
Explainable Multi-Camera 3D Object Detection with Transformer-Based Saliency Maps

Till Beemelmanns, Wassim Zahr, Lutz Eckstein

RWTH Aachen University

{till.beemelmanns, lutz.eckstein}@ika.rwth-aachen.de

wassim.zahr@rwth-aachen.de

Abstract

Vision Transformers (ViTs) have achieved state-of-the-art results on various computer vision tasks, including 3D object detection. However, their end-to-end implementation also makes ViTs less explainable, which can be a challenge for deploying them in safety-critical applications, such as autonomous driving, where it is important for authorities, developers, and users to understand the model’s reasoning behind its predictions. In this paper, we propose a novel method for generating saliency maps for a DetR-like ViT with multiple camera inputs used for 3D object detection. Our method is based on the raw attention and is more efficient than gradient-based methods. We evaluate the proposed method on the nuScenes dataset using extensive perturbation tests and show that it outperforms other explainability methods in terms of visual quality and quantitative metrics. We also demonstrate the importance of aggregating attention across different layers of the transformer. Our work contributes to the development of explainable AI for ViTs, which can help increase trust in AI applications by establishing more transparency regarding the inner workings of AI models.

1 Introduction

Vision Transformers (ViTs) have made significant strides in the field of computer vision, achieving state-of-the-art results on a wide range of scene understanding tasks for automated driving. Transformer-based 3D object detection was successfully applied for LiDAR [1], multi-view cameras [2, 3, 4, 5] or for multi-modal input [6]. The idea is that the transformers’ attention mechanism is able to capture a global understanding of the scene and therefore generates more accurate detections and also eliminates the need for handcrafted postprocessing or object fusion steps. This end-to-end approach offers many advantages, but it also presents a challenge in terms of explainability [7]. This challenge becomes particularly crucial when considering the deployment of ViTs in safety-critical applications like autonomous driving. In such scenarios, it is essential for authorities, developers, and users to have a clear understanding of the model’s reasoning behind its predictions.

The generation of saliency maps is a commonly employed method for enhancing the explainability of DNNs. Saliency maps reveal the most critical areas that are relevant for the model’s output. For CNN networks, many approaches exists that either use expensive perturbation-based methods [8, 9, 10, 11], use the networks’ gradients [12, 13] or apply manual propagation rules [14] to derive the relevancy maps. Transformer-based approaches use attention maps as a source of explainability. This method has been applied to NLP tasks [15], image classification [16] and object detection [17, 18]. However, there remains a gap in the literature when it comes to exploring an approach for 3D object detection in the context of AD with multi-sensory input.

In this work, we propose the first saliency map generation approach for a multi-camera transformer model, aiming to enhance our understanding of the model’s behavior and provide valuable insights

into which regions of input images are most influential in determining object detection results. We propose a simple and effective method to generate saliency maps which outperforms gradient-based methods. We validate the effectiveness of our approach through comprehensive perturbation tests. See Figure 1 for an overview. Our main contributions are the following:

- We propose a novel method for generating saliency maps for a transformer-based multi-camera 3D object detector. This method is based on the raw cross-attention and is more efficient than gradient-based methods.
- We demonstrate the importance of aggregating attention across different layers of the transformer by using extensive perturbation test. This aggregation helps to produce comprehensive and informative saliency maps.

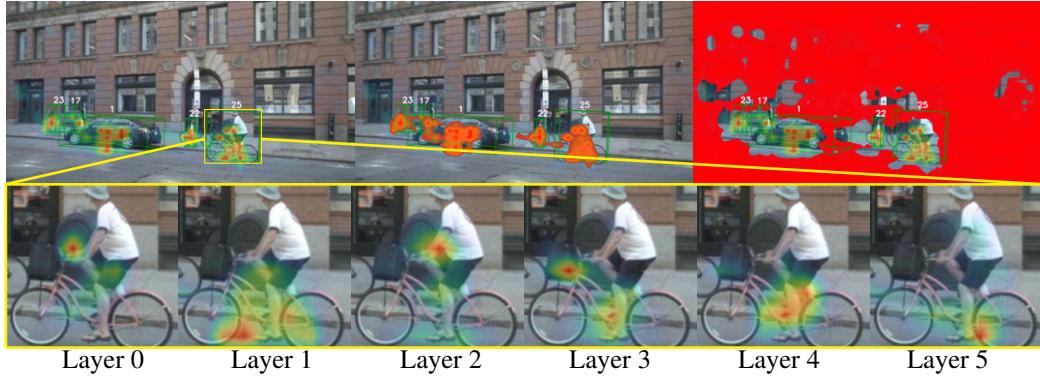


Figure 1: This work aims at generating saliency maps as visual explanations for a multi-camera transformer-based 3D object detector. We perform extensive perturbation tests to evaluate raw attention and gradient-based methods. We conduct positive (top-middle) and negative (top-right) perturbation tests, where certain parts of the six input camera images are masked (red) to measure the effectiveness of each approach. We found that it is necessary to aggregate attention across all transformer layers (bottom) to produce saliency maps that are comprehensive.

2 Related Work

2.1 Explainable AI

Transparent models or white-box models can be easily interpreted by humans straight from their structure [19]. For example, in linear regression models, the weight assigned to each feature directly indicates its importance. Similarly, decision trees provide a clear path from the root node to the leaf nodes. However, applying those models to high-resolution image data is usually not efficient.

Post-hoc explanation methods are used to understand how more complex models, such as DNNs, SVM, and CNNs, make their predictions [19]. For example, Grad-CAM [13] is a post-hoc explanation method for CNNs, which uses gradients of the target class in the final convolutional layer to create a heatmap highlighting important regions in the input image that contributed to the prediction.

Model-specific explainability techniques are specific to certain models, using their unique structures for explanation purposes. For example, a CNN can be interpreted using Layer-wise Relevance Propagation (LRP). This method pushes the output decision backward through the layers to the input, giving a relevance score to each input feature [14]. Utilizing LRP can be challenging because it requires a specific implementation for each architecture.

Model-agnostic techniques do not depend on detailed knowledge of the model’s structure. Methods like Local Interpretable Model-Agnostic Explanations (LIME) [8] and SHapley Additive exPlanations (SHAP) fall under this group [20]. They approximate the local decision boundary of any model and can assign scores to the importance of input features, but they are also computationally inefficient.

2.2 Saliency Maps

Most explainability methods in Computer Vision generate *saliency maps*, revealing the most critical or salient areas for image comprehension [12, 13, 16, 17, 21].

Perturbation-based methods pinpoint key features or regions in input data by systematically altering them and monitoring the effect on the prediction outcome by the model. LIME [8] is a local approximation to the decision boundary of complex classification models. Fong et al. [9, 10] proposed a framework to unveil explanations by identifying the specific image fragment that affects a classifier’s decision. Randomized Input Sampling for Explanations (RISE) [11] generates saliency maps by randomly masking portions of an image and aggregating the resulting outputs. However, these methods entail expensive data perturbation and multiple model forward passes, hindering their efficient use for an application in real-time scenarios with a multi-camera setup.

Gradient-based methods determine the output category’s gradient concerning the input image. This gradient can be visualized and interpreted as a saliency map. Grad-CAM [13] distinguishes between classes based on these visual explanations, which is particularly important for object detection. For ViTs, Grad-CAM generates class-specific visualizations by multiplying the attention maps with the corresponding gradients [16, 17].

LRP [14] distributes the model’s prediction back to its input features by traversing the network backwardly. Each neuron’s relevance in a layer is calculated based on its contribution to the neurons in the subsequent layer. This relevance propagation continues layer by layer until it reaches the input layer. However, applying LRP to transformers has proven challenging due to the complex interaction patterns within the attention mechanism, and its model-specific approach [15].

Attention mechanism in transformer models assigns scores that reflect the significance of various segments within a sequence of input tokens, allowing the model to focus on the most relevant input. *Attention Rollout* interprets these attention scores across layers to give a more comprehensive understanding of a transformer’s decision-making process [15, 16]. However, this method is computational inefficient for large-scale evaluations and it is unable to distinguish between positive and negative contributions to a decision. Furthermore, its application is limited to the self-attention mechanism, whereas modern 3D object detectors usually integrate cross-attention as well.

Gradient Rollout, a method proposed by Chefer et al. [17], considers attention mechanisms across all layers to generate a *relevancy map*, and each layer contributes to this map following a specific set of rules. While its effectiveness has been evaluated for an encoder-decoder transformer for 2D object detection, its applicability to decoder-only architectures mainly used in 3D object detection remains unexplored. Additionally, it requires gradient computation which introduces a computational overhead.

2.3 2D Object Detection

Early object detection relied on two-step approaches, where *region proposals* were first generated and then classified with SVM-based classifiers [22]. RCNNs (Region-based CNNs) made use of CNNs to extract features for region proposal and classification [23]. Fast RCNN [24] and Faster RCNN [25], improved inference speed and accuracy of these region-based methods.

YOLO went on to redefine object detection as a regression problem to predict spatially distinct bounding boxes and class probabilities in a single pass [26]. It divides the input image into a grid structure where each cell predicts multiple bounding boxes and class probabilities.

DetR introduced a different approach to object detection by utilizing the transformer architecture [27]. DetR views object detection as a *set prediction problem* and it learns to directly predict a fixed-size set of bounding boxes and object classes for the entire image in one go, eliminating the need for region proposal, handcrafted anchor boxes and Non-Maximum Suppression (NMS).

2.4 Camera-based 3D Object Detection

Monocular image-based approaches to 3D object detection rely on a single camera as the sensory input. Wang et al. proposed FCOS3D [28], which is based on FCOS [29], a method to transform 3D targets with a center-based paradigm, avoiding any necessary 2D detection or 2D-3D correspondence

priors. In FCOS3D, the 3D targets are projected onto the 2D image plane, resulting in a projected center point.

Pseudo-LiDAR approaches aim to bridge the gap between 3D object detection using LiDAR and monocular or stereo images. Wang et al. proposed a method to transform a camera depth estimation into a point cloud format, called the pseudo-LiDAR [30]. Then existing LiDAR-based 3D object detection pipelines were applied on the 3D pseudo representation. Many other approaches improved this idea and achieved impressive results [31, 32, 33].

