OCTET: Object-aware Counterfactual Explanations

Mehdi Zemni[1], Mickaël Chen[1], Éloi Zablocki[1], Hédi Ben-Younes[1], Patrick Pérez[1], Matthieu Cord[1,2]
[1] Valeo.ai, Paris, France [2] Sorbonne Université, Paris, France

Abstract

Nowadays, deep vision models are being widely deployed in safety-critical applications, e.g., autonomous driving, and explainability of such models is becoming a pressing concern. Among explanation methods, counterfactual explanations aim to find minimal *and* interpretable *changes to the input image that would also change the output of the model to be explained. Such explanations point end-users at the main factors that impact the decision of the model. However, previous methods struggle to explain decision models trained on images with many objects, e.g., urban scenes, which are more difficult to work with but also arguably more critical to explain. In this work, we propose to tackle this issue with an object-centric framework for counterfactual explanation generation. Our method, inspired by recent generative modeling works, encodes the query image into a latent space that is structured in a way to ease object-level manipulations. Doing so, it provides the end-user with control over which search directions (e.g., spatial displacement of objects, style modification, etc.) are to be explored during the counterfactual generation. We conduct a set of experiments on counterfactual explanation benchmarks for driving scenes, and we show that our method can be adapted beyond classification, e.g., to explain semantic segmentation models. To complete our analysis, we design and run a user study that measures the usefulness of counterfactual explanations in understanding a decision model. Code is available at* https://github.com/valeoai/OCTET.

1. Introduction

Deep learning models are now being widely deployed, notably in safety-critical applications such as autonomous driving. In such contexts, their black-box nature is a major concern, and explainability methods have been developed to improve their trustworthiness. Among them, *counterfactual explanations* have recently emerged to provide insights into a model's decision [7, 52, 55]. Given a decision model and an input query, a counterfactual explanation is a data point that differs *minimally* but *meaningfully* from the query in a way that changes the output decision of the model. By looking at the differences between the query and the explanation, a user is able to infer — by contrast — which elements were essential for the model to come to its decision. However, most counterfactual methods have only shown results for explaining classifiers trained on single-object images such as face portraits [4, 28, 29, 42, 46]. Aside from the technical difficulties of scaling up the resolution of the images, explaining decision models trained on scenes composed of many objects also present the challenge that those decisions are often multi-factorial. In autonomous driving, for example, most decisions have to take into account the position of all other road users, as well as the layout of the road and its markings, the traffic light and signs, the overall visibility, and many other factors.

Figure 1. **Counterfactual explanations generated by OCTET.** Given a classifier that predicts whether or not it is possible to go left, and a query image (*top left*), OCTET produces a counterfactual explanation where the most influential features that led to the decision are changed (*top right*). On the bottom row, we show that OCTET can also operate under different settings that result in different focused explanations. We report the prediction made by the decision model at the top left of each image.

In this paper, we present a new framework, dubbed OCTET for Object-aware CounTerfactual ExplanaTions, to generate counterfactual examples for autonomous driving. We leverage recent advances in unsupervised compositional generative modeling [14] to provide a flexible explanation method. Exploiting such a model as our backbone, we can

assess the contribution of each object of the scene independently and look for explanations in relation to their positions, their styles, or combinations of both.

To validate our claims, extensive experiments are conducted for self-driving action decision models trained with the BDD100k dataset [60] and its BDD-OIA extension [59]. Fig. 1 shows counterfactual explanations found by OCTET: given a query image and a visual decision system trained to assess if the ego vehicle is allowed to go left or not, OCTET proposes a counterfactual example with cars parked on the left side that are moved closer. When inspecting specific items (bottom row of the figure), OCTET finds that moving the yellow car to the left, or adding a double line marking on the road, are ways to change the model's decision. These explanations highlight that the absence of such elements on the left side of the road heavily influenced the model.

To sum up, our contributions are as follows:

- We tackle the problem of building counterfactual explanations for visual decision models operating on complex compositional scenes. We specifically target autonomous driving visual scenarios.
- Our method is also a tool to investigate the role of specific objects in a model's decision, empowering the user with control over which type of explanation to look for.
- We thoroughly evaluate the realism of our counterfactual images, the minimality and meaningfulness of changes, and compare against previous reference strategies. Beyond explaining classifiers, we also demonstrate the versatility of our method by addressing explanations for a segmentation network.
- Finally, we conduct a user-centered study to assess the usefulness of our explanations in a practical case. As standard evaluation benchmarks for counterfactual explanations are lacking a concrete way to measure the interpretability of the explanations, our user-centered study is a key element to validate the presented pipeline.

2. Background and Related Work

Local explanations. The overwhelming majority of deep learning based models are designed without explainability in mind, with only a few notable exceptions [8, 63]. This fact has prompted an interest in *post-hoc* explanations of already trained models that can be useful to analyze corner cases, understand failures, and find biases [15, 40]. In safety-critical applications, e.g., autonomous driving or medical imaging, explanations are especially needed for liability purposes and to foster end-user trust [45, 49, 61, 64]. Post-hoc explanations are *global* if they provide a holistic view of the main decision factors driving the model [18, 21, 32, 33], or they are *local* if they target the understanding of the model behavior on a specific input [40, 44]. Historically, the vast majority of local explanation methods are attribution-based: they generate saliency maps highlighting pixels or regions influencing the most the model's decision [5, 16, 39, 44, 47, 53, 56, 62, 66]. In the case of urban scenes, a saliency method would for instance point at the traffic light, or the presence of a pedestrian to explain why a driving model stops [31, 43]. However, saliency explanations may be misleading as they can be independent of the model at hand and merely act as edge detectors [3]. Besides, saliency methods are by nature restricted to show *where* the important regions are and they cannot indicate *what* in these regions was deemed decisive. For example, beyond highlighting things, it would be useful to know if the color of the light (red or green?) or the orientation of the pedestrian's body (towards or away from the car?) are taken into account in the model's decision. Counterfactual explanations are a step in that direction.

