

# Explaining Autonomous Driving by Learning End-to-End Visual Attention

Luca Cultrera   Lorenzo Seidenari   Federico Becattini   Pietro Pala   Alberto Del Bimbo  
MICC, University of Florence

name.surname@unifi.it

## Abstract

*Current deep learning based autonomous driving approaches yield impressive results also leading to in-production deployment in certain controlled scenarios. One of the most popular and fascinating approaches relies on learning vehicle controls directly from data perceived by sensors. This end-to-end learning paradigm can be applied both in classical supervised settings and using reinforcement learning. Nonetheless the main drawback of this approach as also in other learning problems is the lack of explainability. Indeed, a deep network will act as a black-box outputting predictions depending on previously seen driving patterns without giving any feedback on why such decisions were taken.*

*While to obtain optimal performance it is not critical to obtain explainable outputs from a learned agent, especially in such a safety critical field, it is of paramount importance to understand how the network behaves. This is particularly relevant to interpret failures of such systems.*

*In this work we propose to train an imitation learning based agent equipped with an attention model. The attention model allows us to understand what part of the image has been deemed most important. Interestingly, the use of attention also leads to superior performance in a standard benchmark using the CARLA driving simulator.*

## 1. Introduction

Teaching an autonomous vehicle to drive poses a hard challenge. Whereas the problem is inherently difficult, there are several issues that have to be addressed by a fully fledged autonomous system. First of all, the vehicle must be able to observe and understand the surrounding scene. If data acquisition can rely on a vast array of sensors [17], the understanding process must involve sophisticated machine learning algorithms [10, 20, 5, 33, 39]. Once the vehicle is able to perceive the environment, it must learn to navigate under strict constraints dictated by street regulations and, most importantly, safety for itself and others. To learn such a skill, autonomous vehicles can be instructed

to imitate driving behaviors by observing human experts [41, 27, 18, 13, 44]. Simulators are often used to develop and test such models, since machine learning algorithms can be trained without posing a hazard for others [15]. In addition to learning to transform visual stimuli into driving commands, a vehicle also needs to estimate what other agents are or will be doing [35, 4, 30, 32].

Even when it is able to drive correctly, under unusual conditions an autonomous vehicle may still commit mistakes and cause accidents. When this happens, it is of primary importance to assess what caused such a behavior and intervene to correct it. Therefore, explainability in autonomous driving assumes a central role, which is often not addressed as it should. Common approaches in fact, although being effective in driving tasks, do not offer the possibility to inspect what generates decisions and predictions. In this work we propose a conditional imitation learning approach that learns a driving policy from raw RGB frames and exploits a visual attention module to focus on salient image regions. This allows us to obtain a visual explanation of what is leading to certain predictions, thus making the model interpretable. Our model is trained end-to-end to estimate steering angles to be given as input to the vehicle in order to reach a given goal. Goals are given as high-level commands such as drive straight or turn at the next intersection. Since each command reflects in different driving patterns, we propose a multi-headed architecture where each branch learns to perform a specialized attention, looking at what is relevant for the goal at hand. The main contributions of the paper are the following:

- We propose an architecture with visual attention for driving an autonomous vehicle in a simulated environment. To the best of our knowledge we are the first to explicitly learn attention in an autonomous driving setting, allowing predictions to be visually explained.
- We show that the usage of visual attention, other than providing an explainable model, also helps to drive better, obtaining state of the art results on the CARLA driving benchmark [15].

The work is described as follows: In Section 2 the related

works are described to frame our method in the current state of the art; in Section 3 the method is shown using a top-down approach, starting from the whole architecture and then examining all the building blocks of our model. In Section 4 results are analyzed and a comparison with the state of the art is provided. Finally, in Section 7 conclusions are drawn.

## 2. Related Works

Our approach can be framed in the imitation learning line of investigation of autonomous driving. Our model improves the existing imitation learning framework with an attention layer. Attention in imitation learning is not extensively used, therefore in the following we review previous work in imitation learning and end-to-end driving and attention models.

### 2.1. Imitation Learning & end-to-end driving

One of the key approaches for learning to execute complex tasks in robotics is to observe demonstrations, performing so called imitation learning [2, 22]. Bojarski et al. [6] and Codevilla et al. [13] were the first to propose a successful use of imitation learning for autonomous driving.

Bojarski et al [6] predict steering commands for lane following and obstacle avoiding tasks. Differently, the solution proposed by Codevilla et al. [13] performs conditional imitation learning, meaning that the network emits predictions conditioned on high level commands. A similar branched network is proposed by Wang et al. [40]. Liang et al. [27] instead use reinforcement learning to perform conditional imitation learning.

Sauer et al. [36], instead of directly linking perceptual information to controls, use a low-dimensional intermediate representation, called affordance, to improve generalization. A similar hybrid approach, proposed by Chen et al. [9], maps input images to a small number of perception indicators relating to affordance environment states.

