Making Heads or Tails: Towards Semantically Consistent Visual Counterfactuals

Simon Vandenhende Dhruv Mahajan
Filip Radenovic* Deepti Ghadiyaram*

Meta AI

Abstract. A visual counterfactual explanation replaces image regions in a query image with regions from a distractor image such that the system's decision on the transformed image changes to the distractor class. In this work, we present a novel framework for computing visual counterfactual explanations based on two key ideas. First, we enforce that the *replaced* and *replacer* regions contain the same semantic part, resulting in more semantically consistent explanations. Second, we use multiple distractor images in a computationally efficient way and obtain more discriminative explanations with fewer region replacements. Our approach is **27**% more semantically consistent and an order of magnitude faster than a competing method on three fine-grained image recognition datasets. We highlight the utility of our counterfactuals over existing works through machine teaching experiments where we teach humans to classify different bird species. We also complement our explanations with the vocabulary of parts and attributes that contributed the most to the system's decision. In this task as well, we obtain state-of-the-art results when using our counterfactual explanations relative to existing works, reinforcing the importance of semantically consistent explanations. Source code is available at github.com/facebookresearch/visual-counterfactuals.

1 Introduction

Explainable AI (XAI) research aims to develop tools that allow lay-users to comprehend the reasoning behind an AI system's decisions [34,61]. XAI tools are critical given the pervasiveness of computer vision technologies in various human-centric applications such as self-driving vehicles, healthcare systems, and facial recognition tools. These tools serve several purposes [2,57]: (i) they help users understand why a decision was reached thereby making systems more transparent, (ii) they allow system developers to improve their system, and (iii) they offer agency to users affected by the system's decision to change the outcome.

One intuitive way to explain a system's decision is through counterfactual explanations [56,57] which describe *in what way* a data instance would need to be different in order for the system to reach an *alternate* conclusion. In this work, we study counterfactual explanations for fine-grained image recognition tasks,

* Equal contribution.

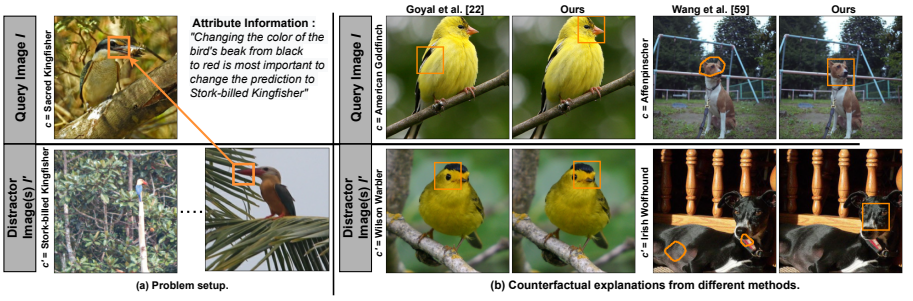

Fig. 1: **Paper overview. (a)** Given a query image I (top row) from class c, we provide counterfactual explanations relative to a distractor image I' (bottom row) from class c'. The explanations highlight what regions in I should be replaced from I' for the transformed image to be classified as c'. We also use attribute information to identify the region attributes that contributed the most for a counterfactual. **(b)** Unlike [22,59], our explanations identify regions that are both discriminative and semantically similar.

where the most confusing classes are often hard to distinguish. The difficulty of this problem makes it a particularly well suited setting to study intuitive and human-understandable explanations. Figure 1-a presents a *query image* I and a *distractor image* I' belonging to the categories *Sacred Kingfisher* (c) and *Stork-billed Kingfisher* (c'), respectively. Given a black-box classification model, a counterfactual explanation aims to answer: "how should the query image I change for the model to predict c' instead of c?" To do this, we utilize the distractor image I' (or a set of distractor images) and identify which regions in I should be replaced with regions from I' for the model's prediction to be c'.

Counterfactual visual explanations are under-explored [22,59], and most popular XAI methods use saliency maps [17,21,38,44,63] or feature importance scores [18,29,33,41,42,51,65] to highlight what image regions or features most contribute to a model's decision. Unlike counterfactual explanations, these methods do not consider alternate scenarios which yield a different result. Additionally, some of these methods [33,41,42] extract explanations via a local model approximation, leading to explanations that are *unfaithful* [3,50], i.e., they misrepresent the model's behavior. By contrast, current counterfactual explanations are faithful by design as they operate on the original model's output to generate explanations. Further, counterfactuals share similarities with how children learn about a concept – by contrasting with other related concepts [9,11]. As studied in [35,56,57], an ideal counterfactual should have the following properties: (i) the highlighted regions in the images I, I' should be <u>discriminative</u> of their respective classes; (ii) the counterfactual should be sensible in that the replaced regions should be <u>semantically consistent</u>, i.e., they correspond to the same object parts; and, (iii) the counterfactual should make as few changes as possible to the query image I as humans find sparse explanations <u>easier to understand</u>.

Prior works [22,59] proposed ways to identify the most discriminative image regions to generate counterfactual explanations. However, naively applying this principle can yield degenerate solutions that are semantically inconsistent. Figure 1-b visualizes such scenarios, where prior works [22,59] replace image re-

gions corresponding to different object parts (e.g., [22] replaces bird's wing in I with a head in I'). Further, these methods rely on a single distractor image I', which often limits the variety of discriminative regions to choose from, leading to explanations that are sometimes less discriminative hence uninformative.

This paper addresses these shortcomings. Specifically, we propose a novel and computationally efficient framework that produces both discriminative and semantically consistent counterfactuals. Our method builds on two key ideas. First, we constrain the identified class-specific image regions that alter a model's decision to allude to the same semantic parts, yielding more semantically consistent explanations. Since we only have access to object category labels, we impose this as a soft constraint in a separate auxiliary feature space learned in a self-supervised way. Second, contrary to prior works, we expand the search space by using multiple distractor images from a given class leading to more discriminative explanations with fewer regions to replace. However, naively extending to multiple distractor images poses a computational bottleneck. We address this by constraining the processing to only the most similar regions by once again leveraging the soft constraint, resulting in an order of magnitude speedup.

Our approach significantly outperforms the s-o-t-a [22,59] across several metrics on three datasets – CUB [58], Stanford-Dogs [28], and iNaturalist-2021 [55] and yields more semantically consistent counterfactuals (Fig. 1-b). While prior work [22] suffers computationally when increasing the number of distractor images, the optimization improvements introduced in our method make it notably efficient. We also study the properties of the auxiliary feature space and justify our design choices. Further, we show the importance of generating semantically consistent counterfactuals via a machine teaching task where we teach lay-humans to recognize bird species. We find that humans perform better when provided with our semantically consistent explanations relative to others [22,59].