Counterfactual explanations. For a given decision model and query input, a counterfactual explanation is defined as a minimally but meaningfully modified version of the query that changes the decision of the model [55]. The notions of 'minimal' and 'meaningful' are crucial here. Adversarial attacks [19, 35, 48] for instance are minimal but not meaningful perturbations: they are imperceptible by design and thus not informative for the end-user [6, 17, 38]. Conversely, if not minimal, the explanation might exhibit unnecessary changes that would hinder its interpretability. While originally developed for models operating on tabular data [55], counterfactuals have been recently considered to explain deep computer vision models [20, 27, 42, 46].

Seminal work [23] explored the use of natural language to produce descriptions of elements that, in case of presence, change the image classifications. Later, retrieval-based methods aimed to explain decisions by mining the dataset for counterfactual explanations [20, 23, 51, 58]. While obtained explanations can be meaningful, the difference to the query can only be as minimal as the dataset granularity allows, limiting overall interpretability.

More recently, generator-based counterfactual explanations have been proposed [4, 27–29, 42]. They exploit the power of deep generative models that map a latent space to the data distribution. By optimizing for small manipulations in the latent space, these methods ensure for both the minimality and the meaningfulness of changes. The properties of the generative backbone and the way the latent space is structured have a crucial impact on the resulting counterfactual explanations. DIVE [42], for instance, exploits the disentangled latent space of a β-TCVAE [10] to generate diverse counterfactual explanations. However, the low visual quality of its counterfactuals hinders interpretability. DIME [28] and DVCE [4] instead build on recently popularized denoising diffusion probabilistic models [12], achieving better-looking results but losing the more curated control on the diversity. These works can explain classifiers working on 'simple' images, e.g., face portraits [34] or

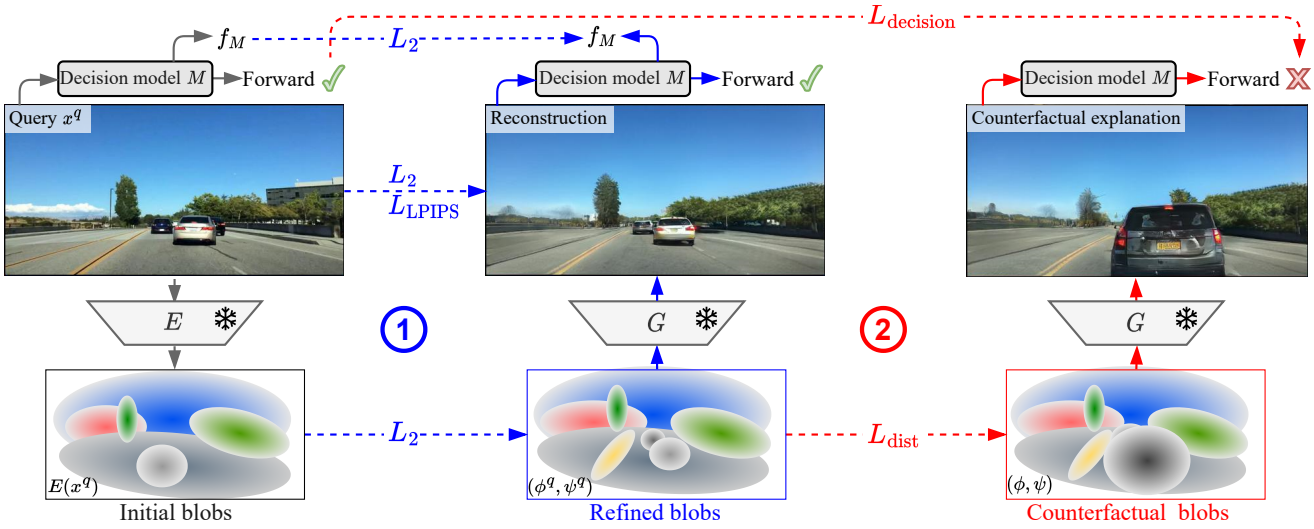

Figure 2. **Overview of OCTET.** From the query image (*top left*), a trained encoder E provides initial blob parameters. The blobs are refined in optimization stage ① using Eq. 3 to better encode the query. The refined blob parameters are again modified in optimization stage ② using Eq. 1 to obtain a counterfactual example. The parameters of the encoder E and generator G are frozen during the optimization, as is M the decision model to be explained. Losses are illustrated with dashed arrows whose directions represent the gradient flow.

featuring a single and centered object [11, 57].

Scaling up to classifiers operating over more complex images such as crowded urban scenes remains a major challenge. To date, only STEEX [27] proposes a method able to explain decision models dealing with scenes composed of many objects. It employs a segmentation-to-image generation backbone [68], which helps for generating high-quality complex images but comes with two important limitations. Firstly, it requires annotated semantic segmentation masks of images to work, which is not always available or is costly for the user to produce. Secondly, it imposes that the explanation keeps the same spatial and semantic layout as the query, thereby severely restricting the search space. It is, for instance, not able to provide insights about the importance of the position of objects. By contrast, OCTET does not impose a fixed spatial layout. Indeed, the structural layout of the scene is an intermediate representation of the generator, inferred from images only. Therefore, we do not use additional annotations, and the spatial layout can be optimized by back-propagation.

3. Object-aware Counterfactual Explanations

We present here OCTET, a method to produce counterfactual examples for compositional scenes, based on a generative architecture. We first formally describe our approach and objective in Sec. 3.1. Then, we instantiate the generative backbone in Sec. 3.2 and the counterfactual objective in Sec. 3.3. Lastly, we explain how we invert the generator to recover the latent code for the query image in Sec. 3.4. The pipeline of OCTET is shown in Fig. 2.