Similar to 2D object detection, *transformer-based* architectures have been adopted to 3D object detection, leveraging the ability of transformers to capture long-range dependencies from multiple-camera streams. DETR3D [2] extracts 2D features from multiple camera images and uses 3D object queries to index these features. This is achieved by linking 3D positions to multi-view images using camera transformation matrices. Similar approaches introduce improved query, key and value design. For example, PETR [3] uses a special Position Embedding TRansformation (PETR) for the same multi-camera setup to encode the position information of 3D coordinates into image features, producing the 3D position-aware features. SpatialDETR [5] introduced a geometric positional encoding which incorporated view geometry to explicitly consider the extrinsic and intrinsic camera setup. BEVFormer [4] creates a discretised 3D world in the bird’s eye view perspective and considers each grid as a query location.

2.5 Transformer

The decoder-only transformer model is composed of a stack of layers. Each layer consists of two modules: a *multi-head attention* mechanism and a position-wise *FFN*. The attention mechanism is deployed usually two times: for the *self-attention* and for the *cross-attention* mechanisms. These mechanisms allow the model to assess the relevance of different input tokens while generating the output. The input of the attention mechanism consists of three learned vectors: *query* (Q), *key* (K), and *value* (V).

The transformations for each head $i = 1, 2, \dots, n_h$ of the *multi-head attention* are given by

$$Q_i = Q \cdot W_{Qi}, \quad K_i = K \cdot W_{Ki}, \quad V_i = V \cdot W_{Vi} \quad (1)$$

where W_{Qi} , W_{Ki} , and W_{Vi} are the learned linear transformations for the head. These transformations project the input vectors into the transformer embedded space, allowing each head to focus on a different aspect of the input data. Suppose the input query vector $Q \in \mathbb{R}^{n_q \times m}$, and the transformer model has embedded dimension d . Every attention computation is executed within a query size s , which is given by the formula $s = d/n_h$, where n_h represents the number of attention heads. For each head, the learned linear transformation is denoted as $Q_i \in R^{m \times s}$. Consequently, the query vector $Q_i \in R^{n_q \times s}$, i.e. is projected into the space with dimension s .

Then, the scaled dot-product attention [34] for each head i is given by

$$A(Q_i, K_i, V_i) = \text{Softmax}\left(\frac{Q_i K_i^T}{\sqrt{d}}\right) V_i, \quad (2)$$

which allows the model to assign different levels of importance to different parts of the input sequence.

The final step in the multi-head attention mechanism involves concatenating the outputs of the n_h attention heads and linearly transforming the result to produce the final output, which will lead to the vectors being projected back into the original embedding space of the transformer

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_{n_h}) W_O, \quad (3)$$

where $\text{head}_i = A(Q_i, K_i, V_i)$ and W_O is a learned linear transformation which combines the outputs of all the heads, allowing the model to capture a wide range of information from the input data.

2.6 SpatialDETR

In this study we examine SpatialDETR [5], a transformer-based architecture that takes six multi-view camera images as its input and predicts 3D Oriented Bounding Boxes (OBBs). The images are passed through a shared backbone network, and the resulting representations are then fed into the transformer decoder, as shown in Figure 2. Like other object detectors [3, 5, 2, 4], SpatialDETR

does not employ an *encoder* for the input to prevent the costly self-attention between the image representation. SpatialDETR follows the design of DETR3D, but additionally introduces a *geometric* positional encoding and a *spatially-aware* attention mechanism, which accounts for the extrinsic spatial information of the multi-camera setup and supports a *global attention* across all cameras, which makes this architecture a particularly good choice to generate global explanations w.r.t. the model’s detections.

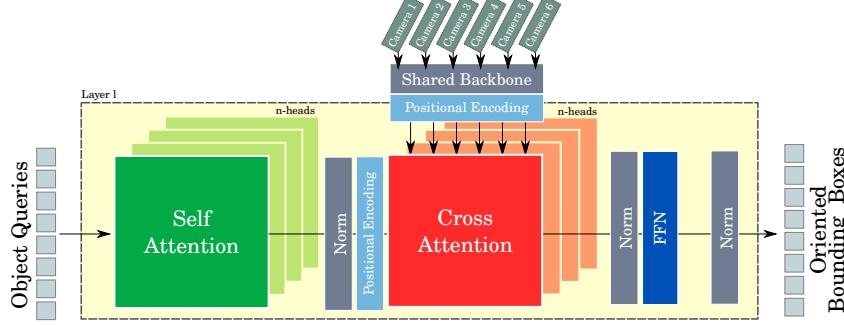


Figure 2: SpatialDETR is a decoder-only transformer architecture. It consists of a query-based self-attention and a cross attention between queries and image features from six cameras [5].

The detections are generated using n_q object queries which are iteratively refined in each layer of the decoder. Within the decoder layer, *self-attention* is applied across the queries and *cross-attention* between object queries and image features. The final bounding boxes are predicted using a FFN, which transforms each latent object query into an OBB.

3 Method

We use the model’s attention layers to produce saliency maps for the interactions in the cross and self-attention. In the following, we discuss different gradient-free and gradient-based computation and propagation rules. The goal of each method is to compute saliency map for each of the input camera images $S \in \mathbb{R}^{n_c \times H \times W}$ in the original camera image dimensions $H \times W$.

3.1 Saliency Map Generation

Let n_q , n_t be number of queries and the number of image tokens per camera respectively. The decoder-only architecture consists of only two types of interactions between the input tokens. The self-attention interaction between the queries A_{SF} and the multi-modal cross-attention between image tokens and queries A_{CR} . In the cross-attention module, the queries are interacting with all tokens of all n_c camera images simultaneously. The interactions happen in each of the n_h heads of the multi-head attention for every layer of the n_l transformer decoder layers. During the inference of the model, we aggregate the attention which results in a multi-dimensional cross-attention $A_{CR} \in \mathbb{R}^{n_c \times n_l \times n_h \times n_q \times n_t}$ and self-attention $A_{SF} \in \mathbb{R}^{n_l \times n_h \times n_q \times n_q}$ tensor. As the model is conditioned in a set-to-set fashion, each query q will result in a classification vector y_q . Each element of this vector represents the sigmoidal class probability for all considered classes. If none of the class probabilities exceeds a certain threshold, the query will be regarded as background and won’t be considered in the computation of the saliency maps.

3.2 Raw-Attention Methods

For the first methods, we will consider only the *raw attention* of the cross-attention. We follow [17] and define a baseline which takes the cross-attention from the last decoder layer

$$S = \mathbb{E}_h(A_{CR}^{n_l})^+ \quad (4)$$

where $A_{CR}^{n_l}$ is the last cross-attention map and \mathbb{E}_h denotes an average across heads. We follow [17] and remove negative contributions before layer averaging. Different fusion approaches across *layers* of the transformer have not been explored by relevant literature [16, 17]. Hence, we additionally

propose a *mean-layer* and a *max-layer* aggregation strategy. The former one computes the mean across all layers and across all heads

$$S = \mathbb{E}_l(\mathbb{E}_h(A_{CR})^+) \quad (5)$$

whereas the latter one filters the maximum values across all layers and heads

$$S = \max_l(\max_h(A_{CR})). \quad (6)$$

3.3 Gradient-Based Methods

For the gradient-based methods we use the *Grad-CAM* [13] adaptation described in [16]. We further adapt the method to the SpatialDETR architecture and compute

$$S = \mathbb{E}_l(\mathbb{E}_h(\nabla A_{CR} \odot A_{CR})^+) \quad (7)$$

for each layer and for each camera, where $\nabla A_{CR} := \frac{\partial y_q}{\partial A_{CR}}$ is the cross-attention gradient w.r.t. model output y_t . In contrast to [16, 17], we not only consider the decoder’s last layer, but instead also aggregate across all layers. We found that this approach captures more information than only considering the last layer.

We adapted *Gradient Rollout* [17], originally developed for encoder-decoder transformers, to our decoder-only detector. We initialize relevancy maps for the self-attention and cross-attention with

$$R_{qq} = \mathbb{I}, \quad R_{qi} = \emptyset \quad (8)$$

and then perform a layer-wise iterative update for the self-attention with

$$\bar{A}_{SF} = \mathbb{E}_h(\nabla A_{SF} \odot A_{SF})^+ \quad (9)$$

where $A_{SF} \in \mathbb{R}^{n_q \times n_q}$ and \mathbb{E}_h computes the mean across all heads. Then, we update the relevancy maps for the self-attention

$$R_{qq} = R_{qq} + \bar{A}_{SF} \cdot R_{qq} \quad (10)$$

$$R_{qi} = R_{qi} + \bar{A}_{SF} \cdot R_{qi}. \quad (11)$$

Next, we aggregate the cross-attentions and update R_{qi}

$$\bar{A}_{CR} = \mathbb{E}_h(\nabla A_{CR} \odot A_{CR})^+ \quad (12)$$

$$R_{qi} = R_{qi} + (\bar{R}_{qq})^T \cdot \bar{A}_{CR} \quad (13)$$

where \bar{R}_{qq} is the row-wise normalized matrix of R_{qq} , as described in [17]. After passing the last layer of the decoder, the saliency map is obtained by the aggregated cross-attention relevancy map, i.e. $S = R_{qi}$. More details on the implementation of the algorithms can be found in Appendix A. Note that all methods require only a few simple hooks in the attention modules. The gradient-based methods need substantially more memory and runtime compared to the raw-attention based methods.

4 Experiments

We perform extensive experiments on the nuScenes dataset [35]. First, we perform a qualitative visual exploration of the attention maps to better understand the mechanism of the architecture. Second, we quantitatively evaluate the quality of the saliency maps produced by each method using *positive* and *negative perturbation* tests. Third, we follow [36] and perform a simple sanity check for saliency maps.