Several sensors are often available and simulators [13] provide direct access to depth and semantic segmentation. Such information is leveraged by several approaches [41, 26, 42]. Recently Xiao et al. [41] improved over [13, 26] adding depth information to RGB images obtaining better performances. Li et al. [26] exploit depth and semantic segmentation at training time performing multi-task learning while in [41] depth is considered as an input to the model. Xu et al. [42] show that it is possible to learn from a crowd sourced video dataset using multiple frames with an LSTM. They also demonstrate that predicting semantic segmentation as a side task improves the results. Multi-task learning proves effective also in [43] for end-to-end steering and throttle prediction. Temporal modelling is also used by [16] and [18] also using Long Short-Term Memory Networks.

Zhang and Cho [44] extend policy learning allowing a system to gracefully fallback into a safe policy avoiding to fall into dangerous states.

### 2.2. Attention models

Attention has been used in classification, recognition and localization tasks, as in [8, 23, 37]. The authors of [7] propose an attention model for object detection and counting on drones, while [28] uses attention for small object detection. Other examples of attention models used for image classification are [38, 29, 12, 45]. Often, attention-based networks are used for video summarization [21, 31] or Image Captioning task, as in the work of Anderson et al. [1] or Cornia et al. [14]. There are some uses of attention based models in autonomous driving [24, 11]. Attention has been used to improve interpretability in autonomous driving by Jinkyu and Canny [24]. Salient regions extracted from a saliency model are used to condition the network output by weighing network feature maps with such attention weights. Chen et al. [11] adopt a brain inspired cognitive model. Both are multi-stage approaches in which attention is not computed in an end-to-end learning framework.

Our approach differs from existing ones since we learn attention directly during training in an end-to-end fashion instead of using an external source of saliency to weigh intermediate network features. We use proposal regions that our model associates with a probability distribution to highlight salient regions of the image. This probability distribution indicates how well the corresponding regions predict steering controls.

## 3. Method

To address the autonomous driving problem in an urban environment, we adopt an Imitation Learning strategy to learn a driving policy from raw RGB frames.

### 3.1. Imitation learning

Imitation learning is the process by which an agent attempts to learn a policy  $\pi$  by observing a set of demonstrations  $D$ , provided by an expert  $E$  [3]. Each demonstration is characterized by an input-output couple  $D = (z_i, a_i)$ , where  $z_i$  is the  $i$ -th state observation and  $a_i$  the action performed in that instant. The agent does not have direct access to the state of the environment, but only to its representation. In the simplest scenario, an imitation learning process, is a direct mapping from state observations to output actions. In this case the policy to be learned is obtained by the mapping [2]:

$$\pi : Z \rightarrow A \quad (1)$$

where  $Z$  represents the set of observations and  $A$  the set of possible actions. In an autonomous driving context, the

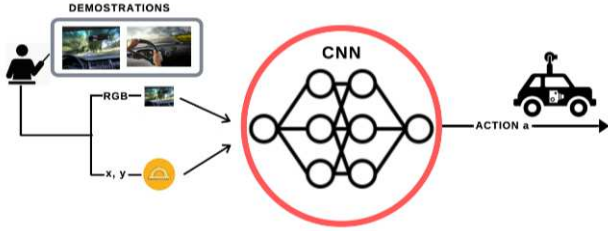


Figure 1: Imitation learning

expert  $E$  is a driver and the policy is “safe driving”. Observations are RGB frames describing the scene captured by a vision system and actions are driving controls such as throttle and steering angle. Therefore, the imitation learning process can be addressed through convolutional neural networks (CNN). Demonstrations are a set of  $(frame, driving-controls)$  pairs, acquired during pre-recorded driving sessions performed by expert drivers. The high level architecture of this approach is shown in Figure 1.

### 3.2. Approach

Our Imitation Learning strategy consists in learning a driving policy that maps observed frame to the steering angle needed by the vehicle to comply with a given high level command. We equip our model with a visual attention mechanisms to make the model interpretable and to be able to explain the estimated maneuvers.

The model, which is end-to-end trainable, first extracts features maps from RGB images with a fully convolutional neural network backbone. A regular grid of regions, obtained by sliding boxes of different scales and aspect ratios over the image, selects Regions of Interest (RoI) from the image. Using a RoI Pooling strategy, we then extract a descriptor for each region from the feature maps generated by the convolutional block. An attention layer assigns weights to each RoI-pooled feature vector. These are then combined together with a weighted sum to build a descriptor that is fed as input to the last block of the model, which includes dense layers of decreasing size and that emits steering angle predictions.

The system is composed of a shared convolutional backbone, followed by a multi-head visual attention block and a final regressor block. The proposed multi-head architecture is shown in Figure 2. The different types of high-level commands that can be given as input to the model include:

- *Follow Lane*: follow the road
- *Go straight*: choose to go straight to an intersection
- *Turn Left*: choose to turn left to an intersection
- *Turn right*: choose to turn right to an intersection

#	Output dim	Channels	Kernel size	Stride
1	$298 \times 130 \times 24$	24	5	2
2	$147 \times 63 \times 36$	36	5	2
3	$72 \times 30 \times 48$	48	5	2
4	$70 \times 28 \times 64$	64	3	1
5	$68 \times 26 \times 64$	64	3	1

Table 1: Convolutional backbone architecture. The size of the input images is  $600 \times 264$ .

In the following, we present in detail each module that is employed in the model.