3.1. Goal and Notations

Given a decision model M, a query image x^q, and a target decision $y \neq M(x^q)$, a counterfactual explanation is an image x that is close to the query image x^q but modified so that the decision $M(x)$ of the model changes to y. Generative counterfactual methods, such as ours, employ a trained generator G [27, 28, 42]. We denote its latent space $\mathcal{Z}$, with $z^q \in \mathcal{Z}$ the latent code that produces the best approximation of x^q when passed through G, *i.e.*, so that $G(z^q) \approx x^q$. To explain differentiable deep vision models, the definition of a counterfactual explanation stated here-above can be translated into the following optimization problem:

$$\underset{z \in \mathcal{Z}}{\operatorname{argmin}} \, L_{\text{decision}}\big(M(G(z)), y\big) + \lambda_{\text{dist}} L_{\text{dist}}(z, z^q). \qquad (1)$$

The role of the first term L_{decision} is to push the decision of the model M for the generated image $G(z)$ to the target class y; The second term L_{dist} is a distance term in the latent space of G, which ensures that the query x^q and the explanation $G(z)$ remain similar; λ_{dist} is an hyper-parameter that controls the relative weight between the two terms. With optimal z, $G(z)$ then recovers a counterfactual image for the query x^q and the decision model M. Since the choice of L_{decision} is constrained by the nature of the model to explain, e.g., a classification loss if M is a classifier, we only have to specify G and L_{dist} to define our counterfactual method.

3.2. Compositional generative backbone

In this work, we focus on explaining decision models that operate on compositional visual scenes and we look for

a generative model capable of modeling such scenes. However, most generative models are designed for images containing centered, isolated objects, and they do not learn a latent space that can easily be disentangled into per-object representations. Instead, we propose that our generative backbone should be object-based by design [14, 37]. Such methods are fully differentiable and trained without supervision to generate images in a compositional fashion.

We choose to build on BlobGAN [14], a compositional generative model that shows good performances and has the advantage of allowing not only edition but also insertion and removal of objects in a differentiable way. In BlobGAN, images are generated in two steps. First, a *layout network* maps a random Gaussian noise vector to an intermediary representation consisting of an ordered set of K '*blobs*'. Each blob $k \in \{1 \dots K\}$ is defined by spatial parameters ϕ_k and a style feature vector ψ_k. The spatial parameters ϕ_k consist of five values that are its center coordinates, scale, aspect ratio, and rotation angle. In practice, the layout network is a neural network that outputs vectors ϕ_k and ψ_k for each blob. Then, in the second stage, the *generator network* is tasked to transform the blob parameters into an image. A rendering layer draws the blobs on a canvas as ellipses whose shapes and positions are given directly by ϕ_k. The output of the rendering layer is a map that associates to each 2D spatial location a feature vector computed from the style vectors of the blobs present at that position. This feature map is transformed by convolutional layers into an image that roughly follows the spatial organization of the canvas.

The layout network and the generator network are trained end-to-end against a discriminator with standard GAN losses only. In particular, we do not use any annotations for the number, size, class, or location of objects. They are discovered by the generator during training. Note also that scale parameters can take non-positive values. In that case, the object is considered absent and the corresponding blob is not rendered on the canvas in the generator network.

3.3. Setting L_{dist} for compositional scenes

Using a compositional generative backbone allows us to design a meaningful, disentangled, per-object distance for images. Indeed, to be interpretable with respect to the query image, a counterfactual explanation should display sparse changes in terms of the number of moved, added, removed, or modified objects. Therefore, we design L_{dist} such that it is split into per-object distances where each object can be addressed independently. Moreover, we propose that modifications of the spatial position of an object should be orthogonal to those made to its shape or colors. Since in a well-trained BlobGAN each blob is associated with a distinct element of the scene (see Appendix), such a distance can be naturally defined in the space of blob parameters.

Formally, let us denote ϕ and ψ the concatenation across all blobs $k \in \{1 \dots K\}$ of spatial parameters ϕ_k and style vectors ψ_k, and $z = (\phi, \psi)$ the complete latent representation that the generator network G can decode into an image. Accordingly, the latent code for the query image x^q reads $z^q = (\phi^q, \psi^q)$. Then, OCTET implements L_{dist} as follows:

$$L_{\text{dist}}(z, z^q) = \sum_{k=1}^{K} \|\phi_k - \phi_k^q\|_1 + \|\psi_k - \psi_k^q\|_1. \quad (2)$$

We use the L_1 distance here to promote sparsity in the number of modified blobs and the type of edits applied to each blob. If the user wishes to inspect the role of the style of objects, their spatial positions, or both, with a counterfactual explanation, she can instead optimize Eq. 2 only on the corresponding features of the blobs.

In order to specify Eq. 2, and thus the final counterfactual objective (Eq. 1), it remains to be solved how to obtain the blob-wise latent codes ϕ_k^q and ψ_k^q associated with the query image x^q. We discuss this matter in the next section.

3.4. Inverting the query image

Obtaining the latent code corresponding to the query image, a process called *inversion*, is not straightforward when using a GAN-based generator. To produce a counterfactual example, such inversion needs to find latent codes (ϕ^q, ψ^q) so that $G(\phi^q, \psi^q)$ recovers x^q. For the downstream counterfactual optimization to make sense, the decision of the model for the query image should be preserved for the reconstructed image, *i.e.*, $M(G(\phi^q, \psi^q)) = M(x^q)$.