4.1 Visual Exploration

We developed a visual exploration tool which let us browse through the nuScenes dataset and investigated the attention mechanism in the model for each head, each layer, different configurations, classes, scenes and aggregation methods. Figure 3 displays raw cross-attention for pedestrians across all six decoder layers. The model pays attention to specific parts of the human body, with a preference for lower torso and legs. Hereby, every layer seems to concentrate on a different part of the body. We could observe this behavior across several different classes, such as car, truck and construction vehicle. Another example is shown in Figure 4 where each layer seems to cover different parts of

the construction vehicle. Additional examples are provided in Appendix D. Furthermore, examples of objects located within the overlapping FOV of two cameras are presented in Appendix E. This observation let us to the conclusion, that aggregating the attentions across all decoder layers is immensely important to give a holistic explanation for a certain detection, which resulted in the *mean-layer* and *max-layer* aggregation methods (*cf.* Eq. 5 and Eq. 6), which have not been explored previously [17]. It seems that the model alternates its attention between specific concepts of a class in every layer to distinguish between other classes, but also find the center, orientation and the dimensions of the OBB. In Figure 5, we compare the different explainability methods on different classes. It becomes clear that if only one layer of the decoder is used, the quality of the saliency map is inferior in many cases as only one "concept" of a class is covered by that layer. The *mean-layer* aggregation gives us a very smooth saliency map for an object, whereas the *max-layer* fusion and *Grad-CAM* give a diversified focus through various parts of the object. Considering *Gradient Rollout*, the resulting saliency map only highlights the most essential areas for detection.

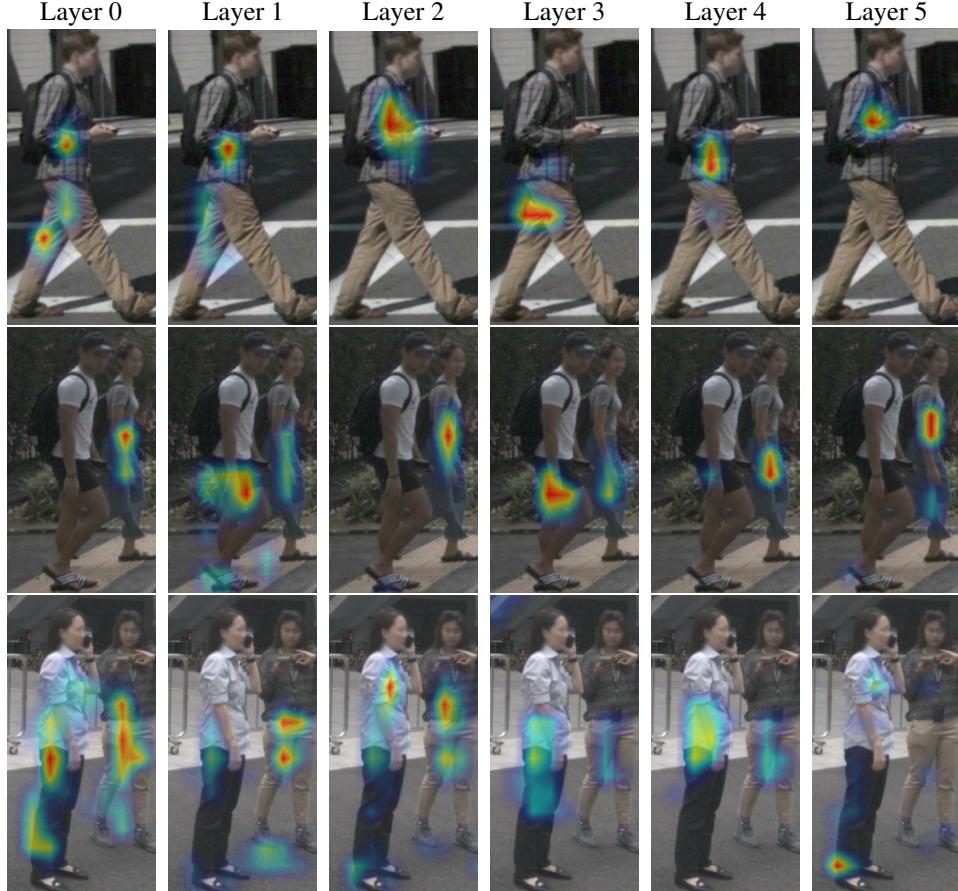


Figure 3: Raw cross-attention $\mathbb{E}_h(A_{CR})^+$ for a single query shown for all six decoder layers for different pedestrian. Each layer shifts it's focus on a different conceptual part of the pedestrian.

4.2 Perturbation Test

We use a pre-trained model, iterate over the validation split of the nuScenes dataset and execute the model with the current sample data. With the information of this forward pass, we compute the saliency map for each camera image and we remove a certain percentage of the input: For the *positive perturbation* test, the regions with the highest activation, and for the *negative perturbation*, the regions with the lowest activation are masked, as show in Appendix B. The masked regions are filled with the mean of the image. Then, the perturbed camera images are used again as input to the model and the model's output is collected in order to compute the nuScenes Detection Score

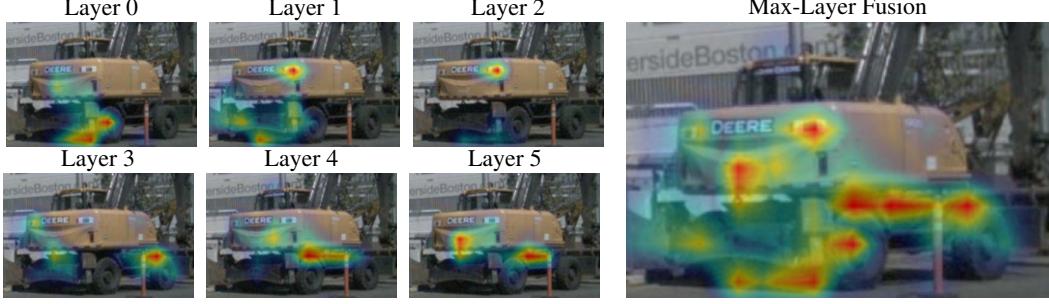


Figure 4: Raw cross-attention $\mathbb{E}_h(A_{CR})^+$ for all six layers of the decoder. The image on the right depicts the *max-layer* aggregation, where the maximum across all layers is aggregated.

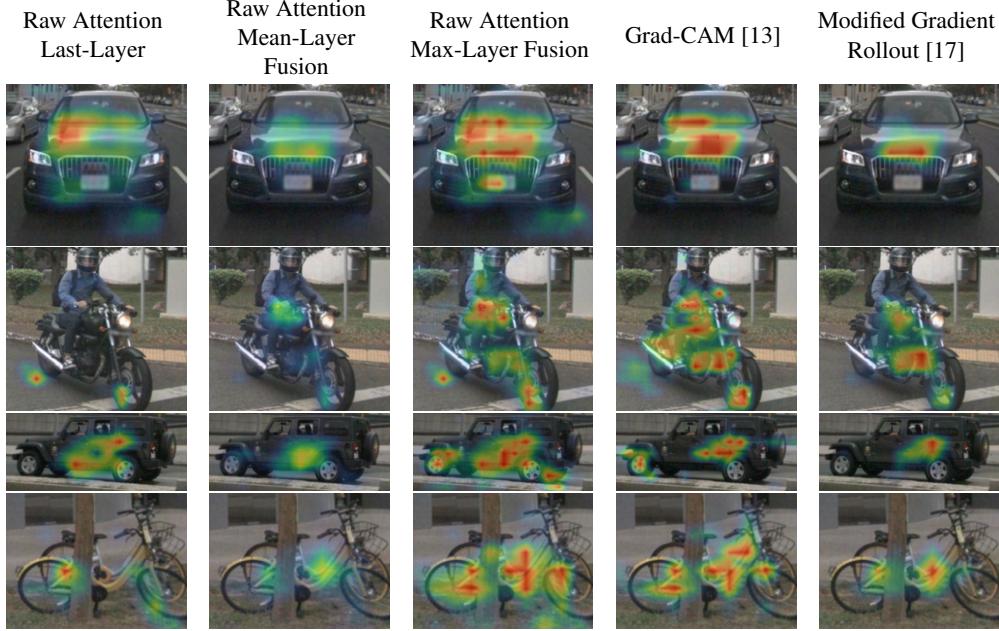


Figure 5: Saliency maps generated for different objects classes using all explainability methods.

(NDS) [35]. During the test, we gradually increase the percentage of masked input pixels. For the *positive perturbation*, we expect that the NDS decreases quickly as the most important regions are removed first. For the *negative perturbation*, a good saliency generation method would result in a curve with a slow decrease as only irrelevant regions are removed at the beginning of the test. For both tests, we compute the area-under-the-curve (AUC), to measure the impact on the model’s performance. To establish a simple baseline for this test, we generate a *random explanation* drawn from a normal distribution $S \sim \mathcal{N}(\mu, \sigma^2)$ for each sample. As shown in Figure 8 and 9, all five methods perform reasonably well and outperform a *random explanation*. The methods *max-layer* fusion and *Grad-CAM* outperform the other methods, followed by *mean-layer* fusion. As expected, taking only the *last-layer* of the decoder’s attentions results in an inferior performance compared to methods that aggregate the attention over all layer. *Gradient Rollout* [17] was also less effective than the other methods. The formulation may not work well with the decoder-only transformer architecture, which was not originally designed for it.

4.3 Saliency Map Sanity Check

We follow [36] and perform a model parameter randomization test. We randomly initialize an untrained model and generate saliency maps with *max-layer* fusion. The test determines if the saliency maps depend on the learned parameters of the model. Some model and saliency generation methods are insensitive to the properties of the trained parameters, if the saliency maps are insensitive

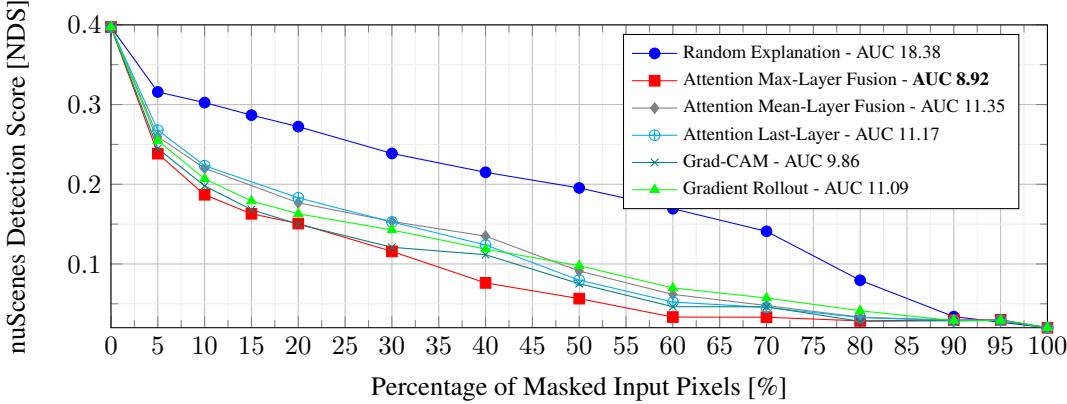


Figure 6: Positive perturbation test evaluated on the nuScenes validation set. Smaller AUC is better.

