98	82	80	77	84.25	100	95	94	89	94.50
MT$^{\diamond\blacklozenge}$ Li et al. (2018)	98	87	81	81	86.75	100	88	88	80	89.00
CAL† Sauer et al. (2018)	100	97	92	83	93.00	100	96	90	82	92.00
EF$^{\blacklozenge}$ Xiao et al. (2019)	99	99	92	89	94.75	96	92	90	90	92.00
CEF$^{\blacklozenge}$ Haris and Glowacz (2022)	98	99	92	89	94.50	96	94	91	86	87.25
MTL$^{\diamond\blacklozenge}$ Ishihara et al. (2021)	100	100	100	99	99.75	100	99	97	97	98.25
CILRS* Codevilla et al. (2019)	96	92	95	92	93.75	96	96	96	96	96.00
RL Dosovitskiy et al. (2017)	89	34	14	7	86.00	16	2	2	36	26.50
IL Codevilla et al. (2018)	95	89	86	83	88.25	98	90	84	82	88.50
EF-RGB Xiao et al. (2019)	96	95	87	84	90.50	84	78	74	66	75.50
CIRL Liang et al. (2018)	98	97	93	82	92.50	**100**	94	86	80	90.00
Ours STN	**100**	**100**	**95**	**90**	**96.25**	**100**	**100**	**94**	**94**	**97.00**
	New town					New weather and new town				
Model	Straight	Turn	Nav	Nav. D	Mean	Straight	Turn	Nav	Nav. D	Mean
MP$^{\diamond}$ Dosovitskiy et al. (2017)	92	61	24	24	51.25	50	50	47	44	47.70
MT$^{\diamond\blacklozenge}$ Li et al. (2018)	**100**	81	72	53	76.50	96	82	78	62	79.50
CAL† Sauer et al. (2018)	93	82	70	64	77.25	94	72	68	64	74.50
EF$^{\blacklozenge}$ Xiao et al. (2019)	96	81	90	87	88.50	96	84	90	94	91.00
CEF$^{\blacklozenge}$ Haris and Glowacz (2022)	96	79	90	84	87.25	97	82	92	94	91.00
MTL$^{\diamond\blacklozenge}$ Ishihara et al. (2021)	99	98	93	91	95.25	99	99	96	91	96.25
CILRS* Codevilla et al. (2019)	96	84	69	66	78.75	96	92	92	90	92.50
RL Dosovitskiy et al. (2017)	74	12	3	2	22.75	68	20	6	4	24.50
IL Codevilla et al. (2018)	97	59	40	38	58.50	80	48	44	42	53.50
EF-RGB Xiao et al. (2019)	82	69	63	**57**	67.75	84	76	56	44	65.00
CIRL Liang et al. (2018)	**100**	71	53	41	66.25	98	82	68	**62**	80.00
Ours STN	95	77	**67**	51	**72.50**	90	80	**70**	54	73.50

Bold values indicate the best results

Additional sources of data used by a model are identified by superscripts: ⧫ (depth), ⋄ (semantic segmentation), † (temporal modeling), ⋆ (different training data). The best result per task is shown in bold for methods using only RGB frames as input

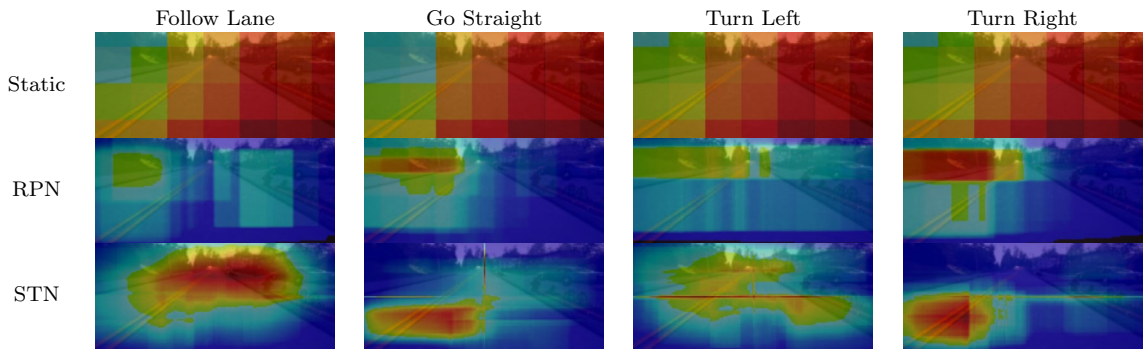

Fig. 4 Heatmap for the top five boxes, ranked by attention. The heatmap is the result of cumulating top proposals over the entire validation set