To achieve those objectives, we follow best practices for image inversion [1, 2, 41, 50, 67]. We first learn an encoder E to predict blob parameters from input images. Generating an image $G(E(x^q))$ with parameters predicted this way yields a coarse reconstruction of x^q. To improve the inversion, we start from predicted blob parameters $E(x^q)$ and further optimize them with the following objective:

$$\begin{aligned} \phi^q, \psi^q = \arg\min_{\phi,\psi} \; & L_{\text{LPIPS}}(G(\phi,\psi), x^q)) + L_2(G(\phi,\psi), x^q) \\ & + L_2(f_M(x^q), f_M(G(\phi,\psi))) \\ & + L_2((\phi,\psi), E(x^q)). \end{aligned} \quad (3)$$

The first two terms ensure that the image $G(\phi, \psi)$ generated from the blob parameters reconstructs the query x^q using a perceptual L_{LPIPS} loss and the L_2 loss. The third term aims to preserve the features that are important to the decision model M by adding a L_2 loss on intermediate features $f_M(x)$ of M. Finally, the last term encourages the parameters to stay close to the reasonably good solution given by the encoder. This inversion method concludes the specification of the optimization problem of Eq. 1 in OCTET and of the pipeline depicted in Fig. 2. More details on the inversion and the training of the encoder are in the appendix.

Figure 3. **Counterfactual explanations on driving scenes**. Explanations are generated for binary classifiers predicting the capacity of the ego-car to respectively Move forward, Stop, turn Left, or Right. We indicate at the top left of each image the output decision of the model for that image. For each row, counterfactuals target the opposite decision to that obtained for the query image. OCTET finds interpretable, sparse, and meaningful semantic and spatial modifications to the query image. OCTET *style-only* only optimizes on the style features, it is analog to STEEX [27] that can only search for changes in object appearance to flip the decision.

4. Experiments

4.1. Experimental protocol and training details

Data and decision model. Our method is designed to build counterfactual examples to explain decision models trained on compositional images. To evaluate our method, we follow previous work [27] and use the BDD100k dataset [60], which contains 100k images of diverse driving scenes that we resize to a resolution of 512x256. In addition, we also use BDD-OIA [59], a 20K scene extension of BDD100k that is annotated with 4 binary attributes labeled *Move Forward*, *Stop*, *Turn Left*, and *Turn Right*. Each label is defined as the capacity in the given situation of the ego-vehicle to perform the corresponding action. Note that by this definition, being allowed to *Stop* and to *Move Forward* are not mutually exclusive. The decision models being explained in the experiments are multi-label binary DenseNet121 [26] classifiers trained on BDD-OIA with the four aforementioned labels.

Baseline. We compare our method against STEEX [27], the only counterfactual explanation method that performs well on compositional images like driving scenes. Note that in STEEX, the conservation of the semantic layout is hard-coded in the architecture: it is less flexible as an explanation method than ours. Moreover, the authors use semantic segmentation annotations of BDD100k to train their generative backbone, while we do not need any additional annotation. We use the pre-trained model from the official release.

Technical setup. For our method, we train the generative backbone (BlobGAN) on the training set of BDD100k, with a number $K = 40$ of blobs. We adapt the architecture of BlobGAN to generate rectangular 512×256 images, and we decrease the size of the feature vectors of the BlobGAN to compensate for the increased number of blobs. Throughout our experiments, we evaluate two main variants of our method: OCTET where both spatial and style features are optimized altogether and OCTET *style-only* where only the style vector ψ is optimized while the spatial vector ϕ is left equal to ϕ^q. This setting only assesses the contribution of style features, and is most similar to the STEEX baseline [27] that finds explanations while conserving the semantic layout of the query. More details in the appendix.

4.2. Quality of the explanations

Counterfactual explanations are defined as images that are similar to the query image but effectively change the decision of the target model. To verify that our explanations fit the criteria, we evaluate them with the following metrics: 1) the FID [24] which measures if they are realistic images, and 2) a perceptual distance LPIPS [65] that mea-

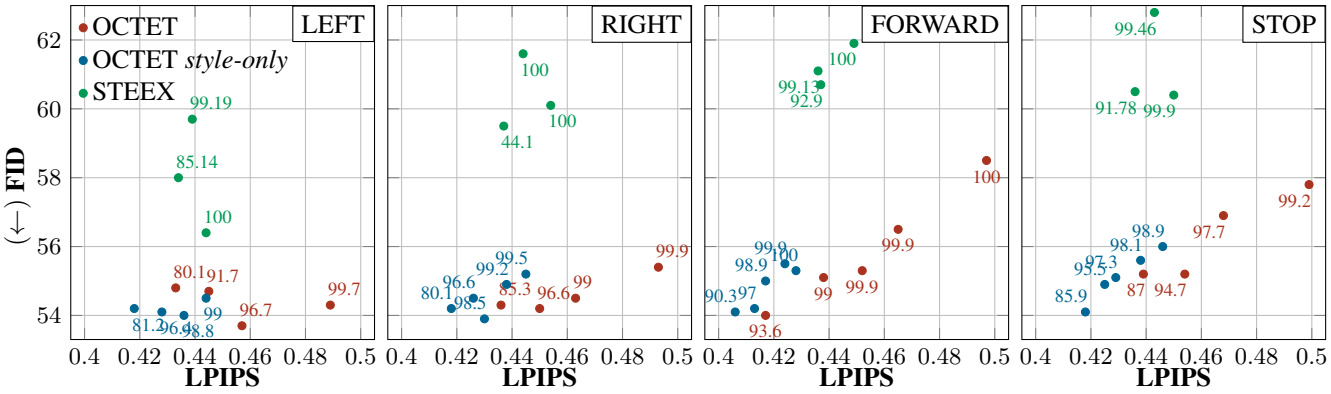

Figure 4. **FID vs. LPIPS, labeled with Success rate** for OCTET and STEEX [27] explaining a decision model trained on the 4 labels of BDD-OIA. Data points are obtained with different values of λ_{dist}, yielding different trade-offs. STEEX is directly comparable to OCTET *style-only* as they operate in similar conditions. See Sec. 4.2.

sures the change magnitude between them and the query image. We also compute the success rate corresponding to the percentage of counterfactual examples that are indeed classified in the target class by the decision model. We compute these metrics for different values of λ_{dist} that lead to different trade-offs and plot the scores in Fig. 4. As STEEX explanations are constrained to style features, we compare it to our OCTET *style-only* setting. Firstly, in terms of FID, we can observe that both versions of OCTET obtain lower FID than STEEX: OCTET produces explanations of better visual quality, even though it does not use any annotations as opposed to STEEX, which needs semantic maps, and handles spatial explanations that are more challenging to generate. Secondly, in terms of LPIPS, for style-based explanations OCTET *style-only* stays closer to the query than STEEX. This can be explained as STEEX latent representation is semantic-based, while ours is instance-based. Our method is therefore able to seek finer-grained modifications, while STEEX tends to change the style for an entire semantic class even when only one instance is important to the decision model. As for OCTET, because LPIPS disproportionately weights structural modifications, the wide range of attainable values indicates that many degrees of changes can be obtained depending on the choice of λ_{dist}. Some settings tend to move objects a lot, while other settings attain values similar to '*style-only*' edits. Finally, the very high achievable Success Rates displayed by all three models confirm that they will propose explanations in almost any situation.