### 3.3. Shared Convolutional Backbone

The first part of the model takes care of extracting features from input images. Since our goal is to make the model interpretable, in the sense that we want to understand which pixels contribute more to a decision, we want feature maps to preserve spatial information. This is made possible using a fully convolutional structure. In particular, this sub-architecture is composed of 5 convolutional layers: the first three layers have respectively 24, 36 and 48 kernels of size  $5 \times 5$  and stride 2 and are followed by two other layers both with  $64 \ 3 \times 3$  filters with stride 1. All convolutional layers use a Rectified Linear Unit (Relu) as activation function. Details of the convolutional backbone are summarized in Table 1. The convolutional block is followed by a RoI pooling layer which maps regions of interest onto the feature maps.

### 3.4. Region Proposals

The shared convolutional backbone, after extracting features from input images, finally conveys into a RoI pooling layer [19]. For each Region of Interest, which can exhibit different sizes and aspect ratios, the RoI pooling layer extracts a fixed-size descriptor  $r_i$  by dividing the region into a number of cells on which a pooling function is applied. Here we adopt the max-pooling operator over  $4 \times 4$  cells.

RoI generation is a fundamental step in our pipeline since extracting good RoIs allows the attention mechanism, explained in Section 3.5, to correctly select salient regions of the image. To extract RoIs from an image of size  $H \times W$  we use a multi-scale regular grid. The grid is obtained by sliding multiple boxes on the input image with a chosen step size. For this purpose we used four sliding windows with different strides as explained in Figure 3:

- $BIG^H$ : a horizontal box of size  $H/2 \times W$  that covers the whole width of the image, ranging from top to bottom with a 60px vertical stride.
- $BIG^V$ : a vertical box of size  $H \times W/2$  with horizontal stride equal to  $W/2$ , therefore yielding two regions dividing the image into a left and right side.

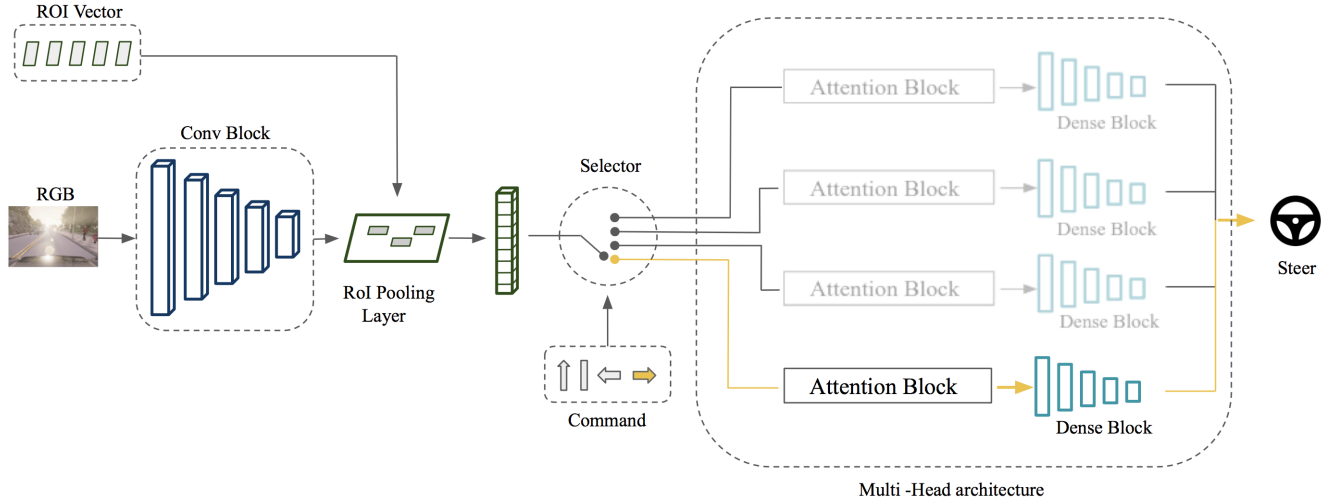


Figure 2: Architecture. A convolutional backbone generates a feature map from the input image. A fixed grid of Regions Of Interest is pooled with ROI pooling and weighed by an attention layer. Separate attention layers are trained for each high level command in order to focus on different regions and output the appropriate steering angle.

- MEDIUM: a box of size  $H/2 \times W/2$  covering a quarter of the image. The sliding window is applied on the upper and lower half of the image with an horizontal stride of 60px.
- SMALL: a square box of size  $H/2 \times W/4$ , applied with stride 30px in both directions over the image.

We identify each window type as BIG, MEDIUM and SMALL to address their scale and we use the H and V superscripts to respectively refer to the horizontal and vertical aspect ratios.

The four box types are thought to take into account different aspects of the scene. The first scale (BIG<sup>H</sup> and BIG<sup>V</sup>) is coarse and follows the structural elements in the scene (e.g. vertical for traffic signs or buildings and horizontal for forthcoming intersections). The remaining scales instead focus on smaller elements such as vehicles or distant turns.