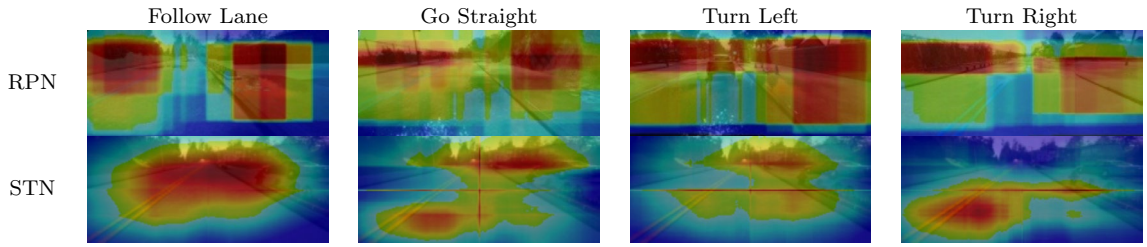

Fig. 5 Heatmap proposal distribution. The heatmap is obtained cumulating all proposals irrespectively of their attention weight on the validation set

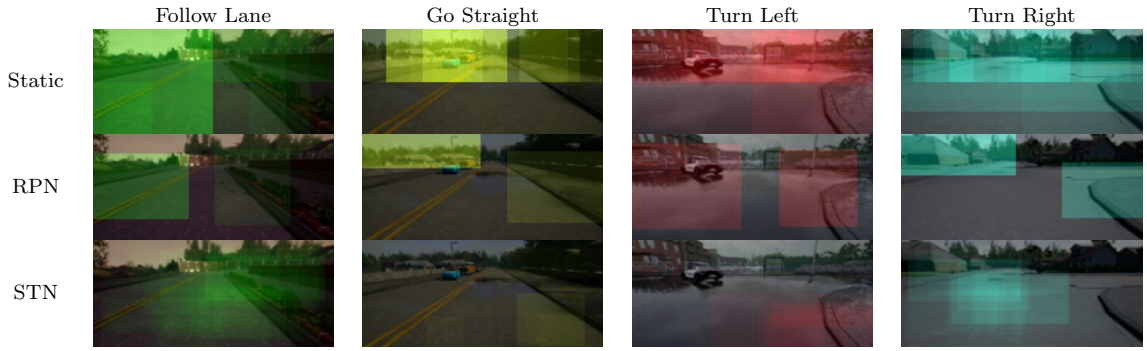

Fig. 6 Attention patterns for the proposed models

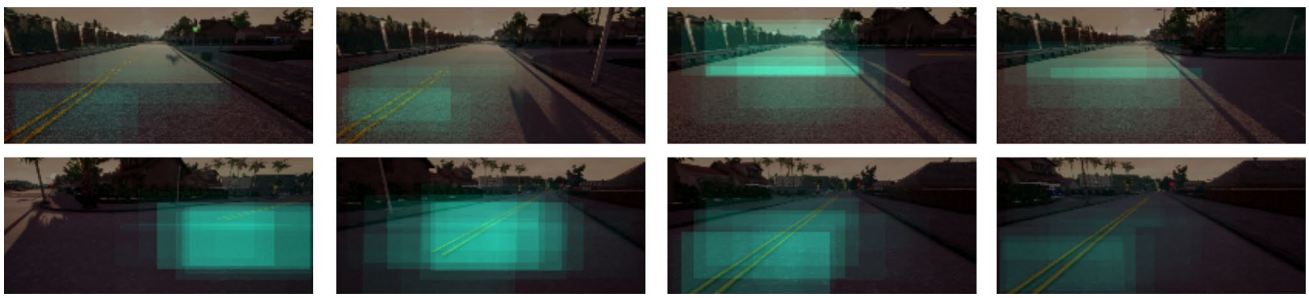

Fig. 7 Turn right example. The attention is first on the centerline to keep the lane. When the centerline disappears, the model looks at the whole road horizontally. As soon as the intersection is visible the model shifts its attention to the right and again on the centerline to start following the new lane. Best viewed in color on a screen