We further reinforce the importance of semantically consistent counterfactuals by proposing a method to complement our explanations with the vocabulary of parts and attributes. Consider Fig. 1-a, where the counterfactual changes both the color of the beak and forehead. Under this setup, we provide nameable parts and attributes corresponding to the selected image regions and inform what attributes contributed the most to the model's decision. For example, in Fig. 1-a, our explanation highlights that the beak's color mattered the most. We find that our explanations identify class discriminative attributes – those that belong to class c but not to c', or vice versa – and are more interpretable.

In summary, our contributions are: **(i)** we present a framework to compute semantically consistent and faithful counterfactual explanations by enforcing the model to only replace semantically matching image regions (Sec. 3.2), **(ii)** we leverage multiple distractor images in a computationally efficient way, achieve an order of magnitude speedup, and generate more discriminative and sparse explanations (Sec. 3.3), **(iii)** we highlight the utility of our framework through extensive experiments (Sec. 4.2 - 4.3) and a human-in-the-loop evaluation through machine teaching (Sec. 4.4), **(iv)** we augment visual counterfactuals with nameable part and attribute information (Sec. 5).

2 Related Work

Feature attribution methods [6] rely on the back propagation algorithm [8,40,44,45,46,62,63] or input perturbations [15,17,20,21,38,65] to identify the image regions that are most important to a model's decision. However, none of these methods can tell how the image should change to get a different outcome. **Counterfactual explanations** [36,39,56,57] transform a query image I of class c such that the model predicts class c' on the transformed image. In computer vision, several works [5,25,26,31,32,43,48,49] used a generative model to synthesize counterfactual examples. However, the difficulties of realistic image synthesis can limit these methods [25,32,43,48] to small-scale problems. A few works [5,26,49] guided the image generation process via pixel-level supervision to tackle more complex scenes. StyleEx [31] uses the latent space of a StyleGAN [27] to identify the visual attributes that underlie the classifier's decision. Despite these efforts, it remains challenging to synthesize realistic counterfactual examples. Our method does not use a generative model but is more related to the works discussed next.

A second group of works [4,22,59] finds the regions or concepts in I that should be changed to get a different outcome. CoCoX [4] identifies visual concepts to add or remove to change the prediction. Still, the most popular methods [22,59] use a distractor image I' from class c' to find and replace the regions in I that change the model's prediction to c'. SCOUT [59] finds these regions via attribute maps. Goyal *et al.* [22] use spatial features of the images to construct counterfactuals. These methods have two key advantages. First, the distractor images are often readily available and thus inexpensive to obtain compared to pixel-level annotations [5,26,49]. Second, these methods fit well with fine-grained recognition tasks, as they can easily identify the distinguishing elements between classes. Our framework follows a similar strategy but differs in two crucial components. First, we enforce that the replaced regions are semantically consistent. Second, our method leverages multiple distractor images in an efficient way.

3 Method

Our key goal is to: (i) generate a counterfactual that selects discriminative and semantically consistent regions in I and I' without using additional annotations, (ii) leverage multiple distractor images efficiently. We first review the foundational method [22] for counterfactual generation that our framework builds on and then introduce our approach, illustrated in Fig. 2.

3.1 Counterfactual problem formulation: preliminaries

Consider a deep neural network with two components: a spatial feature extractor f and a decision network g. Note that any neural network can be divided into such components by selecting an arbitrary layer to split at. In our setup, we split a network after the final down-sampling layer. The spatial feature extractor $f : \mathcal{I} \rightarrow \mathbb{R}^{hw \times d}$ maps the image to a $h \times w \times d$ dimensional spatial feature,

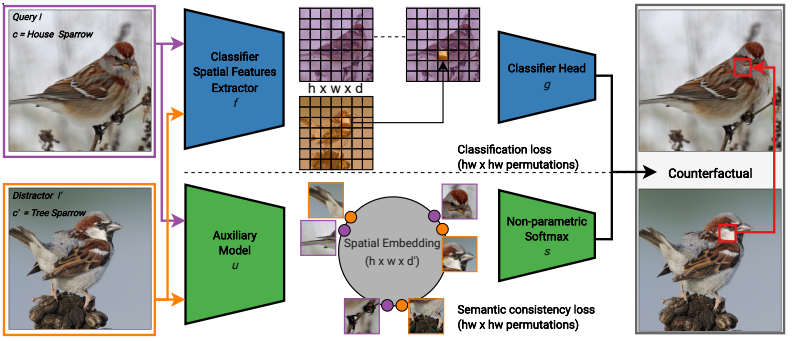

Fig. 2: **Our counterfactual explanation** identifies regions in a query image I from class c and a distractor image I' from class c' such that replacing the regions in I with the regions in I' changes the model's outcome to c'. Instead of considering actual image regions, we operate on $h \times w$ cells in the spatial feature maps. The cells are selected based upon: (i) a classification loss that increases the predicted probability $g_{c'}$ of class c' and (ii) a semantic consistency loss that selects cells containing the same semantic parts. We use a self-supervised auxiliary model to compute the semantic loss.

reshaped to a $hw \times d$ spatial cell matrix, where h and w denote the spatial dimensions and d the number of channels. The decision network $g : \mathbb{R}^{hw \times d} \rightarrow \mathbb{R}^{|\mathcal{C}|}$ takes the spatial cells and predicts probabilities over the output space $\mathcal{C}$. Further, let query and distractor image $I, I' \in \mathcal{I}$ with class predictions $c, c' \in \mathcal{C}$.

Following [22], we construct a counterfactual I^* in the feature space $f(.)$ by replacing spatial cells in $f(I)$ with cells from $f(I')$ such that the classifier predicts c' for I^*. This is done by first rearranging the cells in $f(I')$ to align with $f(I)$ using a permutation matrix $P \in \mathbb{R}^{hw \times hw}$, then selectively replacing entries in $f(I)$ according to a sparse binary gating vector $\mathbf{a} \in \mathbb{R}^{hw}$. Let $\circ$ denote the Hadamard product. The transformed image I^* can be written as:

$$f(I^*) = (\mathbb{1} - \mathbf{a}) \circ f(I) + \mathbf{a} \circ Pf(I') \tag{1}$$

Classification loss: Recall that our first goal is to identify class-specific image regions in I and I' such that replacing the regions in I with those in I' increases the predicted probability $g_{c'}(.)$ of class c' for I^*. To avoid a trivial solution where all cells of I are replaced, a sparsity constraint is applied on $\mathbf{a}$ to minimize the number of cell edits (m). Following the greedy approach from [22], we iteratively replace spatial cells in I by repeatedly solving Eq. 2 that maximizes the predicted probability $g_{c'}(\cdot)$ until the model's decision changes.

$$\max_{P,a} g_{c'}((\mathbb{1} - \mathbf{a}) \circ f(I) + \mathbf{a} \circ Pf(I')) \text{ with } ||\mathbf{a}||_1 = 1 \text{ and } a_i \in \{0, 1\} \tag{2}$$

We evaluate $g_{c'}$ for each of the h^2w^2 permutations constructed by replacing a single cell in $f(I)$ with an arbitrary cell in $f(I')$. The computational complexity is $2 \cdot C_f + mh^2w^2 \cdot C_g$, where C_f and C_g denote the cost of f and g respectively.