In total we obtain a grid of 48 regions: 2 BIG<sup>V</sup>, 6 BIG<sup>H</sup>, 8 MEDIUM and 32 SMALL. Note that despite having a fixed grid may look as a limitation, this is necessary to ensure that the model understands the spatial location of each observed region. Furthermore it allows to take into account all regions at the same time to generate the final attention, which is more effective than generating independent attentions for each region. This aspect is further discussed in Section 5.

### 3.5. Visual Attention

Visual attention allows the system to focus on salient regions of the image. In an autonomous driving context, explicitly modeling saliency can help the network to take decisions considering relevant objects or environmental cues, such as other vehicles or intersections. For this purpose,



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

the attention layer must be able to weigh image regions by estimating a probability distribution over the calculated set of RoI-pooled features. We learn to estimate visual attention directly from the data, training the model end-to-end to predict the steering angle of the ego-vehicle in a driving simulator. Therefore the attention layer is trained to assign weights to image regions by relying on the relevance of each input in the driving task. In order to condition attention on high level driving commands, we adopt a different head for each command. Each head is equipped with an attention block structured as follows.

At first, region features  $r_i$  obtained by RoI-pooling are flattened and concatenated together in a single vector  $r$ . This is then fed to a fully connected layer that outputs a logit for each region. Each logit is then normalized with a softmax activation to be converted into a RoI weight  $\alpha = \alpha_1, \dots, \alpha_R$ , where  $R$  is the number of regions:

$$\alpha = \text{Softmax}(r \cdot W_a + b_a) \quad (2)$$

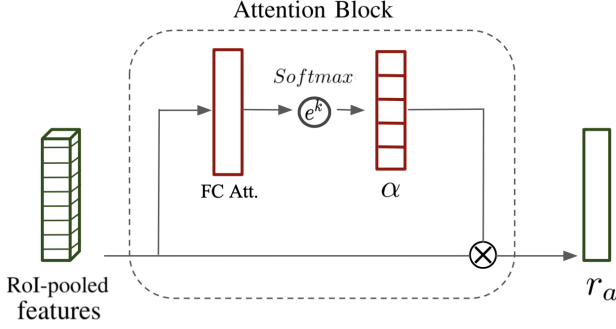


Figure 4: Attention Block. A weight vector  $\alpha$  is generated by a linear layer with softmax activation. The final descriptor is obtained as a weighted sum of RoI-pooled features.

Here  $W_a$  and  $b_a$  are the weights and biases of the linear attention layer, respectively. The softmax function helps the model to focus only on a small subset of regions, since it boosts the importance of the regions with the highest logits and dampens the contribution of all others. The final attention feature  $r_a$  is obtained as a weighted sum of the region features  $r_i$ :

$$r_a = \sum_{i=1}^R r_i \cdot \alpha_i \quad (3)$$

The architecture of the attention block adopted in each head of the model is shown in Figure 4.

### 3.6. Multi-head architecture

The model is requested to output a steering angle based on the observed environment and a high level command. High level commands encode different behaviors, such as moving straight or taking a turn, which therefore entail different driving patterns. To keep our architecture as flexible and extendable as possible, we use a different prediction head for each command, meaning that additional heads could be easily plugged in to address additional high level commands. Moreover, multi-head architectures have been shown to outperform single-headed models [13, 36]. Each command has its own specialized attention layer, followed by a dense block to output predictions. This has also the benefit of increasing explainability, since we can generate different attention maps conditioned on different commands.

The weighted attention feature  $r_a$ , produced by the attention layer, is given as input to a block of dense layers of decreasing size, i.e. 512, 128, 50, 10, respectively. Finally, a last dense layer predicts the steering angle required to comply with the given high level command.

The high level command is provided to the model along with the input image and acts as a selector to determine

which head to use. Therefore, at training time, the error is backpropagated only through the head specified by the current command and then through the shared feature extraction backbone, up to the input in an end-to-end fashion.

### 3.7. Training details

As input for the model we use  $600 \times 264$  images captured by a camera positioned in front of the vehicle. Throughout training and testing we provide a constant throttle to the vehicle, with an average speed of  $10 \text{ Km/h}$  as suggested in the CARLA driving benchmark [15]. For each image, a steering angle  $s$  represents the action performed by the driver. Therefore the loss function is calculated using Mean Squared Error (MSE) between the predicted value and the ground truth steer (4):

$$\mathcal{L} = \|s_{GT} - s_p\|^2 \quad (4)$$

Where  $s_{GT}$  and  $s_p$  represent respectively ground truth and predicted steer values. The model is trained using the Adam optimizer [25] for 20 epochs with batch size of 64 and initial learning rate of 0,0001.

### 3.8. Dataset

For training and evaluating our model, we use data from the CARLA simulator [15]. CARLA is an open source urban driving simulator, that can be used to test and evaluate autonomous driving agents. The simulator provides two realistically reconstructed towns with different day time and weather conditions.

The authors of the simulator, also released a dataset for training and testing [13]. The dataset uses the first town to collect training images and measurements and the second town for evaluation only. The training set is composed by data captured in four different weather conditions for about 14 hours of driving. For each image in the dataset, a vector of measurements is provided which includes values for steering, accelerator, brake, high level command and additional information that can be useful for training an autonomous driving system such as collisions with other agents.