8 Comparison with feature attribution methods

Our proposed model has been specifically designed to provide visual explanations of its predictions. What the model is performing is therefore an importance attribution of regions in the pixel space. Nonetheless, several methods for feature attribution exist in literature and have been successfully used to provide ex-post visual explanations of deep learning models (Bach et al. 2015; Selvaraju et al. 2017; Shrikumar et al. 2017; Sundararajan et al. 2017). Here, we compare the attributions provided by our model, previously discussed in Fig. 6, with off-the-shelf feature attribution methods, namely *DeepLift* (Shrikumar et al. 2017), *LRP* (Bach et al. 2015) and *Integrated Gradients* (Sundararajan et al. 2017). We apply such methods on the baseline architecture without the explicit attention module.

Qualitative results for the explainability models described above are shown in Fig. 8. DeepLift (Shrikumar et al. 2017), LRP (Bach et al. 2015) and Integrated Gradients (Sundararajan et al. 2017) generate a sparse attention that focuses mostly on the road surface. Despite this being a comprehensible behavior for a model without attention, it does not suggest much in terms of interpretability. Overall attention

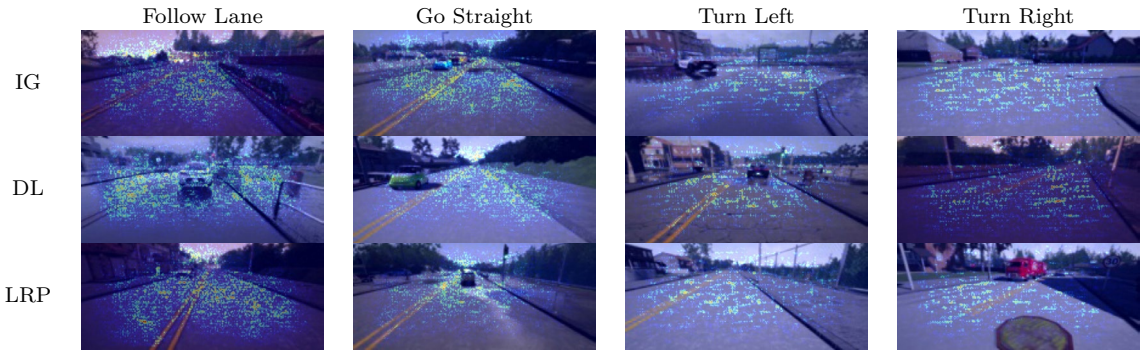

Fig. 8 Each row shows ex-post explanations through attention maps, divided by high level command. We report attention maps for Integrated Gradients (IG) (Sundararajan et al. 2017), DeepLift (DL) (Shrikumar et al. 2017) and LRP (Bach et al. 2015)

maps are quite similar between the various explainability models, moreover it is difficult to identify patterns that can help to understand the motivations of a particular behavior. This experiment suggests that using attention mechanisms based on proposal generation rather than ex-post explanation methods allows us to obtain a visual explanation of what leads to a prediction. Furthermore, it should be considered that these ex-post explainability methods have limitations. In particular, they are not well suited for regression models. Evidence of these limitations can be found in the work of Letzgus et al. (2021). This consideration is further confirmation that using an explicit attention mechanism leads to considerable benefits in terms of interpretability.

9 Ablation studies

In this section we validate the importance of some of the components of our proposed models. Here, we perform the experiments on a subset of the CARLA driving benchmark, namely testing only using the *Training conditions* and *New weather* splits. First of all we experimentally validate the intuition, introduced in Sect. 4.2, according to which the model based on a static proposals learns to derive spatial information from the order of the boxes. In Table 3 we show that simply shuffling the order in which the boxes are presented makes the model unable to emit meaningful steering commands. To overcome this limitation, we retrain the model adding two additional channels to the input, representing normalized spatial coordinates ranging from 0 to 1. The overall success rate is almost on par with the original method, even if the order of the boxes is shuffled.