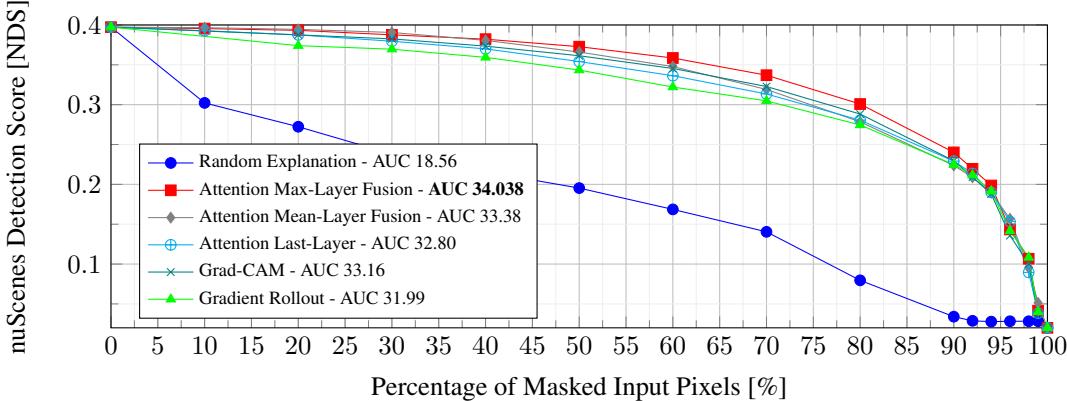


Figure 7: Negative perturbation test evaluated on the nuScenes validation set. Larger AUC is better.

to the learned parameters, then the saliency maps of the trained and randomly initialized model will be similar [36]. We could observe that the randomly initialized model does not produce reasonable saliency maps. Examples are shown in Appendix C. Hence, our approach passes the sanity check.

5 Conclusion

We introduced an approach for generating saliency maps using a transformer-based multi-camera model in the context of 3D object detection. Our goal was to provide insights into which regions of input images are most influential in determining object detection results. Our approach, based on raw cross-attention, proved to be more efficient than gradient-based methods. Our perturbation test demonstrated the effectiveness of our approach. Visual exploration of attention maps showed that the transformer’s attention alternates between different concepts of a class in different layers to distinguish between classes and to determine object properties. Aggregating attention across different layers was crucial for a comprehensive explanation.

The current method is restricted to generating saliency maps solely for queries exceeding the detection threshold. Queries that produce detections with low classification scores are not considered. Additionally, the resolution of the generated saliency maps is relatively coarse, enabling the highlighting of object concepts but not fine details. In the future, we want to extend the approach to a multi-modal transformer model which utilizes LiDAR and multi-view camera images as input. We aim to enhance the transparency of such end-to-end model by tracing the origins of detection, attributing them to the specific sensors that contributed to their identification.

Acknowledgments and Disclosure of Funding

This work has received funding from the European Union’s Horizon Europe Research and Innovation Programme under Grant Agreement No 101076754 - AIthena project.

References

- [1] Zixiang Zhou, Xiangchen Zhao, Yu Wang, Panqu Wang, and Hassan Foroosh. Centerformer: Center-based transformer for 3d object detection, 2022.
- [2] Yue Wang, Vitor Campagnolo Guizilini, Tianyuan Zhang, Yilun Wang, Hang Zhao, and Justin Solomon. Detr3d: 3d object detection from multi-view images via 3d-to-2d queries. In *Conference on Robot Learning*, pages 180–191. PMLR, 2022.
- [3] Yingfei Liu, Tiancai Wang, Xiangyu Zhang, and Jian Sun. Petr: Position embedding transformation for multi-view 3d object detection. *arXiv preprint arXiv:2203.05625*, 2022.
- [4] Zhiqi Li, Wenhui Wang, Hongyang Li, Enze Xie, Chonghao Sima, Tong Lu, Yu Qiao, and Jifeng Dai. Bevformer: Learning bird’s-eye-view representation from multi-camera images via spatiotemporal transformers. *arXiv preprint arXiv:2203.17270*, 2022.
- [5] Simon Doll, Richard Schulz, Lukas Schneider, Viviane Benzin, Markus Enzweiler, and Hendrik PA Lensch. Spatialdetr: Robust scalable transformer-based 3d object detection from multi-view camera images with global cross-sensor attention. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIX*, pages 230–245. Springer, 2022.
- [6] Junjie Yan, Yingfei Liu, Jianjian Sun, Fan Jia, Shuaolin Li, Tiancai Wang, and Xiangyu Zhang. Cross modal transformer: Towards fast and robust 3d object detection, 2023.
- [7] Jiqian Dong, Sikai Chen, Shuya Zong, Tiantian Chen, and Samuel Labi. Image transformer for explainable autonomous driving system. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 2732–2737, 2021.
- [8] M. T. Ribeiro, S. Singh, and C. Guestrin. Why should i trust you? explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144, August 2016.
- [9] Ruth C Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *Proceedings of the IEEE international conference on computer vision*, pages 3429–3437, 2017.
- [10] Ruth Fong, Mandela Patrick, and Andrea Vedaldi. Understanding deep networks via extremal perturbations and smooth masks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2950–2958, 2019.
- [11] Vitali Petsiuk, Abir Das, and Kate Saenko. Rise: Randomized input sampling for explanation of black-box models. *arXiv preprint arXiv:1806.07421*, 2018.
- [12] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.
- [13] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 618–626, 2017.
- [14] S. Bach, A. Binder, G. Montavon, F. Klauschen, K. R. Müller, and W. Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10(7):e0130140, 2015.
- [15] Samira Abnar and Willem Zuidema. Quantifying attention flow in transformers. *arXiv preprint arXiv:2005.00928*, 2020.

- [16] Hila Chefer, Shir Gur, and Lior Wolf. Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 782–791, 2021.
- [17] Hila Chefer, Shir Gur, and Lior Wolf. Generic attention-model explainability for interpreting bi-modal and encoder-decoder transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 397–406, 2021.
- [18] Baptiste Abeloos and Stéphane Herbin. Explaining object detectors: the case of transformer architectures. *Workshop on Trustworthy Artificial Intelligence as a part of the ECML/PKDD 22 program*, 2022.
- [19] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, et al. Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information fusion*, 58:82–115, 2020.
- [20] S. M. Lundberg and S. I. Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774, 2017.
- [21] Amin Ghiasi, Hamid Kazemi, Eitan Borgnia, Steven Reich, Manli Shu, Micah Goldblum, Andrew Gordon Wilson, and Tom Goldstein. What do vision transformers learn? a visual exploration. *arXiv preprint arXiv:2212.06727*, 2022.
- [22] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, volume 1, pages 886–893. Ieee, 2005.
- [23] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.
- [24] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.
- [25] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.
- [26] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.
- [27] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I*, pages 213–229. Springer, 2020.
- [28] Tai Wang, Xinge Zhu, Jiangmiao Pang, and Dahua Lin. Fcos3d: Fully convolutional one-stage monocular 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 913–922, 2021.
- [29] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9627–9636, 2019.
- [30] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8445–8453, 2019.
- [31] Xinshuo Weng and Kris Kitani. Monocular 3d object detection with pseudo-lidar point cloud. *CoRR*, abs/1903.09847, 2019.

- [32] Rui Qian, Divyansh Garg, Yan Wang, Yurong You, Serge J. Belongie, Bharath Hariharan, Mark Campbell, Kilian Q. Weinberger, and Wei-Lun Chao. End-to-end pseudo-lidar for image-based 3d object detection. *CoRR*, abs/2004.03080, 2020.
- [33] Xinzhu Ma, Zhihui Wang, Haojie Li, Wanli Ouyang, and Pengbo Zhang. Accurate monocular 3d object detection via color-embedded 3d reconstruction for autonomous driving. *CoRR*, abs/1903.11444, 2019.
- [34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30:5998–6008, 2017.
- [35] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11621–11631, 2020.
- [36] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. *Advances in neural information processing systems*, 31, 2018.

A Implementation Details

Algorithm 1: Raw Attention Max-Layer

Input : $A_{CR} \in \mathbb{R}^{nc \times nl \times nh \times nq \times nt}$
Output: $S \in \mathbb{R}^{nc \times H \times W}$

```

1  $S \leftarrow \emptyset$ 
2 for  $c = 1$  to  $nc$  do
3   # select camera  $c$ 
4    $S^c \leftarrow \emptyset$ 
5    $A_{CR}^c \in \mathbb{R}^{nl \times nh \times nq \times nt} \leftarrow A_{CR}$ 
6    $A_{CR}^{ch} \in \mathbb{R}^{nl \times nq \times nt} \leftarrow \max_{h \in \{1, \dots, nh\}} A_{CR}^c$ 
7    $A_{CR}^{clh} \in \mathbb{R}^{nq \times nt} \leftarrow \max_{l \in \{1, \dots, nl\}} A_{CR}^{ch}$ 
8   for  $q = 1$  to  $nq$  do
9     if  $y_q > threshold$  then
10    |    $S^q \in \mathbb{R}^{H' \times W'} \leftarrow A_{CR}^{lcq} \in \mathbb{R}^{nt}$ 
11    |    $S^q \in \mathbb{R}^{H \times W} \leftarrow S^q \in \mathbb{R}^{H' \times W'}$ 
12    |    $S^c \in \mathbb{R}^{H \times W} \leftarrow S^c + S^q \in \mathbb{R}^{H \times W}$ 
13    |   end
14   end
15    $S \leftarrow S^c$ 
16 end