Eq. 2 does not guarantee that the replaced cells are semantically similar. For example, in the task of bird classification, the counterfactual could replace the wing in I with head in I' (e.g., Fig. 1-b) leading to nonsensical explanations. We address this problem via a semantic consistency constraint, described next.

3.2 Counterfactuals with a semantic consistency constraint

Consider an embedding model $u : \mathcal{I} \rightarrow \mathbb{R}^{hw \times d'}$ that brings together spatial cells belonging to the same semantic parts and separates dissimilar cells. Let $u(I)_i$ denote the feature of the i-th cell in I. We estimate the likelihood that cell i of I semantically matches with cell j of I' by:

$$\mathcal{L}_s(u(I)_i, u(I')_j) = \frac{\exp(u(I)_i \cdot u(I')_j / \tau)}{\sum_{j' \in u(I')} \exp(u(I)_i \cdot u(I')_{j'} / \tau)}, \tag{3}$$

where τ is a temperature hyper-parameter that relaxes the dot product. Eq. 3 estimates a probability distribution of a given query cell i over all distractor cells j' using a non-parametric softmax function and indicates what distractor cells are most likely to contain semantically similar regions as the query cell i. Like the classification loss (Eq. 2), we compute the semantic loss for all h^2w^2 cell permutations. Thus, the complexity is $2 \cdot C_u + h^2w^2 \cdot C_{\text{dot}}$, where C_u, C_{dot} denote the cost of the auxiliary model u and the dot-product operation respectively. Empirically, we observe that dot-products are very fast to compute and the semantic loss adds a tiny overhead to the overall computation time. Note that unlike the classification loss which is computed for each edit, $\mathcal{L}_s$ is computed only once in practice, i.e., the cost gets amortized for multiple edits.

Total loss: We combine both losses to find the single best cell edit:

$$\max_{P,a} \; \log \underbrace{g_{c'}((\mathbb{1} - \mathbf{a}) \circ f(I) + \mathbf{a} \circ Pf(I'))}_{\text{Classification loss } \mathcal{L}_c} + \lambda \cdot \log \underbrace{\mathcal{L}_s(\mathbf{a}^T u(I), \mathbf{a}^T Pu(I'))}_{\text{Semantic consistency loss } \mathcal{L}_s} \tag{4}$$
$$\text{with } P \in \mathbb{R}^{hw \times hw}, \; ||\mathbf{a}||_1 = 1 \text{ and } a_i \in \{0, 1\}, \text{ and } \lambda \text{ balances } \mathcal{L}_c \text{ and } \mathcal{L}_s.$$

We reiterate that $\mathcal{L}_c$ optimizes to find class-specific regions while $\mathcal{L}_s$ ensures that these regions semantically match. We also stress that our explanations are faithful with respect to the underlying deep neural network, since, the proposed auxiliary model, irrespective of the value of λ, only acts as a regularizer and does not affect the class predictions of the transformed images.

Choice of auxiliary model: An obvious choice is to use the spatial feature extractor f as the auxiliary model u. We empirically found that since f is optimized for an object classification task, it results in an embedding space that often separates instances of similar semantic parts and is thus unfit to model region similarity. We found that self-supervised models are more appropriate as auxiliary models for two reasons: a) they eliminate the need for part location information, b) several recent studies [14,52,54] showed that self-supervised models based on contrastive learning [16,23,60] or clustering [7,12,13,53] learn richer representations that capture the semantic similarity between local image regions as opposed to task-related similarity in a supervised setup. Such representations have been valuable for tasks such as semantic segment retrieval [52]. Thus, the resulting embedding space inherently brings together spatial cells belonging to the same semantic parts and separates dissimilar cells (see Table 4).

3.3 Using multiple distractor images through a semantic constraint

Recall, the method uses spatial cells from $f(I')$ to iteratively construct $f(I^*)$. Thus, the quality of the counterfactual is sensitive to the chosen distractor image I'. Having to select regions from a single distractor image can limit the variety of discriminative parts to choose from due to factors like pose, scale, and occlusions. We address this limitation by leveraging multiple distractor images from class c'. In this way, we expand our search space in Eq. 4, allowing us to find highly discriminative regions that semantically match, while requiring fewer edits.

However, leveraging (n) multiple distractor images efficiently is not straightforward as it poses a significant computational overhead. This is because, in this new setup, for each edit we can pick any of $n \times hw$ cells from the n distractor images. This makes the spatial cell matrix of the distractor images of shape $nhw \times d$, the matrix P $hw \times nhw$, and $\mathbf{a} \in \mathbb{R}^{hw}$. $\mathcal{L}_c$ (Eq. 2) with a single distractor image is already expensive to evaluate due to: (i) its quadratic dependence on hw making the cell edits memory intensive and, (ii) the relatively high cost of evaluating g, involving at least one fully-connected plus zero or more conv layers. This computation gets amplified by a factor n with multiple distractor images.

On the other hand, $\mathcal{L}_s$ (Eq. 3) is computationally efficient as: (i) it does not involve replacing cells and (ii) the dot-product is inexpensive to evaluate. Thus, we first compute $\mathcal{L}_s$ (Eq. 3) to select the top-k% cell permutations with the lowest loss, excluding the ones that are likely to replace semantically dissimilar cells. Next, we compute $\mathcal{L}_c$ (Eq. 2) only on these selected top-k% permutations. With this simple trick, we get a significant overall speedup by a factor k (detailed analysis in suppl.). Thus, our overall framework leverages richer information, produces semantically consistent counterfactuals, and is about an order of magnitude faster than [22]. Note that the multi-distractor setup can be extended to [22] but not to SCOUT [59], as the latter was designed for image pairs.