We now study the effect of the number of boxes in our dynamic proposal functions. Controlling the number of boxes with STN is straightforward since the STN can be modified to generate multiple transformation matrices. For RPN instead, since we do not use the classification head, we change the number of boxes by modifying the stride of the convolutions and the number of anchors. In particular we use $stride = 1$ to generate 432 boxes and $stride = 3$ to generate 72 boxes. The reference RPN model, used in Table 1, has $stride = 2$ to generate 108 boxes, which is comparable to STN which uses 100 boxes. In Table 4 we show the results of the models varying the number of boxes for STN and RPN. In both cases, when using approximately 100 proposals we obtain the best results.

10 Retrieving failed episodes

Indeed, the proposed attention mechanism is beneficial to the explainability of the driving behaviour and can also support the identification of anomalous conditions anticipating possible driving failures. Inspired by prior work (Yang et al.

Table 3 Ablation study. Shuffling the proposals makes the model unable to drive since spatial relations are broken. This issue is avoided by adding spatial coordinates as inputs, which makes the model able to retain approximately its original accuracy even when proposals are shuffled

Training conditions				New weather		
Task	Static	Shuffle	Shuffle + Coords	Static	Shuffle	Shuffle + Coords
Straight	100	40	100	100	40	100
One turn	95	12	97	100	8	100
Navigation	91	8	85	92	4	84
Navigation dynamic	89	4	84	92	4	84

Table 4 Ablation study. We vary the number of proposals produced by STN and RPN. Both STN and RPN perform better using a number of boxes around 100. In general, STN can obtain higher driving accuracy even with a low number of proposals, compared to RPN

	Training conditions						New weather					
	STN			RPN			STN			RPN		
Num boxes	50	100	300	72	108	432	50	100	300	72	108	432
Straight	100	100	100	100	100	100	100	100	100	100	100	100
One turn	100	100	98	80	93	93	94	100	96	84	84	84
Navigation	88	95	91	66	84	84	88	94	86	70	82	72
Navigation dynamic	87	90	90	64	82	84	84	94	86	68	80	76

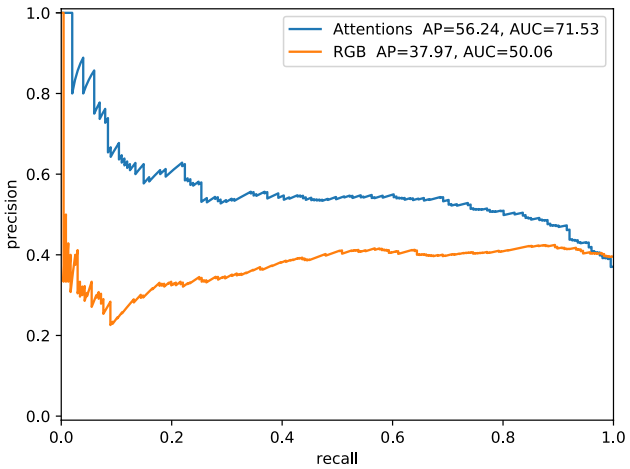

Fig. 9 Precision-recall curves for detecting failed episodes

2022), we address the problem of detecting anomalies using convolutional autoencoders. We have trained two networks with the same architecture, the first fed with RGB frames and the second with attention maps produced by our model. We generate attention maps using the STN model, overlaying each generated box on a reference black image, weighing the RoI with the corresponding attention value.

To test the models we used a test set consisting of 600 episodes extracted from the CARLA benchmark, 300 of which were successfully completed by the model. Failed episodes contain collisions with pedestrians, cars, other objects and/or unusual maneuvers. Our assumption is that failed episodes will contain out of the ordinary events, making the predicted attention anomalous. We thus leverage the reconstruction error of the autoencoders to detect such anomalies. We treat this task as a retrieval task, aiming at automatically identifying failed episodes. To evaluate the task, for each episode we take the maximum reconstruction error and use it to generate precision recall curves, as shown in Fig. 9. The model trained on attention maps reaches an AUC on the precision-recall curve of 71.53, while the model trained on RGB only 50.06. Similarly, computing Average Precision, we obtain 56.24 using attention maps and 37.97 with RGB frames. This experiment demonstrates that modeling attention is