```

Algorithm 2: Grad-CAM

Input : $A_{CR} \in \mathbb{R}^{nc \times nl \times nh \times nq \times nt}$
Output: $S \in \mathbb{R}^{nc \times H \times W}$

```

1  $S \leftarrow \emptyset$ 
2 # compute gradient w.t.r. object scores
3  $\nabla A_{CR} \in \mathbb{R}^{nc \times nl \times nh \times nq \times nt} \leftarrow \frac{\partial y}{\partial A_{CR}}$ 
4 for  $c = 1$  to  $nc$  do
5   # select camera  $c$ 
6    $S^c \leftarrow \emptyset$ 
7    $A_{CR}^c \in \mathbb{R}^{nl \times nh \times nq \times nt} \leftarrow A_{CR}$ 
8    $\nabla A_{CR}^c \in \mathbb{R}^{nl \times nh \times nq \times nt} \leftarrow \nabla A_{CR}$ 
9    $A_{CR}^{ch} \in \mathbb{R}^{nq \times nt} \leftarrow \mathbb{E}_h ((\nabla A_{CR}^c \cdot A_{CR}^c)^+)$ 
10   $A_{CR}^{clh} \in \mathbb{R}^{nq \times nt} \leftarrow \mathbb{E}_l (A_{CR}^{ch})$ 
11  for  $q = 1$  to  $nq$  do
12    if  $y_q > threshold$  then
13       $S^q \in \mathbb{R}^{H' \times W'} \leftarrow A_{CR}^{clhq} \in \mathbb{R}^{nt}$ 
14       $S^q \in \mathbb{R}^{H \times W} \leftarrow S^q \in \mathbb{R}^{H' \times W'}$ 
15       $S^c \in \mathbb{R}^{H \times W} \leftarrow S^c + S^q \in \mathbb{R}^{H \times W}$ 
16    end
17  end
18   $S \leftarrow S^c$ 
19 end

```

Algorithm 3: Gradient Rollout

Input : $A_{CR} \in \mathbb{R}^{nc \times nl \times nh \times nq \times nt}$
Input : $A_{SF} \in \mathbb{R}^{nl \times nh \times nq \times nq}$
Output: $S \in \mathbb{R}^{nc \times H \times W}$

```

1  $S \leftarrow \emptyset$ 
2 # compute gradient w.t.r object scores
3  $\nabla A_{CR} \in \mathbb{R}^{nc \times nl \times nh \times nq \times nt} \leftarrow \frac{\partial y}{\partial A_{CR}}$ 
4  $\nabla A_{SF} \in \mathbb{R}^{nl \times nh \times nq \times nt} \leftarrow \frac{\partial y}{\partial A_{SF}}$ 
5 for  $c = 1$  to  $nc$  do
6   # select camera  $c$ 
7    $S^c \leftarrow \emptyset$ 
8    $A_{CR}^c \in \mathbb{R}^{nl \times nh \times nq \times nt} \leftarrow A_{CR}$ 
9    $\nabla A_{CR}^c \in \mathbb{R}^{nl \times nh \times nq \times nt} \leftarrow \nabla A_{CR}$ 
10   $R_{qq} \in \mathbb{R}^{nq \times nq} \leftarrow \mathbb{I}$ 
11   $R_{qi} \in \mathbb{R}^{nq \times nt} \leftarrow \emptyset$ 
12  for  $l = 1$  to  $nl$  do
13     $A_{SF}^l \in \mathbb{R}^{nh \times nq \times nq} \leftarrow A_{SF}$ 
14     $\nabla A_{SF}^l \in \mathbb{R}^{nh \times nq \times nq} \leftarrow \nabla A_{SF}$ 
15     $\bar{A}_{SF}^{hl} \leftarrow \mathbb{E}_h ((\nabla A_{SF} \odot A_{SF}^l)^+)$ 
16    # update relevancy maps
17     $R_{qq} \leftarrow R_{qq} + \bar{A}_{SF}^l \cdot R_{qq}$ 
18     $R_{qi} \leftarrow R_{qi} + \bar{A}_{SF}^l \cdot R_{qi}$ 
19     $A_{CR}^{lc} \in \mathbb{R}^{nh \times nq \times nt} \leftarrow A_{CR}^c$ 
20     $\nabla A_{CR}^{lc} \in \mathbb{R}^{nh \times nq \times nt} \leftarrow \nabla A_{CR}^c$ 
21     $\bar{A}_{CR}^{lc} \in \mathbb{R}^{nq \times nt} \leftarrow \mathbb{E}_h ((\nabla A_{CR}^{lc} \odot A_{CR}^{lc})^+)$ 
22     $R_{qi} \in \mathbb{R}^{nq \times nt} \leftarrow R_{qi} + (\bar{R}_{qq})^T \cdot \bar{A}_{CR}^{lc}$ 
23    for  $q = 1$  to  $nq$  do
24      if  $y_q > threshold$  then
25         $S^q \in \mathbb{R}^{H' \times W'} \leftarrow R_{qi}^q \in \mathbb{R}^{nt}$ 
26         $S^q \in \mathbb{R}^{H \times W} \leftarrow S^q \in \mathbb{R}^{H' \times W'}$ 
27         $S^c \in \mathbb{R}^{H \times W} \leftarrow S^c + S^q \in \mathbb{R}^{H \times W}$ 
28      end
29    end
30  end
31   $S \leftarrow S^c$ 
32 end

```

B Perturbation Test

A perturbation test is a popular approach to evaluate the effectiveness of explainability methods [16, 17]. We designed the following perturbation test for our 3D object detection setup: We use the whole nuScenes validation split and compute the saliency maps for each camera in each sample. Then we mask a certain percentage of the image input pixels based on the saliency maps and use this perturbed images to compute the nuScenes detection score (NDS). For the *positive* perturbation test we gradually mask the areas in the input camera images with the highest saliency activation, as shown in Fig. 8. For the *negative* perturbation test the area with the lowest activation is masked first, visualized by Fig. 9.

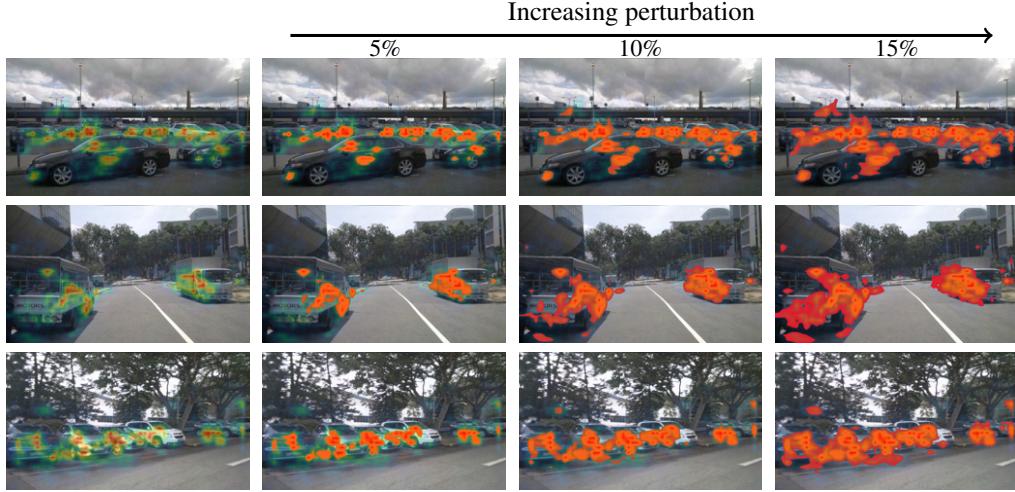


Figure 8: Visualization of the *positive perturbation* test on three sample images from the nuScenes dataset while increasing the number of masked pixels. Masked pixels, here shown in red, are filled with the mean of the image.

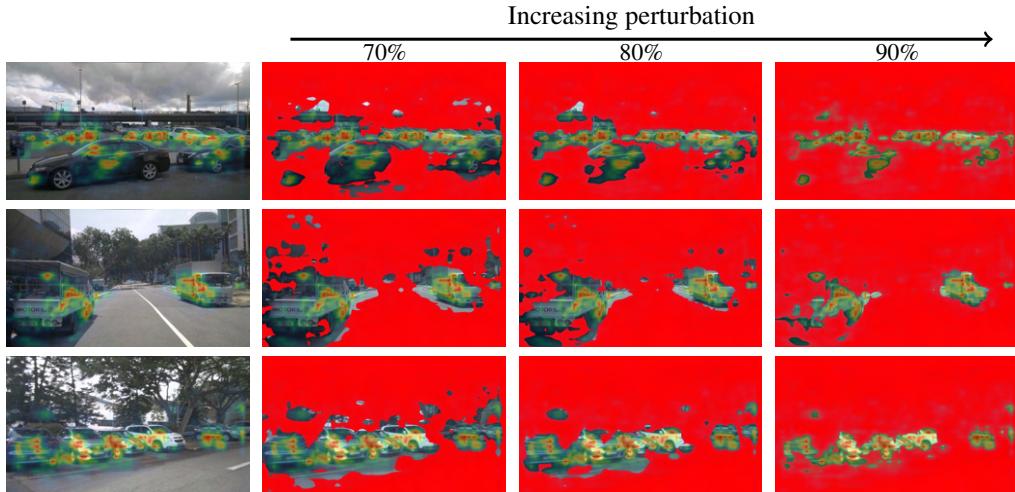


Figure 9: Visualization of the *negative perturbation* test on three sample images from the nuScenes dataset while increasing the number of masked pixels. Masked pixels, here shown in red, are filled with the mean of the image.

C Sanity Checks for Saliency Maps

We follow [36] and conduct a *model parameter randomization test* in which we generate saliency maps of a randomly initialized untrained network, as shown in Fig. 10. The generated saliency maps differ substantially from the saliency maps of the trained model, which indicates that the sanity check has been passed. We perform another test, by initializing the ResNet backbone from a checkpoint while keeping the transformer weights uninitialized. This test should show that the saliency maps do not depend solely on the CNN-based backbone, but rather from the interplay between backbone and transformer. As shown in Fig. 11, the obtained saliency maps are very noisy, but we can also observe some highlights on relevant objects, leading to the conclusion that the saliency maps in our approach are in fact sensitive to the properties of the transformer decoder.