To establish the capabilities of autonomous agents, CARLA also provides a driving benchmark, in which agents are supposed to drive from a starting anchor point on the map to an ending point, under different environmental conditions. The benchmark is goal oriented and is composed of several episodes divided in four main task:

1. *Straight*: drive in a straight road.
2. *One turn*: drive performing at least one turn.
3. *Navigation*: driving in an urban scenario performing several turns.

4. *Navigation Dynamic*: same scenario as *Navigation*, including pedestrians and other vehicles.

For each task, 25 episodes are proposed in several different weather conditions (seen and unseen during training). Two towns are used in the benchmark: *Town1*, i.e. the same town used for training the network, and *Town2*, which is used only at test time. In total the benchmark consists of 1200 episodes, 600 for each town.

## 4. Results

In our experiments, we train our model with the training data from [13] and evaluate the performance using the driving benchmark of CARLA [15]. In order to compare our model with the state of the art, we considered several related works that use the CARLA benchmark [15] as a testing protocol. In Table 3 we use the acronyms MP, IL, RL to refer to the CARLA [15] agents, respectively: Modular Pipeline agent, Imitation Learning agent and Reinforcement Learning agent. Note that IL was first detailed in [13] and then tested on the CARLA benchmark in [15]. With the initials CAL, CIRL and MT we refer to the results from the works of Sauer et al. [36], Liang et al. [27] and Li et al. [26]. Finally, EF indicates results from the work on multi-modal end-to-end autonomous driving system proposed by Xiao et al. [41].

When comparing results it must be taken into account which input modality is used and whether decisions are based on single frames or multiple frames. Our model bases its predictions solely on a single RGB frame. All baselines rely on RGB frames, but some use additional sources of data, namely depth and semantic segmentations. MP [15] in addition to driving commands, predicts semantic segmentation. Similarly MT [26] is built as a multi-task framework that predicts both segmentation and depth. While not using directly these sources of data as input, these models are trained with an additional source of supervision. EF [41] feeds depth images along with RGB as input to the model. To show results comparable to ours, we report also the RGB variant which does not use depth information and we refer to it as EF-RGB. All models work emitting predictions one step at a time, with the exception of CAL [36], which is trained either with LSTM, GRU or temporal convolution to model temporal dynamics and take time into account.

Table 2 reports the average success rate of episode across all tasks in the benchmark. Our method obtains state of the art results when compared to other methods that rely solely on RGB data. Overall, only EF [41] is able to obtain a higher success rate but has to feed also depth data to the model. In fact, the success rate of its RGB counterpart (EF-RGB) has a 17% drop, i.e. 8% lower than our approach.

Table 3 shows the percentage of completed episodes for each task. Results are divided according to weather condi-

tions and town. It can be seen that using attention to guide predictions allows the model to obtain good results in the evaluation benchmark, especially in the *New weather* setting where our method achieves state of the art performance in 3 tasks out of 4. We observe that methods using depth, i.e. MT [26] and EF [41], perform very well in navigation tasks, hinting that adding this additional cue will likely improve the performance of our model too.

Moreover, compared to the other methods, our model has the advantage of being explainable thanks to the attention layer. In fact, instead of treating the architecture as a black box, we are able to understand what is guiding predictions by looking at attention weights over Regions of Interest. Visual attention activations highlight which image features are useful in a goal oriented autonomous driving task, hence rejecting possible noisy features.

To underline the importance of attention, we report in Table 2 and Table 3 also a variant of our model without attention. This considers the whole image instead of performing ROI-pooling on a set of boxes, but still maintains an identical multi-head structure to emit predictions. Adding the attention layer yields to an increase in performance for each task. The most significant improvements are for navigation tasks, especially in the *Dynamic* setting, i.e. with the presence of other agents. Attention in fact helps the model to better take into account other vehicles, which is pivotal in automotive.

Examples of visual attention activations for each high level command are shown in Figure 5. It can be seen that each head of the model focuses on different parts of the scene, which are more useful for the given command. As easily imaginable, the *Turn Right* and *Turn Left* commands focus respectively on the right and left parts of the image. The model though, is able to identify where the intersection is, which allows the vehicle to keep moving forward until it reaches the point where it has to start turning. When a turning command is given and there is no intersection ahead, the model keeps focusing on the road, enlarging its attention area until it is able to spot a possible turning point. Interestingly, the *Turn Right* usually focuses on the lower part of the image and the *Turn Left* on the upper part. This is due to the fact that right turns can be performed following the right sidewalk along the curve, while for left turns the vehicle has to cross the road and merge onto the intersecting road.

The head specialized for the *Go Straight* command instead focuses on both sides of the road at once, since it has to maintain an intermediate position in the lane, without turning at any intersection. The model here is focusing on the right sidewalk and the road markings to be able to keep a certain distance from both sides.

Finally, with the *Follow Lane* command, the model looks ahead to understand whether the road will keep leading forward or if it will make a turn. An interesting case is given

Method	MP [15]	MT [26]	CAL [36]	EF [41]	RL [15]	IL [15, 13]	EF-RGB [41]	CIRL [27]	Ours no Att.	Ours
Additional data	S	S+D	T	D	-	-	-	-	-	-
Success Rate	69	83	84	<b>92</b>	27	72	75	82	69	<b>84</b>

Table 2: Success Rate averaged across all tasks. For each method we show whether additional data is used other than a single RGB frame: temporal sequence of frames (T), semantic segmentations (S), depth (D).