We display examples of OCTET's counterfactual explanations in Fig. 3. OCTET finds sparse spatial and/or semantic modifications to objects that strongly influence the output decision. For example, we can see with OCTET that the distance with the car in front plays a role when deciding for 'Forward' (1st row) or 'Stop' (2nd row). On the other hand, OCTET *style-only* shows that for those same decisions, the brake lights also are taken into account. For going 'Left' class (3rd row), the absence of the continuous line seems

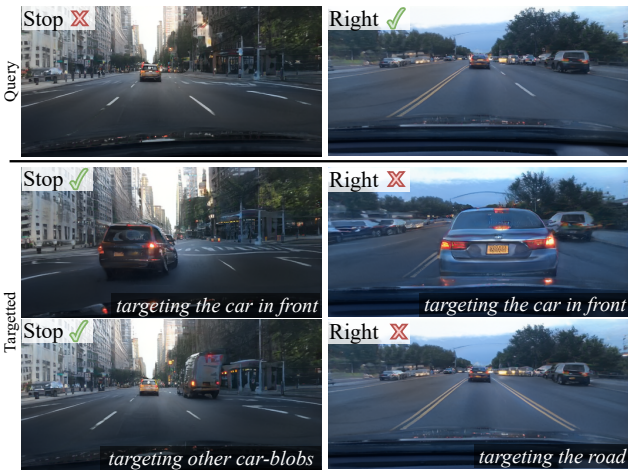

Figure 5. **Targeted counterfactual explanations**. For each query image (top row), we target different blobs (2nd and 3rd row) to inspect the role of corresponding objects. The output of the decision model is reported on the top right corner for each image.

crucial. On the 3rd and 4th row, we can see that OCTET mixes both style (brake lights) and spatial (proximity) explanations. On the same tasks, STEEX changes the style of the whole class, and not just the necessary instance.

4.3. Targeted counterfactuals

Models that operate on compositional scenes can have their decisions influenced by many different factors, and we have shown that OCTET can expose those factors in its counterfactual examples. However, the user may want to test some particular hypothesis dealing with specific objects or areas of the scene. Accordingly, the user can decide to optimize Eq. 2 only on selected blobs parameters. Finding the blob corresponding to a specific object in the image is discussed in Appendix. For instance, in the situation shown at the top left of Fig. 5, the car at the front seems to have

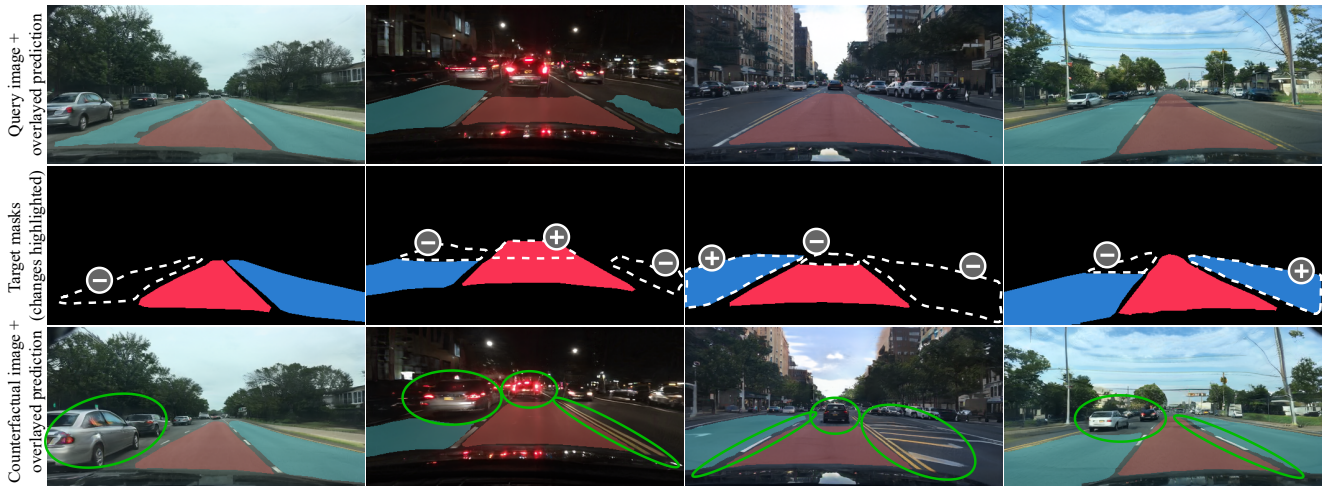

Figure 6. **Counterfactual explanations for a segmentation model.** The segmentation model predicts the driveable area with primary and secondary lanes shown in red and blue respectively. The first row shows query images and segmentations by the model. In the second row, we add or remove parts of the original masks to build target masks (changes are shown with white dotted lines). The third row displays counterfactual explanations for the target masks and predictions on the new images. Green ellipses are manually added to highlight details.

its brake lights on, but the decision is still not to 'Stop'. Targeting the front car (2nd row left image) reveals that the decision would have been 'Stop' if the car was closer, and the brake light more clearly contrasted. OCTET can also target blobs that are currently inactive, corresponding to no visible object in the image. In this example, doing so for car blobs shows that if another vehicle also had brake lights on, the model would also decide to 'Stop' (3rd row left image). This finding is unexpected, and could warrant further investigation of the decision model. We show in Fig. 5 and Fig. 1 some other results we obtain by targeting specific blobs.