Figure 10: Raw saliency map when transformer decoder and backbone weights are randomly initialized. The resulting saliency maps are meaningless and not comparable to the saliency maps of the trained model. This indicates that the sanity check for saliency maps has been passed.

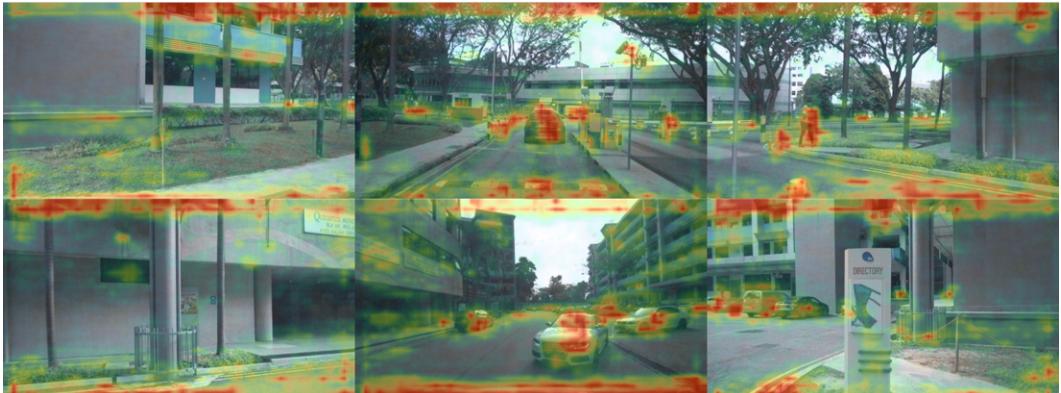


Figure 11: Raw saliency map output when the ResNet backbone weights are initialized from an object detection checkpoint, but the weights of the transformer decoder are randomly initialized. This setting seems to highlight some areas of relevant objects, but also a lot of noise is generated. This underlines that the saliency maps generated with our approach are in fact sensitive to the properties of the transformer model weights.

D Examples

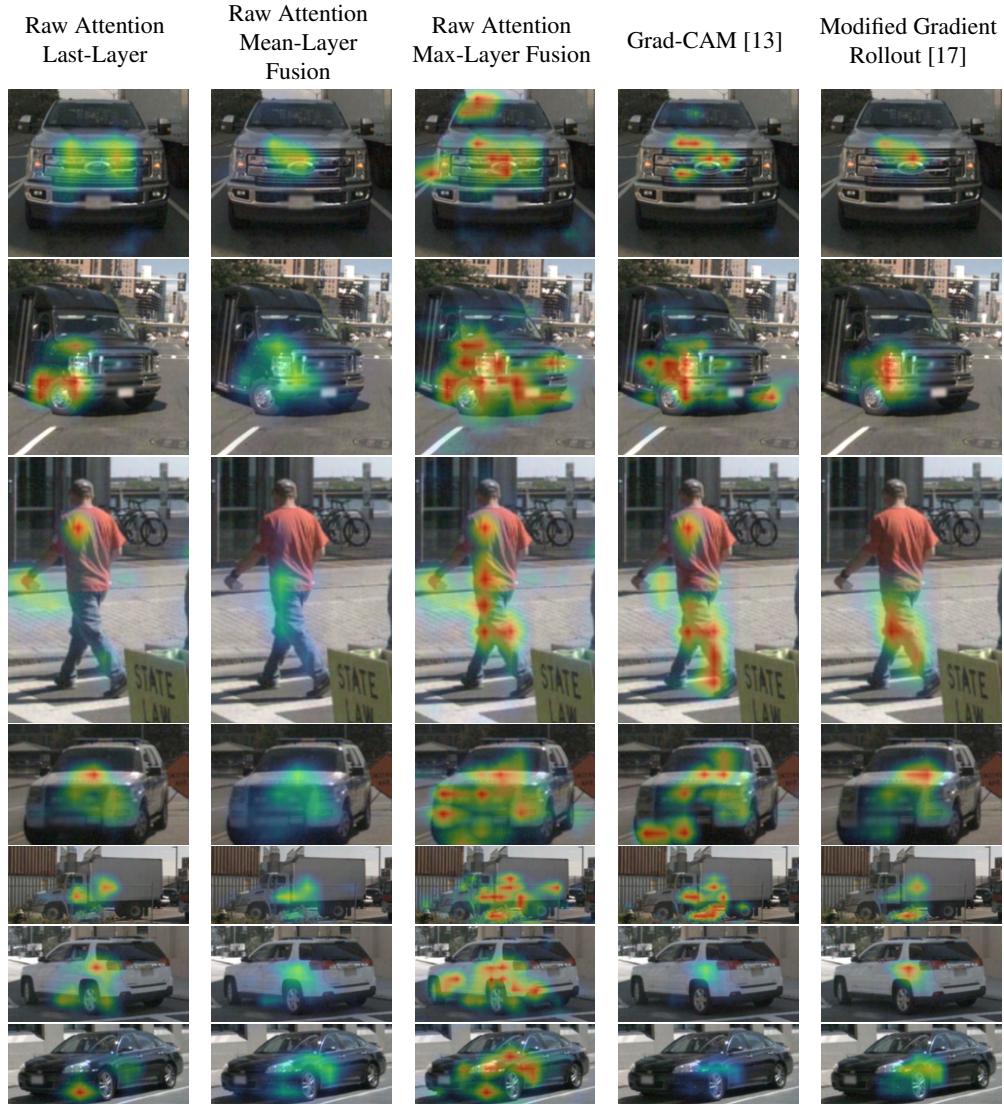


Figure 12: Saliency maps generated for Raw Attention (Last, Mean, Max), Grad-CAM and Gradient Rollout for different objects.

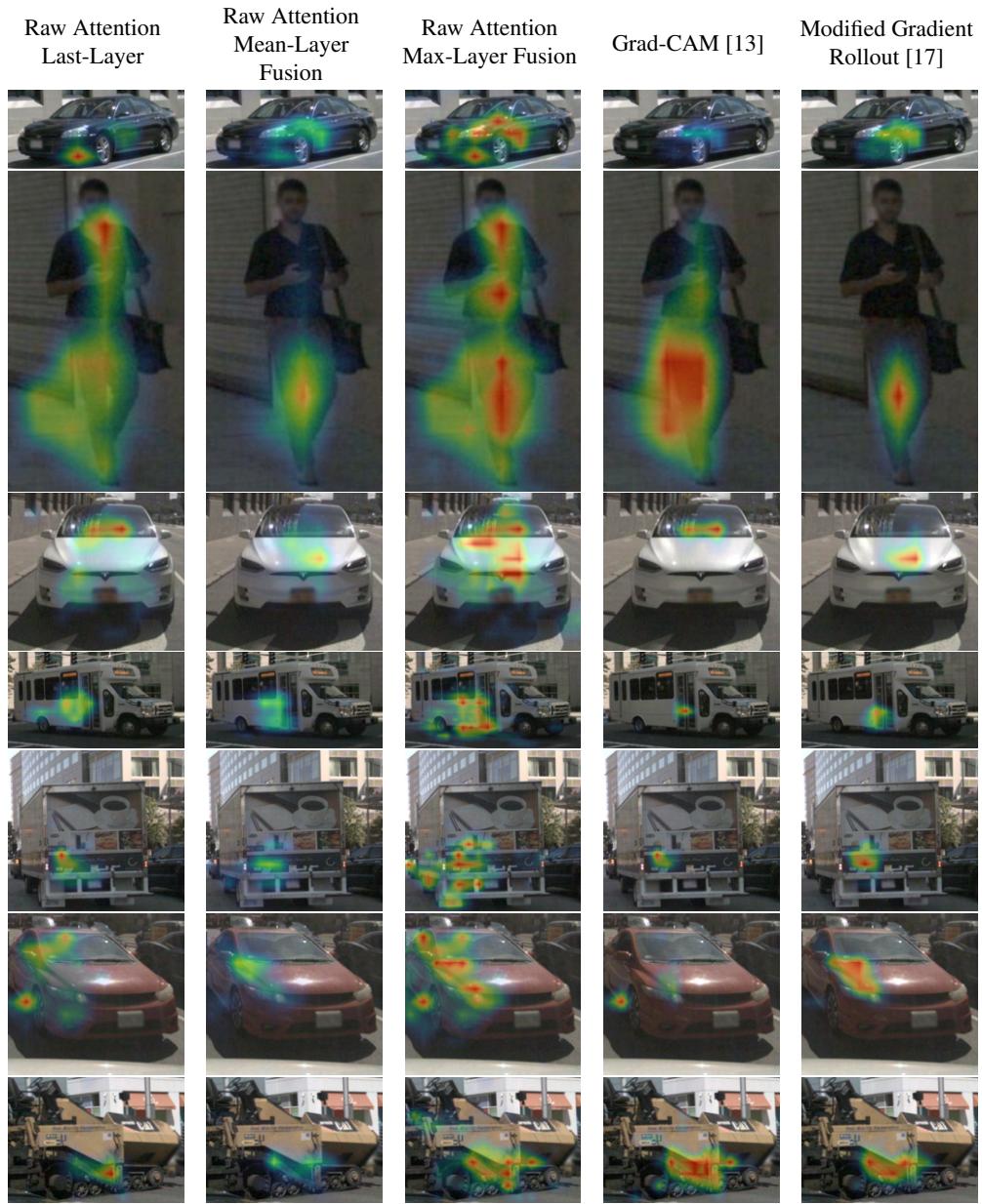


Figure 13: Saliency maps generated for Raw Attention (Last, Mean, Max), Grad-CAM and Gradient Rollout for different objects.