4 Experiments

4.1 Implementation details and evaluation setup

Implementation details: We evaluate our approach on top of two backbones – VGG-16 [46] for fair comparison with [22] and ResNet-50 [24] for generalizability. As mentioned in Sec. 3.1, we split both networks into components f and g after the final down-sampling layer `max_pooling2d_5` in VGG-16 and at `conv5_1` in ResNet-50. The input images are of size 224×224 pixels and the output features of f have spatial dimensions 7×7. We examine counterfactual examples for query-distractor class pairs obtained via the confusion matrix – for a query class c, we select the distractor class c' as the class with which images from c are most often confused. This procedure differs from the approach in [22] which uses attribute annotations to select the classes c, c'. Our setup is more generic as it does not use extra annotations. Distractor images are picked randomly from c'.
Auxiliary model: We adopt the pre-trained ResNet-50 [24] model from DeepCluster [13] to measure the semantic similarity of the spatial cells. We remove

the final pooling layer and apply up- or down-sampling to match the 7×7 spatial dimensions of features from f. As in [13], we use $\tau = 0.1$ in the non-parametric softmax (Eq. 3). The weight $\lambda = 0.4$ (Eq. 4) is found through grid search. We set $k = 10$ and select top-10% most similar cell pairs to pre-filter.

Evaluation metrics: We follow the evaluation procedure from [22] and report the following metrics using keypoint part annotations.

- **Near-KP:** measures if the image regions contain object keypoints (KP). This is a proxy for how often we select discriminative cells, i.e., spatial cells that can explain the class differences.
- **Same-KP:** measures how often we select the same keypoints in the query and distractor image, thus measures semantic consistency of counterfactuals.
- **#Edits:** the average number of edits until the classification model predicts the distractor class c' on the transformed image I^*.

Table 1: **Datasets overview.**

Dataset	Statistics			Top-1	
	#Class	#Train	#Val	VGG-16	Res-50
CUB	200	5,994	5,794	81.5	82.0
iNat. (Birds)	1,486	414 k	14,860	78.6	78.8
Stanf. Dogs	120	12 k	8,580	86.7	88.4

Datasets: We evaluate the counterfactuals on three datasets for fine-grained image classification (see Table 1). The CUB dataset [58] consists of images of 200 bird classes. All images are annotated with keypoint locations of 15 bird parts. The iNaturalist-2021 birds dataset [55] contains 1,486 bird classes and more challenging scenes compared to CUB, but lacks keypoint annotations. So we hired raters to annotate bird keypoint locations for 2,060 random val images from iNaturalist-2021 birds and evaluate on this subset. Stanford Dogs [28] contains images of dogs annotated with keypoint locations [10] of 24 parts. The explanations are computed on the validation splits of these datasets.

4.2 State-of-the-art comparison

Table 2 compares our method to other competing methods. We report the results for both (i) the **single edit** found by solving Eq. 4 once and (ii) **all edits** found by repeatedly solving Eq. 4 until the model's decision changes. Our results are directly comparable with [22]. By contrast, SCOUT [59] returns heatmaps that require post-processing. We follow the post-processing from [59] where from the heatmaps, select those regions in I and I' that match the area of a single cell edit to compute the metrics. From Table 2, we observe that our method consistently outperforms prior works across all metrics and datasets. As an example, consider the **all edits** rows for the CUB dataset in Table 2a. The Near-KP metric improved by **13.9**% over [22], indicating that our explanations select more discriminative image regions. More importantly, the Same-KP metric improved by **27**% compared to [22], demonstrating that our explanations are significantly more semantically consistent. The average number of edits have also reduced from 5.5 in [22] to **3.9**, meaning that our explanations require fewer changes to I and are thus sparser, which is a desirable property of counterfactuals [35,57]. Similar performance trends hold on the other two datasets and architectures (Table 2b) indicating the generalizability of the proposed approach. Figure 3 shows

Table 2: **State-of-the-art comparison** against our full proposed pipeline.

(a) Comparison of visual counterfactuals using a VGG-16 model.

	Method	CUB-200-2011			INaturalist-2021 Birds			Stanford Dogs Extra		
		Near-KP	Same-KP	# Edits	Near-KP	Same-KP	# Edits	Near-KP	Same-KP	# Edits
Single Edit	SCOUT [59]	68.1	18.1	-	74.3	23.1	-	41.7	5.5	-
	Goyal *et al.* [22]	67.8	17.2	-	78.3	29.4	-	42.6	6.8	-
	Ours	**73.5**	**39.6**	-	**83.6**	**51.0**	-	**49.8**	**23.5**	-
All Edits	Goyal *et al.* [22]	54.6	8.3	5.5	55.2	11.5	5.5	35.7	3.7	**6.3**
	Ours	**68.5**	**35.3**	**3.9**	**70.4**	**36.9**	**4.3**	**37.5**	**16.4**	6.6

(b) Comparison of visual counterfactuals using a ResNet-50 model.

	Method	CUB-200-2011			INaturalist-2021 Birds			Stanford Dogs Extra		
		Near-KP	Same-KP	# Edits	Near-KP	Same-KP	# Edits	Near-KP	Same-KP	# Edits
Single Edit	SCOUT [59]	43.0	4.4	-	53.9	8.8	-	35.3	3.1	-
	Goyal *et al.* [22]	61.4	11.5	-	70.5	17.1	-	42.7	6.4	-
	Ours	**71.7**	**36.1**	-	**79.2**	**33.3**	-	**51.2**	**22.6**	-
All Edits	Goyal *et al.* [22]	50.9	6.8	3.6	56.3	10.4	3.3	34.9	3.6	**4.3**
	Ours	**60.3**	**30.2**	**3.2**	**70.9**	**32.1**	**2.6**	**37.2**	**16.7**	4.8

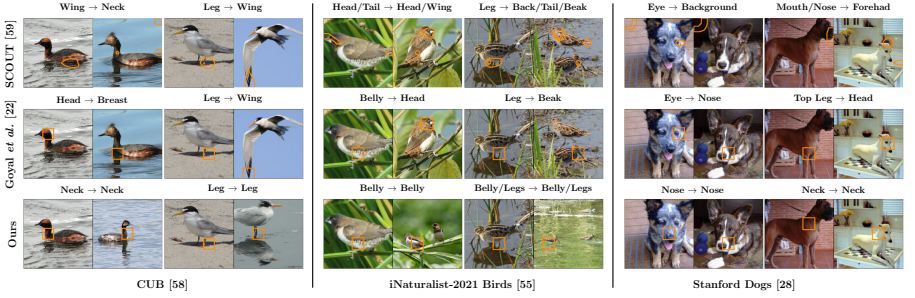

Fig. 3: **State-of-the-art comparison** of counterfactual explanations (Single Edit - VGG-16). Part labels are included only for better visualization. Image credit: [1]

a few qualitative examples where we note that our method consistently identifies semantically matched and class-specific image regions, while explanations from [22] and [59] often select regions belonging to different parts.