4.4. Explaining a semantic segmentation model

For complex tasks, like autonomous driving, decision models usually rely on multiple neural networks that address different subtasks (e.g., traffic sign recognition, depth estimation, motion forecasting). In this context, it is important to explain models beyond classifiers [25,54], a question not addressed in previous counterfactual methods. Here, we show the versatility of our method as we extend it to explain a semantic segmentation network. To do so, we use the same optimization problem described in Eq. 1 and Sec. 3 but we change L_{decision} to an image segmentation loss.

In Fig. 6, we show counterfactual explanations for a driveable area segmentation model. More precisely, OCTET explains outputs of a DeepLabv3 [9] trained to segment images from BDD100k into classes of driveable area: the primary lane, the secondary side lanes, and the background. We define the target masks by manually modifying predictions made on the query images as illustrated in the second row. We see in the obtained counterfactuals, in the third row, changes in road markings and vehicles' positions. They confirm that the segmentation model will consider as driveable a lane behind broken white lines (col. 3) but not behind a double yellow line (col. 2), especially if, in addition, the lane is hatched (col. 3). If there is a car on a driveable lane, the model can detect that the occupied portion is non-driveable while considering that the rest of the lane remains accessible. With these examples, OCTET provides explanations for both large semantic regions, e.g., when we add or remove an entire region in the target mask, or small subparts, e.g., when we slightly crop or extend a region.

4.5. User-study to assess explanations

Metrics presented in Sec. 4.2 measure how well the produced examples match the definition of counterfactual explanations. We want here to assess how *useful* the explanations are for a user to better understand a decision model. However, designing such an evaluation is challenging. Indeed, it has to avoid the many biases that appear [13, 36]. Moreover, we usually do not have access to the inner workings of a model, and therefore we cannot compare users' insights on its behavior against a ground truth. To overcome these challenges, we take inspiration from recent work on the usefulness of saliency maps as explanations [15] and we adapt its evaluation protocol to counterfactual explanations.

The goal is to measure if accessing counterfactual explanations on some examples helps users to better *understand* the model. Grounded within previous literature [16, 22, 30], we instantiate *understand* as 'being able to predict the model's output for new instances', a concept known as *simulatability*. In particular, our user-study has two phases:

	Cohort size	Replication	Bias Detection
Group w/ OCTET	20	**70%**	**65%**
Group control	20	52.5%	0%

Table 1. **Results of the user-study.** The performance of users who had access to explanations ('w/ OCTET') in the observation phase is compared to that of users who didn't ('control'). 'Replication' measures the accuracy of the users in predicting the model's output on new samples and 'bias detection' is the proportion of users who found the spurious correlation in the model.

1. *Observation phase*: participants are shown a series of triplets, each being composed of (1) a query image, (2) the model's decision on that image, and (3), a counterfactual explanation for the decision.
2. *Questionnaire phase*: participants are presented with new images and are asked to guess the output of the model on these images. We stress that during the questionnaire, no counterfactual images are shown.

To measure the impact that counterfactual explanations of the first phase have on the users' ability to predict the model output on new samples (second phase), we use a control group. This control group, composed of different participants, undertakes the exact same tasks except that counterfactual explanations are *not* shown during the first phase. The decision model used in this study is a classifier trained on BDD-OIA [59] to predict the binary 'Turn Right' label. Besides, the decision model is trained, by design, with a flaw: the presence of a car on any side of the road can impact the decision, not only obstacles on the right side. As discussed further below, this bias is introduced to study if users are able to identify model defects thanks to counterfactuals. More details are provided in the appendix.

Replication score. We first measure the 'replication score': the proportion of correct match between the user's answer and the prediction of the decision model, averaged over participants of the group. We report these results in Tab. 1 and observe that participants in the group accessing counterfactuals during the observation phase can successfully guess the decision model's prediction 70% of the time while the control group's performance is 52.5%, barely above random. A statistical unpaired t-test, with as null hypothesis that the counterfactual explanations have no effect, confirms the significance of our result with a p-value of 0.0028. This demonstrates the usefulness of OCTET in understanding the decision model as participants with explanations could better anticipate the model's behavior on new instances.

Bias detection. Another promise of counterfactual explanations is to enable bias detection. We then want to assess whether, by looking at explanations, users can tell if and how a decision model is biased. In that direction, previous works dealing with human face datasets train biased models by artificially confounding attributes, e.g., smile and age, in the training set. Then, they develop bias detection benchmarks that measure, by means of an oracle neural network, how well attributes remain confounded in the counterfactual explanations [28, 42, 46]. However, a serious limitation of that benchmark is that it keeps end-users out of the evaluation loop and it cannot tell whether or not a user will be able to uncover the spurious correlations by looking at the counterfactual explanation. Moreover, detecting the issue with the model can be especially difficult when the bias is not obvious or its presence not suspected in the first place. Instead of an automated evaluation, at the end of the questionnaire, we asked the participants to describe in free text what they had understood from the model. The goal is to prompt them to mention the issue we embedded in the decision model, i.e., that the presence of a car on the left-hand side of the image factors in the 'Turn Right' decision. The 'bias detection' score simply counts, for each group, the proportion of participants that used the word 'left' in their answers. In Tab. 1, we see that 65% of the participants that had access to the explanations from OCTET identify the spurious correlation learned by the model, while none of the users that did not have the explanations (control group) mentions it. This clearly shows that our method is useful to pinpoint issues with the decision model. Also, the fact that not every participant detected it confirms our suspicions against automated bias detection benchmarks: the presence of the information in the counterfactual examples is not sufficient to ensure that the issue will be found in practice.