Training conditions										
Task	$MP^{\S}$ [15]	$MT^{\S\blacklozenge}$ [26]	$CAL^{\dagger}$ [36]	$EF^{\blacklozenge}$ [41]	RL [15]	IL [15, 13]	EF-RGB [41]	CIRL [27]	Ours no Att.	Ours
Straight	98	98	<b>100</b>	99	89	95	96	<u>98</u>	<b>100</b>	<b>100</b>
One turn	82	87	<u>97</u>	<b>99</b>	34	89	<u>95</u>	<b>97</b>	91	<u>95</u>
Navigation	80	<u>81</u>	<b>92</b>	<b>92</b>	14	86	87	<b>93</b>	80	<u>91</u>
Nav. Dynamic	77	81	<u>83</u>	<b>89</b>	7	83	<u>84</u>	82	79	<b>89</b>
New weather										
Task	$MP^{\S}$ [15]	$MT^{\S\blacklozenge}$ [26]	$CAL^{\dagger}$ [36]	$EF^{\blacklozenge}$ [41]	RL [15]	IL [15, 13]	EF-RGB [41]	CIRL [27]	Ours no Att.	Ours
Straight	<b>100</b>	<b>100</b>	<b>100</b>	<u>96</u>	86	<u>98</u>	84	<b>100</b>	<b>100</b>	<b>100</b>
One turn	<u>95</u>	88	<b>96</b>	92	16	90	78	94	<u>96</u>	<b>100</b>
Navigation	<b>94</b>	88	<u>90</u>	<u>90</u>	2	84	74	<u>86</u>	76	<b>92</b>
Nav. Dynamic	<u>89</u>	80	82	<b>90</b>	2	<u>82</u>	66	80	72	<b>92</b>
New Town										
Task	$MP^{\S}$ [15]	$MT^{\S\blacklozenge}$ [26]	$CAL^{\dagger}$ [36]	$EF^{\blacklozenge}$ [41]	RL [15]	IL [15, 13]	EF-RGB [41]	CIRL [27]	Ours no Att.	Ours
Straight	92	<b>100</b>	93	<u>96</u>	74	97	82	<b>100</b>	94	<u>99</u>
One turn	61	<u>81</u>	<b>82</b>	<u>81</u>	12	59	69	<u>71</u>	37	<b>79</b>
Navigation	24	<u>72</u>	70	<b>90</b>	3	40	<b>63</b>	<u>53</u>	25	<u>53</u>
Nav. Dynamic	24	53	64	<b>87</b>	2	38	<b>57</b>	<u>41</u>	18	40
New weather and new Town										
Task	$MP^{\S}$ [15]	$MT^{\S\blacklozenge}$ [26]	$CAL^{\dagger}$ [36]	$EF^{\blacklozenge}$ [41]	RL [15]	IL [15, 13]	EF-RGB [41]	CIRL [27]	Ours no Att.	Ours
Straight	50	<b>96</b>	<u>94</u>	<b>96</b>	68	80	84	<u>98</u>	92	<b>100</b>
One turn	50	<u>82</u>	72	<b>84</b>	20	48	76	<u>82</u>	52	<b>88</b>
Navigation	47	<u>78</u>	68	<b>90</b>	6	44	56	<b>68</b>	52	<u>67</u>
Nav. Dynamic	44	62	64	<b>94</b>	4	42	44	<u>62</u>	36	<u>56</u>

Table 3: Comparison with the state of the art, measured in percentage of completed tasks. Two variants of the proposed method are reported: the full method and a variant without attention. Additional sources of data used by a model are identified by superscripts:  $\blacklozenge$  (depth),  $\S$  (semantic segmentation),  $\dagger$  (temporal modeling). The best result per task is shown in bold and the second best is underlined, both for methods RGB-based and that use additional data.

by the presence of a T-junction under the *Follow Lane* command (first row in Figure 5). This is a command that is not present in the dataset nor the evaluation benchmark, since the behavior is not well defined as the agent might turn right or left. We observed that in these cases the model stops looking only ahead and is still able to correctly make a turn in one of the two directions, without going off road.

## 5. Ablation Study

**Box type importance** In Table 3 we have shown the importance of using attention in our model. We now investigate the importance of the attention boxes types. As explained in Sec. 3.4, we adopt four different box formats, varying scale and aspect ratio. We train a different model, selectively removing a targeted box type at a time. To carry out the ablation we use a subset of the CARLA benchmark composed of 10 episodes for the *Straight* and *One turn* tasks and 15 episodes for *Navigation* and *Navigation Dynamic*. Each episode is repeated for 6 different weather conditions

(4 conditions can also be found in the training set and 2 are testing conditions), for a total of 300 episodes. All episodes are taken from *Town1*. Results are shown in Table 4.