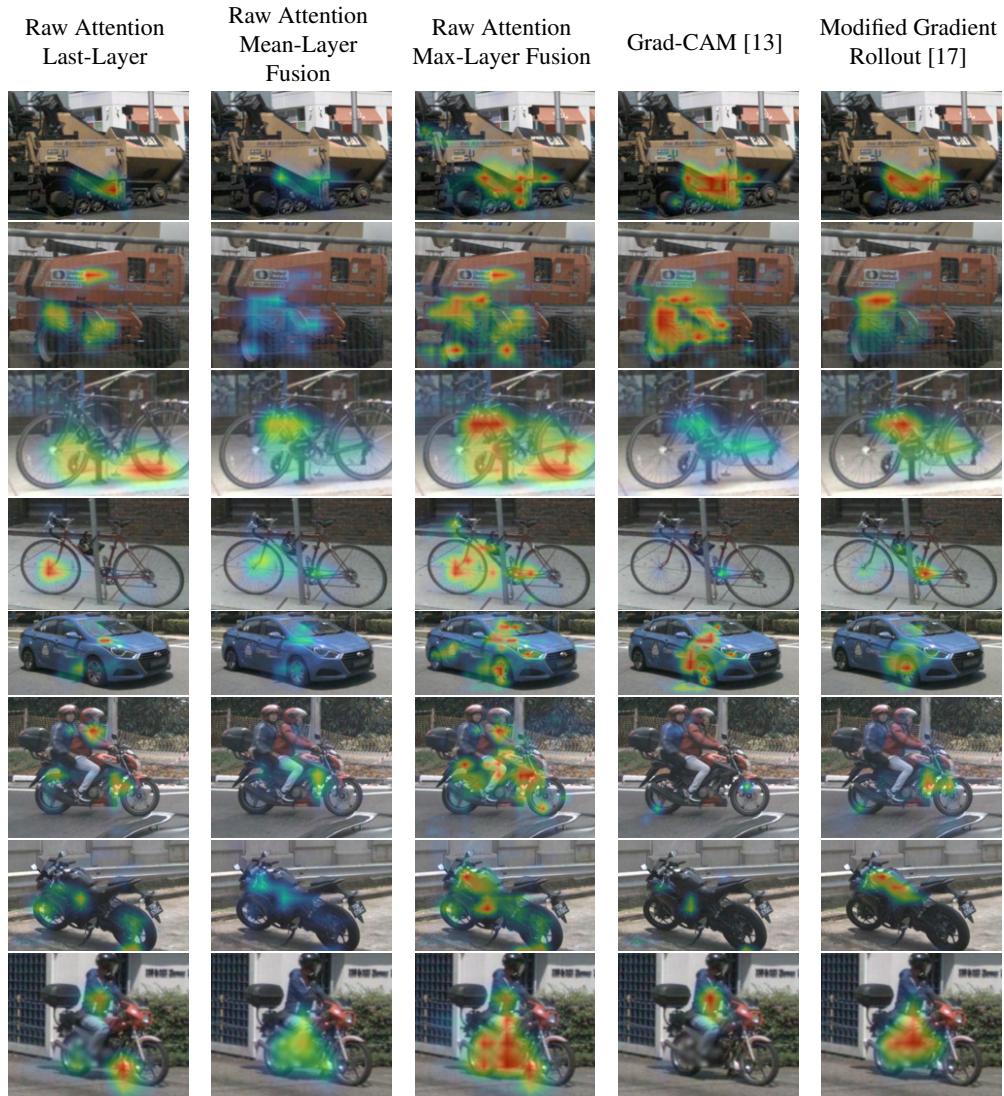


Figure 14: Saliency maps generated for Raw Attention (Last, Mean, Max), Grad-CAM and Gradient Rollout for different objects.

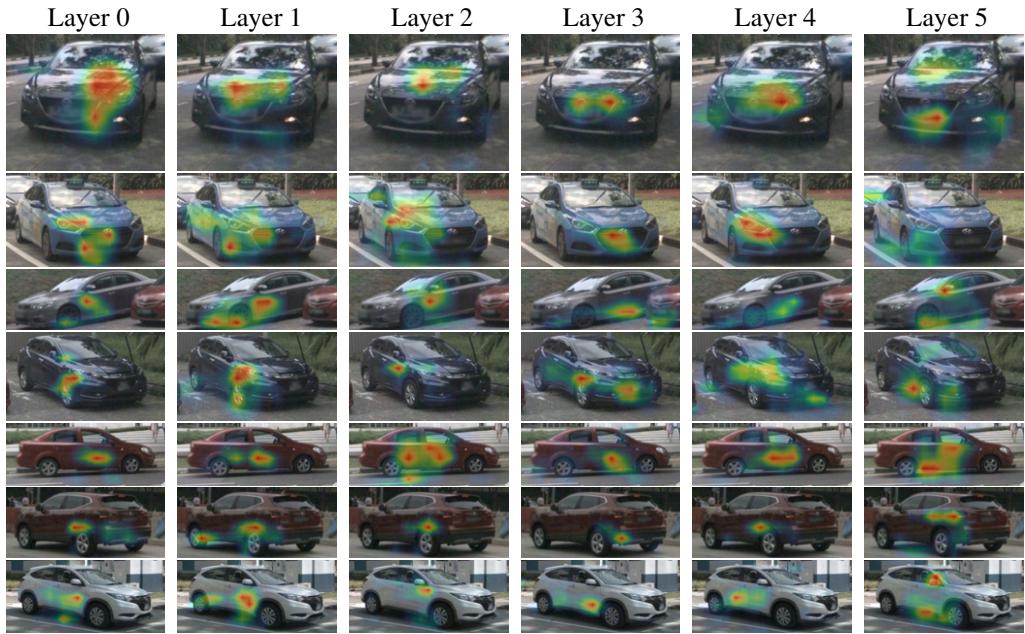


Figure 15: Raw cross-attention $\mathbb{E}_h(A_{CR})^+$ examples for class car.

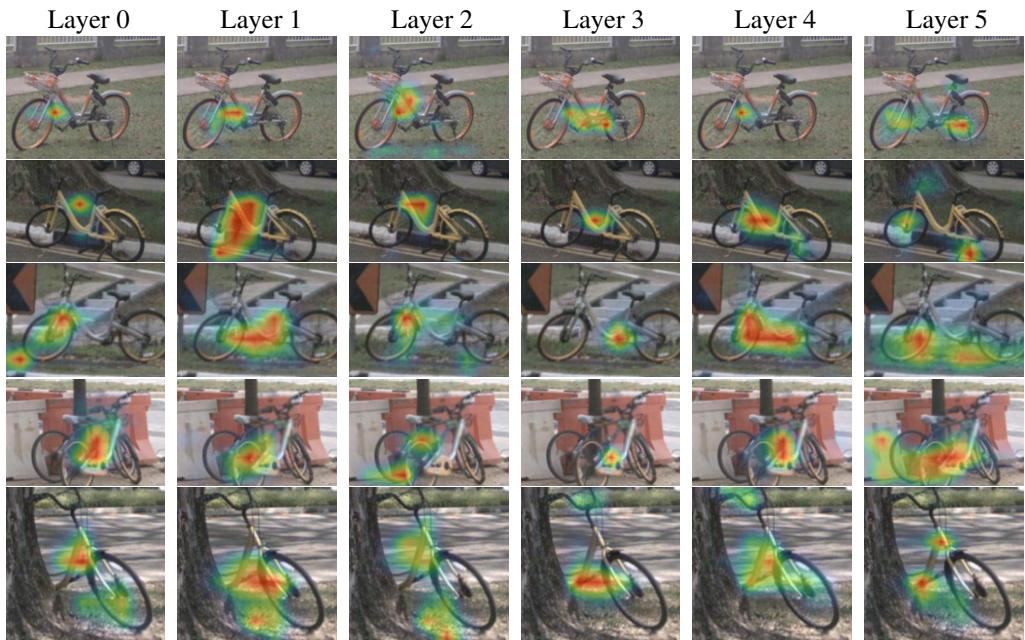
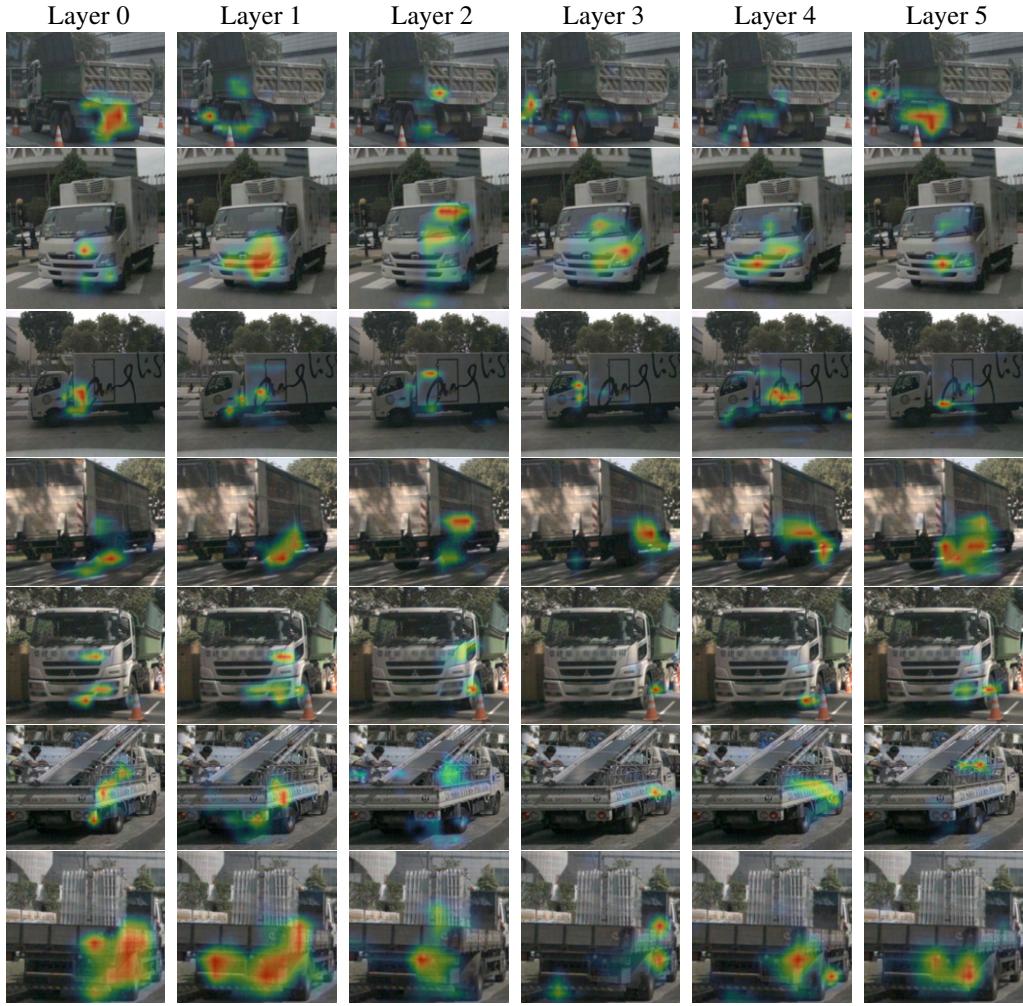


Figure 16: Raw cross-attention $\mathbb{E}_h(A_{CR})^+$ examples for class bicycles.