5. Conclusion

In this work, we presented OCTET, a generative method to produce counterfactual explanations for deep vision models working on complex scenes. Thanks to an object-centric representation of the scene, OCTET is able to process complex compositional scenes and to assess the impact of both the spatial positions and the style features of each object. Our tool allows in-depth exploration of the different explaining factors by letting the user adjust on the fly the weights of the different search directions independently. We evaluated our method thoroughly on real driving scenes from BDD100k, with an improved evaluation benchmark compared to previous works, including a human-centered study that confirmed the usefulness of the approach.

Acknowledgements

This work was supported in part by the ANR grants VISA DEEP (ANR-20-CHIA-0022) and MultiTrans (ANR-21-CE23-0032). Authors would like to thank the voluntary participants of the user-study, as well as Remi Cadene, Julien Colin, Thomas Fel, and Guillaume Jeanneret for helpful discussions.

References

[1] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan: How to embed images into the stylegan latent space? In *ICCV*, 2019. 4

[2] Rameen Abdal, Yipeng Qin, and Peter Wonka. Image2stylegan++: How to edit the embedded images? In *CVPR*, 2020. 4

[3] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian J. Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. In *NeurIPS*, 2018. 2

[4] Maximilian Augustin, Valentyn Boreiko, Francesco Croce, and Matthias Hein. Diffusion visual counterfactual explanations. In *NeurIPS*, 2022. 1, 2

[5] Mariusz Bojarski, Anna Choromanska, Krzysztof Choromanski, Bernhard Firner, Larry J Ackel, Urs Muller, Phil Yeres, and Karol Zieba. Visualbackprop: Efficient visualization of cnns for autonomous driving. In *ICRA*, 2018. 2

[6] Kieran Browne and Ben Swift. Semantics and explanation: why counterfactual explanations produce adversarial examples in deep neural networks. *CoRR*, abs/2012.10076, 2020. 2

[7] Chun-Hao Chang, Elliot Creager, Anna Goldenberg, and David Duvenaud. Explaining image classifiers by counterfactual generation. In *ICLR*, 2019. 1

[8] Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan Su. This looks like that: Deep learning for interpretable image recognition. In *NeurIPS*, 2019. 2

[9] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *CoRR*, abs/1706.05587, 2017. 7

[10] Ricky T. Q. Chen, Xuechen Li, Roger Grosse, and David Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *NeurIPS*, 2018. 2

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 3

[12] Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat gans on image synthesis. In *NeurIPS*, 2021. 2

[13] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv preprint arXiv:1702.08608*, 2017. 7

[14] Dave Epstein, Taesung Park, Richard Zhang, Eli Shechtman, and Alexei A. Efros. Blobgan: Spatially disentangled scene representations. In *ECCV*, 2022. 1, 4

[15] Thomas Fel, Julien Colin, Rémi Cadène, and Thomas Serre. What I cannot predict, I do not understand: A human-centered evaluation framework for explainability methods. In *NeurIPS*, 2022. 2, 7

[16] Ruth C. Fong and Andrea Vedaldi. Interpretable explanations of black boxes by meaningful perturbation. In *ICCV*, 2017. 2, 7

[17] Timo Freiesleben. Counterfactual explanations & adversarial examples - common grounds, essential differences, and potential transfers. *CoRR*, abs/2009.05487, 2020. 2

[18] Nicholas Frosst and Geoffrey E. Hinton. Distilling a neural network into a soft decision tree. In *Workshop on Comprehensibility and Explanation in AI and ML @AI*IA*, 2017. 2

[19] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *ICLR*, 2015. 2

[20] Yash Goyal, Ziyan Wu, Jan Ernst, Dhruv Batra, Devi Parikh, and Stefan Lee. Counterfactual visual explanations. In *ICML*, 2019. 2

[21] Michael Harradon, Jeff Druce, and Brian E. Ruttenberg. Causal learning and explanation of deep neural networks via autoencoded activations. *CoRR*, 2018. 2

[22] Peter Hase and Mohit Bansal. Evaluating explainable AI: which algorithmic explanations help users predict model behavior? In *ACL*, 2020. 7

[23] Lisa Anne Hendricks, Ronghang Hu, Trevor Darrell, and Zeynep Akata. Grounding visual explanations. In *ECCV*, 2018. 2

[24] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017. 5

[25] Lukas Hoyer, Mauricio Munoz, Prateek Katiyar, Anna Khoreva, and Volker Fischer. Grid saliency for context explanations of semantic segmentation. In *NeurIPS*, 2019. 7

[26] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 5

[27] Paul Jacob, Éloi Zablocki, Hedi Ben-Younes, Mickaël Chen, Patrick Pérez, and Matthieu Cord. STEEX: steering counterfactual explanations with semantics. In *ECCV*, 2022. 2, 3, 5, 6

[28] Guillaume Jeanneret, Loïc Simon, and Frédéric Jurie. Diffusion models for counterfactual explanations. *CoRR*, abs/2203.15636, 2022. 1, 2, 3, 8

[29] Saeed Khorram and Fuxin Li. Cycle-consistent counterfactuals by latent transformations. In *CVPR*, 2022. 1, 2

[30] Been Kim, Oluwasanmi Koyejo, and Rajiv Khanna. Examples are not enough, learn to criticize! criticism for interpretability. In *NeurIPS*, 2016. 7

[31] Jinkyu Kim and John F. Canny. Interpretable learning for self-driving cars by visualizing causal attention. In *ICCV*, 2017. 2

[32] Oran Lang, Yossi Gandelsman, Michal Yarom, Yoav Wald, Gal Elidan, Avinatan Hassidim, William T. Freeman, Phillip Isola, Amir Globerson, Michal Irani, and Inbar Mosseri. Explaining in style: Training a GAN to explain a classifier in stylespace. In *ICCV*, 2021. 2