4.3 Ablation studies

We now study the different design choices of our framework with [22] as our baseline and use a VGG-16 model for consistent evaluation on CUB.

Analysis of different components. Table 3 reports different variants as we add or remove the following components: semantic loss (Sec. 3.2), multiple distractor images (Sec. 3.3), and pre-filtering cells (Sec. 3.3). Our baseline [22] (row 1) establishes a performance limit for the Near-KP and number of edits under the single-distractor setup as the image regions are selected solely based on the predicted class probabilities $g_{c'}(\cdot)$. First, we observe that the semantic loss improves the semantic meaningfulness of the replacements (row 2), i.e., the Same-KP metric increases by **13.7%**. However, the Near-KP slightly decreases by 2.5% and the number of edits increases by 1.3. This may be due to the fact that row 2 considers both the class probabilities $g_{c'}$ and semantic consistency, thereby potentially favoring semantically similar cells over dissimilar cells that

Table 3: **Effect of different components of our method:** Row 1 is our baseline from [22]. Our method (row 5) uses multiple distractor images combined pre-filtering irrelevant cells and semantic consistency loss. Time measured on a single V-100 GPU.

Row #	Semantic Loss	Multi Distractor	Filters Cells	Near-KP	Same-KP	Time (s)	#Edits
1	✗	✗	✗	54.6	8.3	0.81	5.5
2	✓	✗	✗	52.1 (−2.5)	22.0 (+13.7)	1.02	6.8
3	✗	✓	✗	65.6 (+11.0)	13.8 (+5.5)	9.98	3.5
4	✓	✓	✗	69.2 (+14.6)	36.0 (+23.7)	10.82	3.8
5	✓	✓	✓	68.5 (+13.9)	35.3 (+23.0)	1.15	3.9

yield a larger increase in $g_{c'}$. Second, from rows 1 and 3, we find that allowing multiple distractor images enlarges the search space when solving Eq. 4, resulting in better solutions that are more discriminative (Near-KP ↑), more semantically consistent (Same-KP ↑) and sparser (fewer edits). Combining the semantic loss with multiple distractor images (row 4) further boosts the metrics. However, using multiple distractor images comes at a significant increase in runtime (almost by 10X). We address this by filtering out semantically dissimilar cell pairs. Indeed, comparing rows 4 and 5, we note that the runtime improves significantly while maintaining the performance. Putting everything together, our method outperforms [22] across all metrics (row 1 vs. row 5) and generates explanations that are sparser, more discriminative, and more semantically consistent.

Auxiliary model: Recall from Sec. 3.2 that representations from self-supervised models efficiently capture richer semantic similarity between local image regions compared to those from supervised models. We empirically verify this by using different pre-training tasks to instantiate the auxiliary model: (i) supervised pre-training with class labels, (ii) self-supervised (SSL) pre-training [12,13,23] with no labels, and (iii) supervised parts detection with keypoint annotations. We train the parts detector to predict keypoint presence in the $h \times w$ spatial cell matrix using keypoint annotations. We stress that the parts detector is used only as an *upperbound* as it uses part ground-truth to model the semantic constraint.

We evaluate each auxiliary model by: (i) measuring the Same-KP metric to study if this model improves the semantic matching, and (ii) measuring clustering accuracy to capture the extent of semantic part disentanglement. To measure the clustering accuracy, we first cluster the d-dimensional cells in a 7×7 spatial matrix from $u(\cdot)$ of all images via K-Means and assign each spatial cell to a cluster. Then, we apply majority voting and associate each cluster with a semantic part using the keypoint annotations. The clustering accuracy measures how often the cells contain the assigned part. From Table 4, we observe that better part disentanglement (high clustering accuracy) correlates with improved semantic matching in the counterfactuals (high Same-KP). Thus, embeddings that disentangle parts are better suited for the semantic consistency constraint via the non-parametric softmax in Eq. 3. The CUB classifier fails to model our constraint because it distinguishes between different types of beaks, wings, etc.,

Table 4: **Comparison of auxiliary models on CUB:** We study the Same-KP metric of the counterfactuals (single distractor) and whether the aux. features can be clustered into parts. †Parts detector establishes an upperbound as it uses parts ground-truth.

Auxiliary Model	**Annotations**	**Counterfactuals**	**Clustering** (K-Means Acc.)		
		(Same-KP)	K=15	K=50	K=250
CUB Classifier	Class labels	10.1	18.0	19.3	21.6
IN-1k Classifier	Class labels	19.3	42.0	49.5	57.1
IN-1k MoCo [23]	None (SSL)	18.1	33.8	44.1	52.2
IN-1k SWAV [13]	None (SSL)	22.1	45.3	54.2	62.6
IN-1k DeepCluster [12]	None (SSL)	22.0	45.3	54.9	63.5
CUB Parts Detector†	Keypoints	22.2	46.0	59.2	75.4

to optimize for the classification task (Same-KP drops by 12.1% vs. the upperbound). Differently, the SSL features are more generic, making them suitable for our method (Same-KP using DeepCluster drops only 0.2% vs. the upperbound).

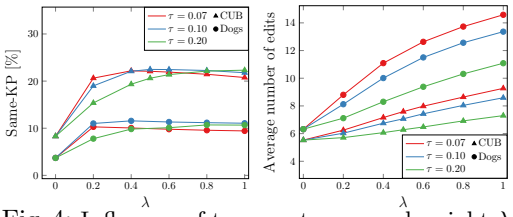

Fig. 4: Influence of temperature τ and weight λ.

Influence of τ and λ: We study how the temperature τ in Eq. 3 and the weight λ parameter in Eq. 4 influence different metrics. Recall that high values of λ favor the semantic loss over the classification loss. Selecting semantically similar cells over dissimilar ones directly improves the Same-KP metric (Fig. 4 (left)), but that comes at a cost of an increased number of edits until the model's decision changes (Fig. 4 (right)). We observe that $\lambda = 0.4$ is a saturation point, after which the Same-KP metric does not notably change. Further, lower values of τ sharpen the softmax distribution making it closer to one-hot, while higher τ yield a distribution closer to a uniform. This has an effect on the number of edits, as a sharper distribution is more selective. We found that for a fixed $\lambda = 0.4$, $\tau = 0.1$ as in [13] is a sweet spot between good Same-KP performance and a small increase in the number of edits. We verified values via 5-fold cross-validation across multiple datasets.