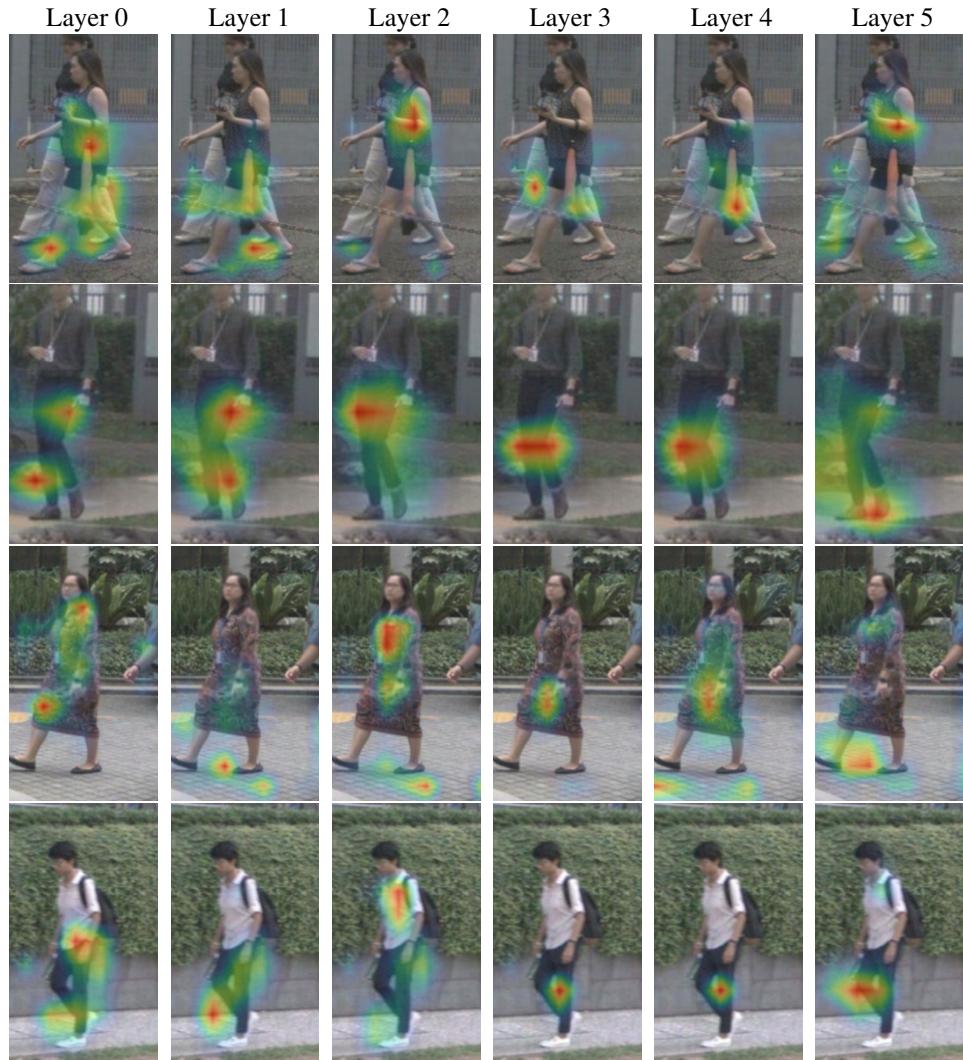


Figure 19: Raw cross-attention $\mathbb{E}_h(A_{CR})^+$ examples for class **pedestrians**.

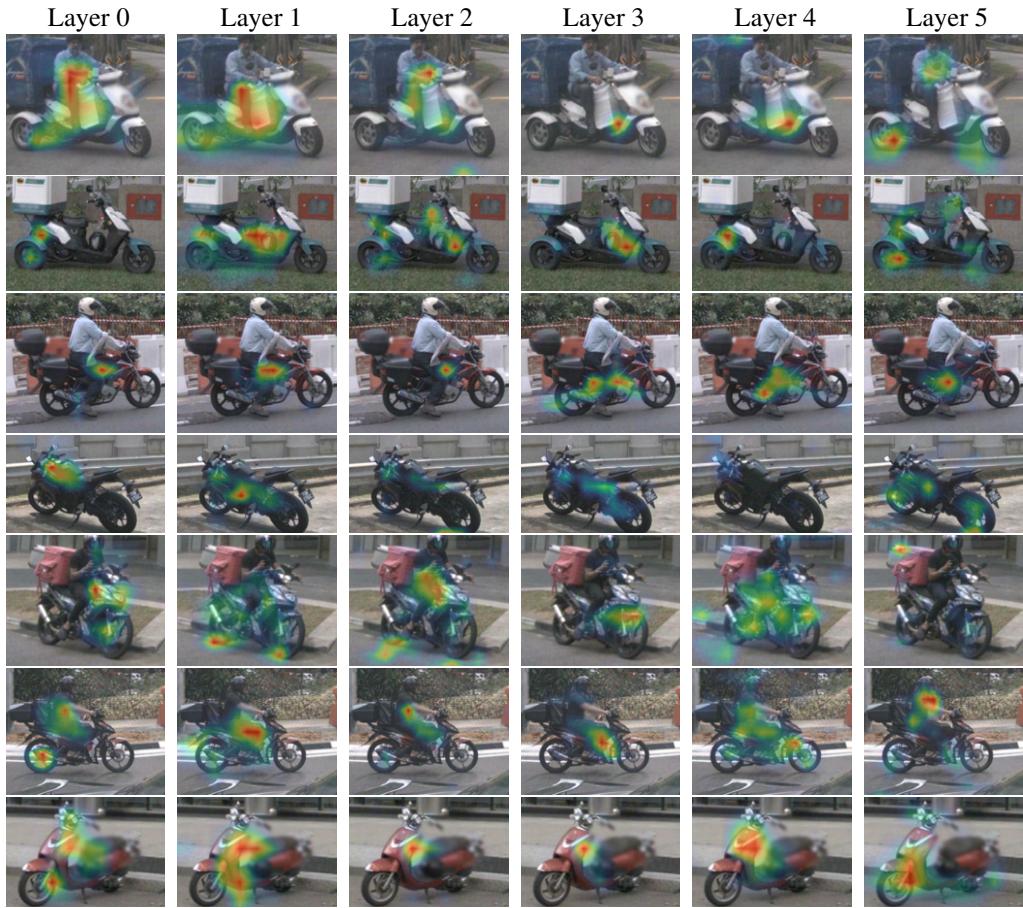


Figure 20: Raw cross-attention $\mathbb{E}_h(A_{CR})^+$ examples for class motorcycles.

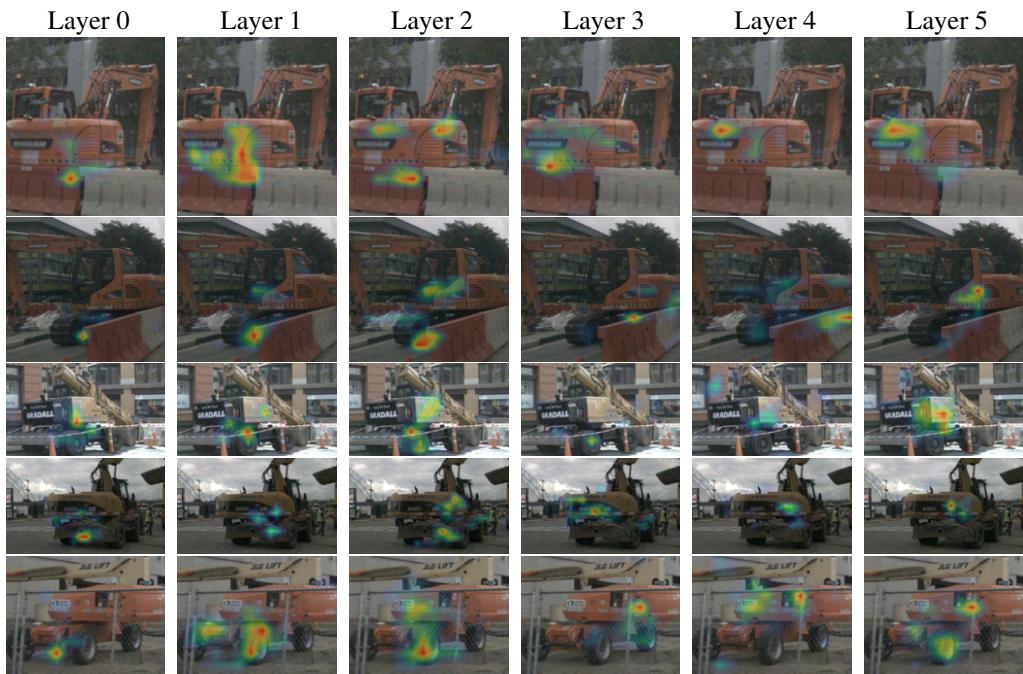


Figure 21: Raw cross-attention $\mathbb{E}_h(A_{CR})^+$ examples for class construction vehicles.

E Examples with Camera FOV Overlap

The nuScenes dataset [35] was recorded with a multi-camera setup consisting of six cameras which have a partial overlap in their FOVs. The SpatialDETR transformer architecture projects each query onto all camera frames, enabling a *global attention* on all input images. This end-to-end approach generates a single detection for objects that appear in two or more cameras at the same time. In such cases, the attention is distributed on the overlapping parts of the object, as shown in Fig. 22.



Figure 22: Raw cross-attention examples for objects (green OBB) that lie in the overlapping FOV of two cameras. In each example, a single query is used to generate the saliency maps for all camera images. Attention can be observed on the object on both overlapping camera images. We further approximated the contribution of each camera for a detection by computing the fraction of attention for each camera, indicated by the green bar and the percentage data.