[33] Zhiheng Li and Chenliang Xu. Discover the Unknown Biased Attribute of an Image Classifier. In *ICCV*, 2021. 2

[34] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *ICCV*, 2015. 2

[35] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: A simple and accurate method to fool deep neural networks. In *CVPR*, 2016. 2

[36] Giang Nguyen, Daeyoung Kim, and Anh Nguyen. The effectiveness of feature attribution methods and its correlation with automatic evaluation scores. In *NeurIPS*, 2021. 7

[37] Michael Niemeyer and Andreas Geiger. GIRAFFE: representing scenes as compositional generative neural feature fields. In *CVPR*, 2021. 4

[38] Martin Pawelczyk, Shalmali Joshi, Chirag Agarwal, Sohini Upadhyay, and Himabindu Lakkaraju. On the connections between counterfactual explanations and adversarial examples. *CoRR*, abs/2106.09992, 2021. 2

[39] Sylvestre-Alvise Rebuffi, Ruth Fong, Xu Ji, and Andrea Vedaldi. There and back again: Revisiting backpropagation saliency methods. In *CVPR*, 2020. 2

[40] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you?": Explaining the predictions of any classifier. In *SIGKDD*, 2016. 2

[41] Elad Richardson, Yuval Alaluf, Or Patashnik, Yotam Nitzan, Yaniv Azar, Stav Shapiro, and Daniel Cohen-Or. Encoding in style: A stylegan encoder for image-to-image translation. In *CVPR*, 2021. 4

[42] Pau Rodríguez, Massimo Caccia, Alexandre Lacoste, Lee Zamparo, Issam H. Laradji, Laurent Charlin, and David Vázquez. Beyond trivial counterfactual explanations with diverse valuable explanations. In *ICCV*, 2021. 1, 2, 3, 8

[43] Axel Sauer, Nikolay Savinov, and Andreas Geiger. Conditional affordance learning for driving in urban environments. In *CoRL*, 2018. 2

[44] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, 2017. 2

[45] Yuan Shen, Shanduojiao Jiang, Yanlin Chen, Eileen Yang, Xilun Jin, Yuliang Fan, and Katie Driggs Campbell. To explain or not to explain: A study on the necessity of explanations for autonomous vehicles. *CoRR*, 2020. 2

[46] Sumedha Singla, Brian Pollack, Junxiang Chen, and Kayhan Batmanghelich. Explanation by progressive exaggeration. In *ICLR*, 2020. 1, 2, 8

[47] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *ICML*, 2017. 2

[48] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *ICLR*, 2014. 2

[49] Erico Tjoa and Cuntai Guan. A survey on explainable artificial intelligence (XAI): towards medical XAI. *CoRR*, abs/1907.07374, 2019. 2

[50] Omer Tov, Yuval Alaluf, Yotam Nitzan, Or Patashnik, and Daniel Cohen-Or. Designing an encoder for stylegan image manipulation. *ACM Transactions on Graphics (TOG)*, 2021. 4

[51] Simon Vandenhende, Dhruv Kumar Mahajan, Filip Radenović, and Deepti Ghadiyaram. Making heads or tails: Towards semantically consistent visual counterfactuals. In *ECCV*, 2022. 2

[52] Sahil Verma, John P. Dickerson, and Keegan Hines. Counterfactual explanations for machine learning: A review. *CoRR*, abs/2010.10596, 2020. 1

[53] Tom Vermeire, Dieter Brughmans, Sofie Goethals, Raphael Mazzine Barbossa de Oliveira, and David Martens. Explainable image classification with evidence counterfactual. *Pattern Anal. Appl.*, 2022. 2

[54] Kira Vinogradova, Alexandr Dibrov, and Gene Myers. Towards interpretable semantic segmentation via gradient-weighted class activation mapping (student abstract). In *AAAI*, 2020. 7

[55] Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harvard Journal of Law & Technology*, 2017. 1, 2

[56] Jörg Wagner, Jan Mathias Köhler, Tobias Gindele, Leon Hetzel, Jakob Thaddäus Wiedemer, and Sven Behnke. Interpretable and fine-grained visual explanations for convolutional neural networks. In *CVPR*, 2019. 2

[57] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. 3

[58] Pei Wang and Nuno Vasconcelos. SCOUT: self-aware discriminant counterfactual explanations. In *CVPR*, 2020. 2

[59] Yiran Xu, Xiaoyin Yang, Lihang Gong, Hsuan-Chu Lin, Tz-Ying Wu, Yunsheng Li, and Nuno Vasconcelos. Explainable object-induced action decision for autonomous vehicles. In *CVPR*, 2020. 2, 5, 8

[60] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. BDD100K: A diverse driving dataset for heterogeneous multitask learning. In *CVPR*, 2020. 2, 5

[61] Éloi Zablocki, Hédi Ben-Younes, Patrick Pérez, and Matthieu Cord. Explainability of deep vision-based autonomous driving systems: Review and challenges. *Int. J. Comput. Vis.*, 2022. 2

[62] Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *ECCV*, 2014. 2

[63] Quanshi Zhang, Ying Nian Wu, and Song-Chun Zhu. Interpretable convolutional neural networks. In *CVPR*, 2018. 2

[64] Qiaoning Zhang, X. Jessie Yang, and Lionel Peter Robert. Expectations and trust in automated vehicles. In *CHI*, 2020. 2

[65] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 5

[66] Bolei Zhou, Aditya Khosla, Àgata Lapedriza, Aude Oliva, and Antonio Torralba. Object detectors emerge in deep scene cnns. In *ICLR*, 2015. 2

[67] Jiapeng Zhu, Yujun Shen, Deli Zhao, and Bolei Zhou. In-domain gan inversion for real image editing. In *ECCV*, 2020. 4

[68] Peihao Zhu, Rameen Abdal, Yipeng Qin, and Peter Wonka. SEAN: image synthesis with semantic region-adaptive normalization. In *CVPR*, 2020. 3