4.4 Online evaluation through machine teaching

To further demonstrate the utility of high-quality visual counterfactuals, we setup a machine teaching experiment, where humans learn to discern between bird species with the help of counterfactual explanations. Through the experiment detailed below, we verify our hypothesis that humans perform better at this task with more informative and accurate counterfactual explanations.

Study setup: We follow the setup from [59], but differ in two crucial ways: (i) ours is a larger study on 155 query-distractor class pairs, while [59] was done only on one class pair; (ii) we obfuscate the bird class names and replace them with "class A" and "class B". We do this because some class names contain identifiable descriptions (e.g., *Red Headed Woodpecker*) without needing visual

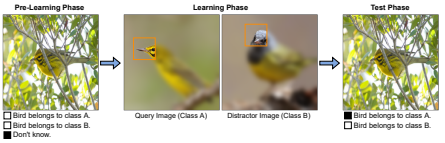

Fig. 5: Machine teaching task phases.

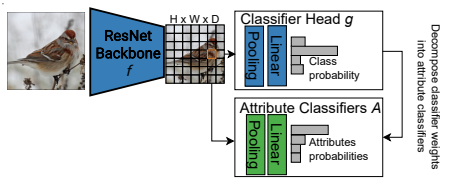

Fig. 6: Attribute-based decomposition.

cues. The study comprises three phases (simplified visualization in Fig. 5). The **pre-learning phase** gives AMT raters 10 test image examples of 2 bird classes. The raters need to choose one of three options: 'Bird belongs to class A', 'Bird belongs to class B,' or 'Don't know'. The purpose of this stage is for the raters to get familiarized with the user interface, and as in [59] all raters chose 'Don't know' for each example in this stage. Next, during the **learning phase**, we show counterfactual explanations of 10 train image pairs where the query image belongs to class A and the distractor image to class B. We highlight the image content from the counterfactual region, with all other content being blurred (Fig. 5). This ensures that the humans do not perform the classification task based on any other visual cues except the ones identified by a given counterfactual method. Finally, the **test phase** presents to raters 10 test image pairs (same as in the pre-learning stage), and asks to classify them into either class A or B. This time, the option 'Don't know' is not provided. Once the task is done, a different set of bird class pair is selected, and the three stages are repeated.

Task details: We hired 25 AMT raters, use images from CUB, and compare counterfactuals produced from our method with two baselines: [22] and [59]. For all three methods, we mine query-distractor classes via the approach mentioned in Sec. 4.1, resulting in 155 unique binary classification tasks. The learning phase visualizes the counterfactual generated from the first edit. To ensure a fair comparison across all methods, we do not use multiple distractor images for generating counterfactuals, use the exact same set of images across all the compared methods, and use the same backbone (VGG-16 [47]) throughout. This controlled setup ensures that any difference in the human study performance can be only due to the underlying counterfactual method. We report results under two setups, which differ in how we select the image pairs (I, I'): **1. random**: we generate explanations from random images using different methods. This is a fair comparison between all methods. **2. semantically-consistent**:

Table 5: **Machine teaching task.** The learning phase selects random image pairs (†), or pairs that show the largest improvement in terms of being semantically consistent (∗).

Method	**Test Acc.** (%) (Random)†	(Semantically-acc.)∗
SCOUT [59]	77.4	62.8
Goyal *et al.* [22]	76.7	64.3
Ours	**80.5**	**82.1**

Table 6: **Attribute-based counterfactuals.** We evaluate whether the top-1 attributes are discriminative of the classes.

Method	**Test Acc.** (%)
SCOUT [59]	46.7
Goyal *et al.* [22]	67.0
Ours	**74.5**

we study whether semantically consistent explanations lead to better human teaching. Hence, we exaggerate the differences in Same-KP between our method and [22,59] by selecting images where our approach has a higher Same-KP metric. If semantic consistency is important in machine teaching, our approach should do much better than 'random', and the baselines should do worse than 'random'. **Results:** Table 5 shows that the raters perform better when shown explanations from our method under the 'random' setup. Further, the differences in test accuracy are more pronounced (82.1% vs. 64.3%) when the raters were presented with semantically consistent explanations. This result highlights the importance of semantically consistent visual counterfactuals for teaching humans.

5 Towards language-based counterfactual explanations

In this section, we propose a novel method to augment visual counterfactual explanations with natural language via the vocabulary of parts and attributes. Parts and attributes bring notable benefits as they enrich the explanations and make them more interpretable [29]. Through this experiment, we further emphasize the importance of semantically consistent counterfactuals and prove them to be a key ingredient towards generating natural-language-based explanations.

Our proof-of-concept experiment uses a ResNet-50 model, where $f(\cdot)$ computes the $h \times w \times d$ spatial feature output of the last conv layer, and $g(\cdot)$ performs a global average pooling operation followed by a linear classifier. We use the CUB [58] dataset with 15 bird parts, where each part (e.g., beak, wing, belly, etc.) is associated with a keypoint location. Additionally, this dataset contains part-attribute annotations (e.g., hooked beak, striped wing, yellow belly, etc.). We perform our analysis on a subset of 77 subsequently denoised part-attributes. Following [30], denoising is performed by majority voting, e.g., if more than 50% of crows have black wings in the data, then all crows are set to have black wings.

In the first step, given a query I from class c and a distractor I' from c', we construct a counterfactual I^*, following our approach from Sec. 3. For fair comparison with [22,59], we limit to single best cell edits. Next, we identify the part corresponding to this best-edit cell in I. We train a parts detector that predicts the top-3 parts for each cell location in the $h \times w$ spatial grid. Note that if the corresponding cell in I' is not semantically consistent with I, the detected parts will not match, and the attribute explanations will be nonsensical. Finally, we find *the most important* attribute for the best-edit via the procedure below.

Finding the best attribute: We train a part-attribute model A that performs global average pooling followed on the output of $f(.)$ by a linear classifier, thus operating on the same feature space as g. We then use an interpretable basis decomposition [63] to decompose the object classifier weights from $g(\cdot)$ into a weighted sum of part-attribute classifier weights from $A(\cdot)$. A simplified visualization is presented in Fig. 6, see [63] for details. The interpretable basis decomposition yields an importance score s_t for each part-attribute t, and we additionally constrain the part-attributes to only the detected parts in the best-edit cells. E.g., if the detected part is a beak, we only consider the {hooked,

Fig. 7: **Augmenting counterfactuals with part-attributes.** We identify the attribute that is most important for changing the model's decision. Best viewed in color.

long, orange, ...}-beak attribute classifiers. Similarly, we compute an importance score s'_t for the best-edit cell in I'. Finally, we compute the differences of importance scores $s'_t - s_t$, where a positive difference indicates that part-attribute t contributed more to the model's decision being c' compared to c. We select the top-k such part-attributes. Again, note that the difference $s'_t - s_t$ makes sense only if the selected parts are semantically same in I and I' (details in suppl.).