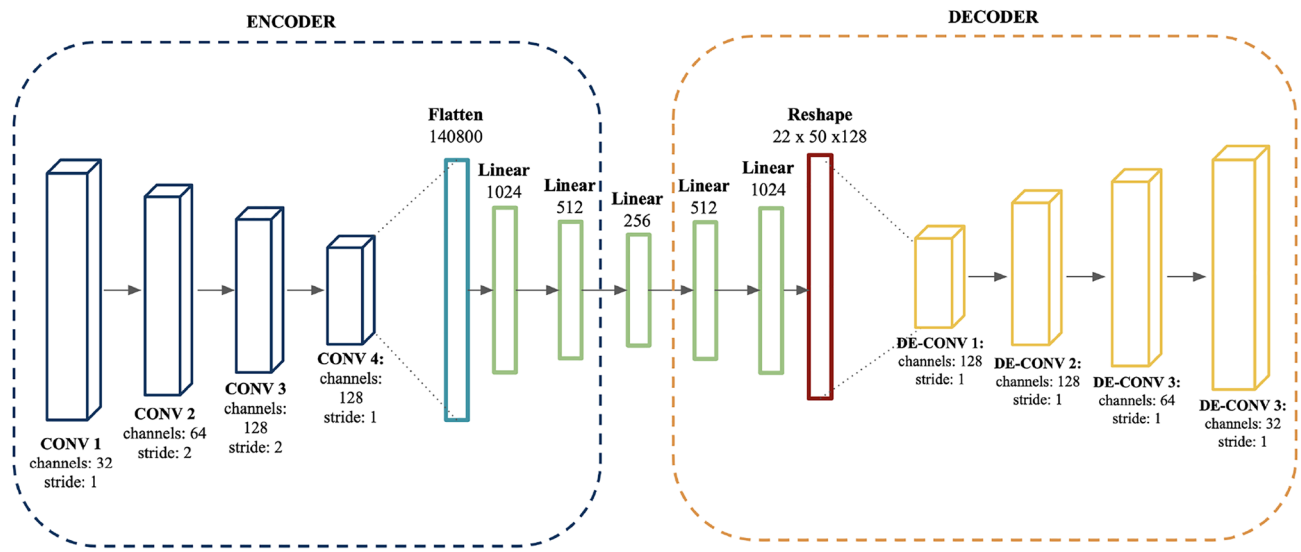

Fig. 10 AutoEncoder architecture

also effective in retrieving challenging episodes, which can be used to retrain the model and improve its performance. Details of the autoencoder architecture are shown in Fig. 10.

11 Conclusions

In this paper we describe the architecture of an end-to-end trainable driving system capable of generating driving controls (e.g. steering angle) from an RGB frame representing the scene captured by a vision system. The architecture is designed to implement an attention mechanism that induces a selection of regions of the RGB frame that are most relevant for the prediction of the driving controls. This contributes to the explainability of the model by showing which regions of the observed scene are used for driving. Such an indication can help improving the training process but also entails the potential for detecting anomalies in the observed scene, anticipating potential driving failures. The accuracy of different region proposal mechanisms is reported by measuring the driving success rate on the CARLA Benchmark and demonstrating that region proposal by STN (Jaderberg et al. 2015) yields the best overall success rate compared both to RPN (Ren et al. 2015) and to a fixed frame partitioning scheme. Reported experiments also demonstrate that the proposed attention mechanism leads to considerable benefits in terms of interpretability compared to methods providing ex-post visual explanations of deep learning models (Shrikumar et al. 2017; Bach et al. 2015; Sundararajan et al. 2017).

Funding Open access funding provided by Università degli Studi di Firenze within the CRUI-CARE Agreement. This work was supported by the European Commission under European Horizon 2020 Programme, grant number 951911—AI4Media.

Code availability Not applicable.

Declarations

Conflict of interest None.

References

Anderson P, He X, Buehler C, Teney D, Johnson M, Gould S, Zhang L (2018) Bottom-up and top-down attention for image captioning and visual question answering. In: Proc. of the IEEE conference on computer vision and pattern recognition, pp 6077–6086

Argall BD, Chernova S, Veloso M, Browning B (2009) A survey of robot learning from demonstration. Robot Auton Syst 57(5):469–483

Attia A, Dayan S (2018) Global overview of imitation learning. arXiv:1801.06503v1

Bach S, Binder A, Montavon G, Klauschen F, Müller K-R, Samek W (2015) On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PLoS ONE 10(7):e0130140

Berlincioni L, Becattini F, Galteri L, Seidenari L, Del Bimbo A (2019) Road layout understanding by generative adversarial inpainting. In: Inpainting and denoising challenges. Springer, pp 111–128