It emerges that all versions perform sufficiently well on simpler tasks such as *Straight* and *One turn*, although all models with less boxes still report a small drop in success rate. For more complex tasks though, the lack of  $BIG^V$  and  $MEDIUM$  lead to consistently worse driving performance. The box type that appears to be the most important is  $MEDIUM$ , which due to its aspect ratio and scale is able to focus on most elements in the scene. Interestingly, we observe that despite most of the boxes belong to the  $SMALL$  type (32 out of 48), when removing them the model is still able to obtain sufficiently high results, with a loss of 2.5 points on average. Removing these boxes though will reduce the interpretability of the model, producing much coarser attention maps.

**Fixed grid analysis** Our model adopts a fixed grid to obtain an attention map over the image. This may look like



Figure 5: Each row shows attention patterns on the same scene with different high level commands: *Follow Lane* (green), *Turn Left* (red), *Turn Right* (cyan), *Go Straight* (yellow).

Task	All	No $BIG^H$	No $BIG^V$	No MEDIUM	No SMALL	Independent RoIs
Straight	<b>100</b>	<b>100</b>	95	95	<b>100</b>	<b>100</b>
One turn	<b>97</b>	92	95	90	95	47
Navigation	<b>91</b>	86	84	83	86	45
Nav. Dynamic	<b>91</b>	88	81	83	88	39

Table 4: Ablation study selectively removing a box type. Box types refer to the regions depicted in Figure 3:  $BIG^H$  (Green),  $BIG^V$  (Red), MEDIUM (Yellow), SMALL (Blue). We also evaluate our model with attention scores generated independently for each RoI.

a limitation of our architecture, restricting its flexibility. Whereas to some extent this is certainly true, designing an attention layer with a variable grid, i.e. with boxes that change in number and shape, is not trivial. Generating a variable amount of boxes, e.g. using a region proposal [34], would require to process each box independently, depriving the attention layer of a global overview of the scene. The main issue lays in the lack of spatial information about each box: the model is indeed able to observe elements such as vehicles, traffic lights or lanes, but does not know where they belong in the image without this position being explicitly specified.

To demonstrate the inability of the model to work without a fixed grid, we modified our attention layer to emit attention scores for each RoI-pooled feature independently.

This means that instead of concatenating all features and feeding them to a single dense layer, we adopt a smaller dense layer, shared across all RoIs, to predict a single attention logit. All logits are then concatenated and jointly normalized with a softmax to obtain the attention vector  $\alpha$ .

We show the results obtained by this model in Table 4. The only task that this architecture is able to successfully address is *Straight*. As soon as the model is required to take a turn, it is unable to perform the maneuver and reach its destination. On the other hand, using a fixed grid, allows the model to learn a correlation between what is observed and where it appears in the image and jointly generating attention scores for each box. A flexible grid with variable boxes is currently an open issue and we plan to address it in future work.

## 6. Acknowledgements

We thank NVIDIA for the Titan XP GPU used for this research.

## 7. Conclusions

In this paper we have presented an autonomous driving system based on imitation learning. Our approach is equipped with a visual attention layer that weighs image regions and allows predictions to be explained. Moreover we have shown how adopting attention, the model improves its driving capabilities, obtaining state of the art results.