Evaluation: For each class pair (c, c'), we use the available annotations to define part-attributes that belong to class c but not to class c', and vice-versa, as proxy counterfactual ground-truth. Our final explanations are evaluated by measuring how often the top-1 part-attribute, identified via the difference between the estimated importance scores, belongs to the set of ground-truth part-attributes.

Results: Table 6 shows the results using visual counterfactuals from our method and from [22,59]. We observe that our method is significantly better compared to prior work in correctly identifying discriminative part-attributes. Given that all other factors were controlled across the three methods, we argue that this improvement is due to our counterfactuals being semantically consistent. Figure 7 shows the qualitative results. Notice that both the wing's color and pattern are visually distinct in Fig. 7 (left), but the part-attribute explanation points out that the wing's pattern mattered the most to the model while generating the counterfactual. Similarly in Fig. 7 (middle), the part-attribute explanation tells us that the crown color is most important. In both cases, the part-attribute information helps disambiguate the visual explanation. Figure 7 (right) shows a failure case caused by a wrongful prediction from the part-attribute classifiers.

6 Conclusion and future work

We presented a novel framework to generate semantically consistent visual counterfactuals. Our evaluation shows that (i) our counterfactuals consistently match semantically similar and class-specific regions, (ii) our proposed method is computationally efficient, and (iii) our explanations significantly outperform the s-o-t-a. Further, we demonstrated the importance of semantically consistent visual counterfactuals via: (i) a machine teaching task on fine-grained bird recognition, and (ii) an approach to augment our counterfactuals with a human interpretable part and attribute vocabulary. Currently, our method greedily searches for one cell replacement at a time. Relaxing this constraint to explore multiple regions in parallel is a fruitful future research problem. Finally, we only scratched the surface in augmenting visual counterfactuals with attribute information. We hope that our work will spark more interest in this worthy topic by the community.

References

1. Authors: Copyright for Figure 3 images from inaturalist-2021, employed for illustration of research work. iNaturalist people: longhairedlizzy: CC BY-NC 4.0, Volker Heinrich: CC BY-NC 4.0, Lee: CC BY-NC 4.0, Jonny Chung: CC BY-NC 4.0, romanvrbicek: CC BY-NC 4.0, poloyellow23: CC BY-NC 4.0, note = Accessed: 2022-03-02
2. Adadi, A., Berrada, M.: Peeking inside the black-box: a survey on explainable artificial intelligence (xai). IEEE access (2018)
3. Adebayo, J., Gilmer, J., Muelly, M., Goodfellow, I., Hardt, M., Kim, B.: Sanity checks for saliency maps. NeurIPS (2018)
4. Akula, A., Wang, S., Zhu, S.C.: Cocox: Generating conceptual and counterfactual explanations via fault-lines. In: AAAI (2020)
5. Alipour, K., Ray, A., Lin, X., Cogswell, M., Schulze, J.P., Yao, Y., Burachas, G.T.: Improving users' mental model with attention-directed counterfactual edits. Applied AI Letters (2021)
6. Ancona, M., Ceolini, E., Öztireli, C., Gross, M.: Towards better understanding of gradient-based attribution methods for deep neural networks. In: ICLR (2018)
7. Asano, Y., Rupprecht, C., Vedaldi, A.: Self-labelling via simultaneous clustering and representation learning. In: ICLR (2019)
8. Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.R., Samek, W.: On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PloS one (2015)
9. Beck, S.R., Riggs, K.J., Gorniak, S.L.: Relating developments in children's counterfactual thinking and executive functions. Thinking & reasoning (2009)
10. Biggs, B., Boyne, O., Charles, J., Fitzgibbon, A., Cipolla, R.: Who left the dogs out?: 3D animal reconstruction with expectation maximization in the loop. In: ECCV (2020)
11. Buchsbaum, D., Bridgers, S., Skolnick Weisberg, D., Gopnik, A.: The power of possibility: Causal learning, counterfactual reasoning, and pretend play. Philosophical Transactions of the Royal Society B: Biological Sciences (2012)
12. Caron, M., Bojanowski, P., Joulin, A., Douze, M.: Deep clustering for unsupervised learning of visual features. In: ECCV (2018)
13. Caron, M., Misra, I., Mairal, J., Goyal, P., Bojanowski, P., Joulin, A.: Unsupervised learning of visual features by contrasting cluster assignments. In: NeurIPS (2020)
14. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: ICCV (2021)
15. Chang, C.H., Creager, E., Goldenberg, A., Duvenaud, D.: Explaining image classifiers by counterfactual generation. In: ICLR (2018)
16. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: ICML (2020)
17. Dabkowski, P., Gal, Y.: Real time image saliency for black box classifiers. In: NeurIPS (2017)
18. Datta, A., Sen, S., Zick, Y.: Algorithmic transparency via quantitative input influence: Theory and experiments with learning systems. In: IEEE SSP (2016)
19. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
20. Dhurandhar, A., Chen, P.Y., Luss, R., Tu, C.C., Ting, P., Shanmugam, K., Das, P.: Explanations based on the missing: towards contrastive explanations with pertinent negatives. In: NeurIPS (2018)