Berlincioni L, Becattini F, Seidenari L, Del Bimbo A (2021) Multiple future prediction leveraging synthetic trajectories. In: 2020 25th International conference on pattern recognition (ICPR). IEEE, pp 6081–6088

Bojarski M, Del Testa D, Dworakowski D, Firner B, Flepp B, Goyal P, Jackel LD, Monfort M, Muller U, Zhang J, et al (2016)End to end learning for self-driving cars. arXiv preprint arXiv:1604.07316

Bojarski M, Yeres P, Choromanska A, Choromanski K, Firner B, Jackel L, Muller U (2017) Explaining how a deep neural network trained with end-to-end learning steers a car. arXiv preprint arXiv:1704.07911

Chen C, Steff A, Kornhauser A, Xiao J (2015) Deepdriving: learning affordance for direct perception in autonomous drivings. In: Proc. of the IEEE international conference on computer vision, pp 2722–2730

Chen S, Zhang S, Shang J, Chen B, Zheng N (2017) Brain-inspired cognitive model with attention for self-driving cars. IEEE Trans Cogn Dev Syst 11(1):13–25

Codevilla F, Müller M, López A, Koltun V, Dosovitskiy A (2018) End-to-end driving via conditional imitation learning. In: 2018 IEEE international conference on robotics and automation (ICRA). IEEE, pp 4693–4700

Codevilla F, Santana E, López AM, Gaidon A (2019) Exploring the limitations of behavior cloning for autonomous driving. In: Proc. of the IEEE/CVF international conference on computer vision, pp 9329–9338

Cuffaro G, Becattini F, Baecchi C, Seidenari L, Bimbo AD (2016) Segmentation free object discovery in video. In: European conference on computer vision. Springer, pp 25–31

Cultrera L, Seidenari L, Becattini F, Pala P, Del Bimbo A (2020) Explaining autonomous driving by learning end-to-end visual attention. In: Proc. of the IEEE/CVF conference on computer vision and pattern recognition workshops, pp 340–341

Dong J, Chen S, Zong S, Chen T, Labi S (2021) Image transformer for explainable autonomous driving system. In: 2021 IEEE international intelligent transportation systems conference (ITSC). IEEE, pp 2732–2737

Dosovitskiy A, Ros G, Codevilla F, López A (2017) Carla: an open urban driving simulator. In: Conference on robot learning (CoRL), PMLR, pp 1–16

Eraqi HM, Moustafa MN, Honer J (2017) End-to-end deep learning for steering autonomous vehicles considering temporal dependencies. arXiv preprint arXiv:1710.03804

George L, Buhet T, Wirbel E, Le-Gall G, Perrotton X (2018) Imitation learning for end to end vehicle longitudinal control with forward camera. arXiv preprint arXiv:1812.05841