## References

- [1] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6077–6086, 2018.
- [2] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.
- [3] A. Attia and S. Dayan. Global overview of imitation learning. *arXiv 1801.06503v1*, 2018.
- [4] Federico Becattini, Lorenzo Seidenari, Lorenzo Berlincioni, Leonardo Galteri, and Alberto Del Bimbo. Vehicle trajectories from unlabeled data through iterative plane registration. In *International Conference on Image Analysis and Processing*, pages 60–70. Springer, 2019.
- [5] Lorenzo Berlincioni, Federico Becattini, Leonardo Galteri, Lorenzo Seidenari, and Alberto Del Bimbo. Road layout understanding by generative adversarial inpainting. In *Inpainting and Denoising Challenges*, pages 111–128. Springer, 2019.
- [6] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.
- [7] Y. Cai, D. Du, L. Zhang, L. Wen, W. Wang, Y. Wu, and S. Lyu. Guided attention network for object detection and counting on drones. *arXiv preprint arXiv:1909.113071*, 2019.
- [8] C. Cao, X. Liu, Y. Yang, Yu Y., Z. Wang, J. Wang, Y. Huang, L. Wang, C. Huang, W. Xu, D. Ramanan, and T. S. Huang. Look and think twice: Capturing top-down visual attention with feedback convolutional neural networks. *Proceedings of the IEEE international conference on computer vision*, pages 2956–2964, 2015.
- [9] C. Chen, A. Steff, A. Kornhauser, and J. Xiao. Deepdriving: Learning affordance for direct perception in autonomous drivings. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2722–2730, 2015.
- [10] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.
- [11] S. Chen, S. Zhang, J. Shang, B. Chen, and N. Zheng. Brain-inspired cognitive model with attention for self-driving cars. *IEEE Transactions on Cognitive and Developmental Systems*, 2017.
- [12] Y. Chen, D. Zhao, L. Lv, and C. Li. A visual attention based convolutional neural network for image classification. In *2016 12th World Congress on Intelligent Control and Automation (WCICA)*, pages 764–769, 2016.
- [13] F. Codevilla, M. Müller, A. López, V. Koltun, and A. Dosovitskiy. End-to-end driving via conditional imitation learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–9, 2018.
- [14] M. Cornia, L. Baraldi, and R. Cucchiara. Show, control and tell: A framework for generating controllable and grounded captions. *arXiv preprint arXiv:1811.10652*, 2018.
- [15] A. Dosovitskiy, G. Ros, F. Codevilla, and A. López. Carla: An open urban driving simulator. In *Conference on Robot Learning (CoRL)*, 2017.
- [16] H. M. Eraqi, M. N. Moustafa, and J. Honer. End-to-end deep learning for steering autonomous vehicles considering temporal dependencies. *arXiv preprint arXiv:1710.03804*, 2017.
- [17] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.
- [18] L. George, T. Buhet, E. Wirbel, G. Le-Gall, and X. Perrotton. Imitation learning for end to end vehicle longitudinal control with forward camera. *arXiv preprint arXiv:1812.05841*, 2018.
- [19] R. Girshick. Fast r-cnn. *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.
- [20] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.
- [21] X. S. Hua, L. Lu, H. J. Zhang, and H. District. A generic framework of user attention model and its application in video summarization. *IEEE Transaction on multimedia*, 7(5):907–919, 2005.
- [22] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):21, 2017.
- [23] S. Jetley, N. A. Lord, N. Lee, and P. H. Torr. Learn to pay attention. *arXiv preprint arXiv:1804.02391*, 2018.
- [24] K. Jinkyu and J. Canny. Interpretable learning for self-driving cars by visualizing causal attention. *Proceedings of the IEEE international conference on computer vision*, 2017.
- [25] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [26] Z. Li, T. Motoyoshi, T. Ogata, K. Sasaki, and S. Sugano. Rethinking self-driving: Multi-task knowledge for better generalization and accident explanation ability. in *European Conference on Computer Vision (ECCV)*, 2018.
- [27] X. Liang, T. Wang, and E. Xing L. Yang. Cirl: Controllable imitative reinforcement learning for vision-based self-driving. in *European Conference on Computer Vision (ECCV)*, 2018.
- [28] J. S. Lim, M. Astrid, H. J. Yoon, and S. I. Lee. Small object detection using context and attention. *arXiv preprint arXiv:1912.06319*, 2019.
- [29] Y. Luo, M. Jiang, and Q. Zhao. Visual attention in multi-label image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2109.
- [30] Yuexin Ma, Xinge Zhu, Sibozhang, Ruigang Yang, Wenping Wang, and Dinesh Manocha. Trafficpredict: Trajectory prediction for heterogeneous traffic-agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6120–6127, 2019.

- [31] Y. F. Ma, L. Lu, H.J. Zhang, and M. Li. A user attention model for video summarization. In *Proceedings of the tenth ACM international conference on Multimedia*, pages 532–542, 2002.
- [32] Francesco Marchetti, Federico Becattini, Lorenzo Seidenari, and Alberto Del Bimbo. Mantra: Memory augmented networks for multiple trajectory prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.
- [33] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, 2017.
- [34] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.
- [35] Nicholas Rhinehart, Rowan McAllister, Kris Kitani, and Sergey Levine. Precog: Prediction conditioned on goals in visual multi-agent settings. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2821–2830, 2019.
- [36] A. Sauer, N. Savi-nov, and A. Geiger. Conditional affordance learning for driving in urban environments. in *Conference on Robot Learning (CoRL)*, 2018.
- [37] D. Walther, U. Rutishauser, C. Koch, and P. Perona. On the usefulness of attention for object recognition. In *Workshop on Attention and Performance in Computational Vision at ECCV*, pages 96–103, 2004.
- [38] F. Wang, M. Jiang, C. Qian, S. Yang., C. Li, H. Zhang, X. Wang, and X. Tang. Residual attention network for image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3156–3164, 2107.
- [39] Heng Wang, Bin Wang, Bingbing Liu, Xiaoli Meng, and Guanghong Yang. Pedestrian recognition and tracking using 3d lidar for autonomous vehicle. *Robotics and Autonomous Systems*, 88:71–78, 2017.
- [40] Q. Wang, L. Chen, and W. Tian. End-to-end driving simulation via angle branched network. *arXiv preprint arXiv:1805.07545.*, 2018.
- [41] Y. Xiao, F. Codevilla, A. Gurram, O. Urfalioglu, and A. M. Lopez. Multimodal end-to-end au-tonomous driving. *arXiv preprint arXiv:1906.03199*, 2019.
- [42] H. Xu, Y. Gao, F. Yu, and T. Darrell. End-to-end learning of driving models from large-scale video datasets. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2174–2182, 2017.
- [43] Z. Yang, J. Yu Y. Zhang, and J. Luo J. Cai. 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)*, pages 2722–2730, 2016.
- [44] J. Zhang and K. Cho. Query-efficient imitation learning for end-to-end autonomous driving. *arXiv preprint arXiv:1605.06450*, 2016.
- [45] D. Zoran, M. Chrzanowski, P. S. Huang, S. Goyal, A. Mott, and P. Kohl. Towards robust image classification using sequential attention models. *arXiv preprint arXiv:1912.02184*, 2019.