Girshick R (2015) Fast r-cnn. In: Proc. of the IEEE international conference on computer vision, pp 1440–1448
Greco A, Rundo L, Saggese A, Vento M, Vicinanza A (2022) Imitation learning for autonomous vehicle driving: How does the representation matter? In: International conference on image analysis and processing. Springer, pp 15–26
Guo Z, Zhang S, Han S, Lin Y (2021) Improving the environmental adaptability of conditional imitation learning driving model. In: 2021 International conference on high performance big data and intelligent systems (HPBD &IS). IEEE, pp 271–275
Haris M, Glowacz A (2022) Navigating an automated driving vehicle via the early fusion of multi-modality. Sensors 22(4):1425
Ishihara K, Kanervisto A, Miura J, Hautamaki V (2021)Multi-task learning with attention for end-to-end autonomous driving. In: Proc. of the IEEE/CVF conference on computer vision and pattern recognition, pp 2902–2911
Jaderberg M, Simonyan K, Zisserman A et al (2015) Spatial transformer networks. In: Proc of Adv Neural Inf Process Syst, 28
Jiang B, Luo R, Mao J, Xiao T, Jiang Y (2018) Acquisition of localization confidence for accurate object detection. In: Proc. of the European conference on computer vision (ECCV), pp 784–799
Kim J, Canny J (2017) Interpretable learning for self-driving cars by visualizing causal attention. In: Proc. of the IEEE international conference on computer vision, pp 2942–2950
Kim J, Moon S, Rohrbach A, Darrell T, Canny J (2020) Advisable learning for self-driving vehicles by internalizing observation-to-action rules. In: Proc. of the IEEE/CVF conference on computer vision and pattern recognition, pp 9661–9670
Lee D, Liu S, Gu J, Liu M-Y, Yang M-H, Kautz J (2018) Context-aware synthesis and placement of object instances. In: Proc of Adv Neural Inf Process Syst, 31
Letzgus S, Wagner P, Lederer J, Samek W, Müller K-R, Montavon G (2021) Toward explainable AI for regression models. arXiv preprint arXiv:2112.11407
Li Z, Motoyoshi T, Sasaki TOK, Sugano S (2018) Rethinking self-driving: multi-task knowledge for better generalization and accident explanation ability. arXiv preprint arXiv:1809.11100
Liang X, Wang T, Yang EX L (2018) Cirl: controllable imitative reinforcement learning for vision-based self-driving. In: Proc of European conference on computer vision (ECCV), pp 584–599
Lim BY, Dey AK (2009) Assessing demand for intelligibility in context-aware applications. In: Proc. of the 11th international conference on Ubiquitous computing, pp 195–204
Marchetti F, Becattini F, Seidenari L, Del Bimbo A (2022) Smemo: social memory for trajectory forecasting. arXiv preprint arXiv:2203.12446
Omeiza D, Webb H, Jirotka M, Kunze L (2021) Explanations in autonomous driving: a survey. IEEE Trans Intell Transport Syst 23(8):10142–10162
Orden Tv, Visser A (2021) End-to-end imitation learning for autonomous vehicle steering on a single-camera stream. In: International conference on intelligent autonomous systems. Springer, pp 212–224
Ren S, He K, Girshick R, Sun J (2015) Faster r-cnn: towards real-time object detection with region proposal networks. In: Advances in neural information processing systems, pp 91–99
Sauer A, Savi-nov N, Geiger A (2018) Conditional affordance learning for driving in urban environments. In: Proc of Conference on robot learning (CoRL), pp 237–252
Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D (2017) Grad-cam: visual explanations from deep networks via gradient-based localization. In: Proc. of the IEEE international conference on computer vision, pp 618–626
Shrikumar A, Greenside P, Kundaje A (2017) Learning important features through propagating activation differences. In: International conference on machine learning. PMLR, pp 3145–3153
Sundararajan M, Taly A, Yan Q (2017) Axiomatic attribution for deep networks. In: International conference on machine learning. PMLR, pp 3319–3328
Uijlings JR, Van De Sande KE, Gevers T, Smeulders AW (2013) Selective search for object recognition. Int J Comput Vision 104(2):154–171
Xiao Y, Codevilla F, Gurram A, Urfalioglu O, Lopez AM (2019) Multimodal end-to-end autonomous driving. arXiv preprint arXiv:1906.03199
Xu H, Gao Y, Yu F, Darrell T (2017) End-to-end learning of driving models from large-scale video datasets. In: Proc. of the IEEE conference on computer vision and pattern recognition, pp 2174–2182
Xu Y, Yang X, Gong L, Lin H-C, Wu T-Y, Li Y, Vasconcelos N (2020) Explainable object-induced action decision for autonomous vehicles. In: Proc. of the IEEE/CVF conference on computer vision and pattern recognition, pp 9523–9532
Yang Z, Zhang Y, Yu J, Cai J, Luo J (2018) End-to-end multi-modal multi-task vehicle control for self-driving cars with visual perceptions. In: 2018 24th International conference on pattern recognition (ICPR). IEEE, pp 2289–2294
Yang J, Xu R, Qi Z, Shi Y (2022) Visual anomaly detection for images: a systematic survey. Proc Comput Sci 199:471–478
Zablocki É, Ben-Younes H, Pérez P, Cord M (2021) Explainability of vision-based autonomous driving systems: review and challenges. arXiv preprint arXiv:2101.05307
Zhang J, Cho K (2016) Query-efficient imitation learning for end-to-end autonomous driving. arXiv preprint arXiv:1605.06450
Zitnick CL, Dollár P (2014) Edge boxes: locating object proposals from edges. In: European conference on computer vision. Springer, pp 391–405