21. Fong, R.C., Vedaldi, A.: Interpretable explanations of black boxes by meaningful perturbation. In: ICCV (2017)
22. Goyal, Y., Wu, Z., Ernst, J., Batra, D., Parikh, D., Lee, S.: Counterfactual visual explanations. In: ICML (2019)
23. He, K., Fan, H., Wu, Y., Xie, S., Girshick, R.: Momentum contrast for unsupervised visual representation learning. In: CVPR (2020)
24. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
25. Hvilshøj, F., Iosifidis, A., Assent, I.: Ecinn: Efficient counterfactuals from invertible neural networks. In: BMVC (2021)
26. Jacob, P., Zablocki, É., Ben-Younes, H., Chen, M., Pérez, P., Cord, M.: Steex: Steering counterfactual explanations with semantics. arXiv:2111.09094 (2021)
27. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: CVPR (2020)
28. Khosla, A., Jayadevaprakash, N., Yao, B., Fei-Fei, L.: Novel dataset for fine-grained image categorization. In: CVPR Workshop (2011)
29. Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., Viegas, F., et al.: Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In: ICML (2018)
30. Koh, P.W., Nguyen, T., Tang, Y.S., Mussmann, S., Pierson, E., Kim, B., Liang, P.: Concept bottleneck models. In: ICML (2020)
31. Lang, O., Gandelsman, Y., Yarom, M., Wald, Y., Elidan, G., Hassidim, A., Freeman, W.T., Isola, P., Globerson, A., Irani, M., et al.: Explaining in style: Training a gan to explain a classifier in stylespace. In: ICCV (2021)
32. Liu, S., Kailkhura, B., Loveland, D., Han, Y.: Generative counterfactual introspection for explainable deep learning. In: GlobalSIP (2019)
33. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. In: NeurIPS (2017)
34. Markus, A.F., Kors, J.A., Rijnbeek, P.R.: The role of explainability in creating trustworthy artificial intelligence for health care: a comprehensive survey of the terminology, design choices, and evaluation strategies. JBI (2021)
35. Miller, T.: Explanation in artificial intelligence: Insights from the social sciences. Artificial intelligence (2019)
36. Mothilal, R.K., Sharma, A., Tan, C.: Explaining machine learning classifiers through diverse counterfactual explanations. In: ACM FAccT (2020)
37. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. NeurIPS (2019)
38. Petsiuk, V., Das, A., Saenko, K.: Rise: Randomized input sampling for explanation of black-box models. In: BMVC (2018)
39. Poyiadzi, R., Sokol, K., Santos-Rodriguez, R., De Bie, T., Flach, P.: Face: Feasible and actionable counterfactual explanations. In: AAAI/ACM AIES (2020)
40. Rebuffi, S.A., Fong, R., Ji, X., Vedaldi, A.: There and back again: Revisiting back-propagation saliency methods. In: CVPR (2020)
41. Ribeiro, M.T., Singh, S., Guestrin, C.: "why should i trust you?" explaining the predictions of any classifier. In: SIGKDD (2016)
42. Ribeiro, M.T., Singh, S., Guestrin, C.: Anchors: High-precision model-agnostic explanations. In: AAAI (2018)
43. Rodriguez, P., Caccia, M., Lacoste, A., Zamparo, L., Laradji, I., Charlin, L., Vazquez, D.: Beyond trivial counterfactual explanations with diverse valuable explanations. In: ICCV (2021)

44. Selvaraju, R.R., Cogswell, M., Das, A., Vedantam, R., Parikh, D., Batra, D.: Grad-cam: Visual explanations from deep networks via gradient-based localization. In: ICCV (2017)
45. Shrikumar, A., Greenside, P., Shcherbina, A., Kundaje, A.: Not just a black box: Learning important features through propagating activation differences. arXiv:1605.01713 (2016)
46. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: Visualising image classification models and saliency maps. arXiv:1312.6034 (2013)
47. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556 (2014)
48. Singla, S., Pollack, B., Chen, J., Batmanghelich, K.: Explanation by progressive exaggeration. In: ICLR (2019)
49. Singla, S., Pollack, B., Wallace, S., Batmanghelich, K.: Explaining the black-box smoothly-a counterfactual approach. arXiv:2101.04230 (2021)
50. Slack, D., Hilgard, S., Jia, E., Singh, S., Lakkaraju, H.: Fooling lime and shap: Adversarial attacks on post hoc explanation methods. In: AAAI (2020)
51. Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: ICML (2017)
52. Van Gansbeke, W., Vandenhende, S., Georgoulis, S., Gool, L.V.: Revisiting contrastive methods for unsupervised learning of visual representations. NeurIPS (2021)
53. Van Gansbeke, W., Vandenhende, S., Georgoulis, S., Proesmans, M., Van Gool, L.: Scan: Learning to classify images without labels. In: ECCV (2020)
54. Van Gansbeke, W., Vandenhende, S., Georgoulis, S., Van Gool, L.: Unsupervised semantic segmentation by contrasting object mask proposals. In: ICCV (2021)
55. Van Horn, G., Cole, E., Beery, S., Wilber, K., Belongie, S., Mac Aodha, O.: Benchmarking representation learning for natural world image collections. In: CVPR (2021)
56. Verma, S., Dickerson, J., Hines, K.: Counterfactual explanations for machine learning: A review. arXiv:2010.10596 (2020)
57. Wachter, S., Mittelstadt, B., Russell, C.: Counterfactual explanations without opening the black box: automated decisions and the gdpr. Harvard Journal of Law and Technology (2018)
58. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The Caltech-UCSD Birds-200-2011 Dataset. Tech. rep., California Institute of Technology (2011)
59. Wang, P., Vasconcelos, N.: Scout: Self-aware discriminant counterfactual explanations. In: CVPR (2020)
60. Wu, Z., Xiong, Y., Yu, S.X., Lin, D.: Unsupervised feature learning via non-parametric instance discrimination. In: CVPR (2018)
61. Zablocki, É., Ben-Younes, H., Pérez, P., Cord, M.: Explainability of vision-based autonomous driving systems: Review and challenges. arXiv:2101.05307 (2021)
62. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: ECCV (2014)
63. Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., Torralba, A.: Learning deep features for discriminative localization. In: CVPR (2016)
64. Zhou, B., Sun, Y., Bau, D., Torralba, A.: Interpretable basis decomposition for visual explanation. In: ECCV (2018)
65. Zintgraf, L.M., Cohen, T.S., Adel, T., Welling, M.: Visualizing deep neural network decisions: Prediction difference analysis. In: ICLR (2017)

A Implementation details

This section discusses additional implementation details of our experiments in Sec. 4 and Sec. 5.

Data. We use the following publicly available datasets for fine-grained image recognition: CUB [58], iNaturalist-2021 [55], and StanfordDogs [28]. The image classifier implementation follows the typical implementation in PyTorch and uses standard image augmentations (i.e., random resized cropping and random horizontal flipping). We provide additional details for each dataset below:

- **CUB**: The CUB dataset consists of images of 200 bird species annotated with keypoint locations of 15 bird parts, e.g., crown, beak, etc. Some of the keypoint annotations distinguish between the left-right instances of parts: 'left wing' / 'right wing', 'left leg' / 'right leg', and 'left eye' / 'right eye'. We treat these as a single part during the evaluation of the Near-KP and Same-KP metrics, i.e., 'left wing' and 'right wing' as 'wing'.
- **iNaturalist-2021**: The iNaturalist-2021 dataset consists of various supercategories (e.g., plants, insects, birds, etc.), covering 10,000 species in total. The dataset contains a larger number of classes and more complex scenes compared to other fine-grained image recognition datasets. Therefore, the iNaturalist-2021 dataset can be considered as a more challenging testbed for our approach. However, this dataset lacks keypoint annotations. We used the bird supercategory for our quantitative evaluation in Sec. 4, and requested human annotators to provide part keypoint information for 2,060 validation images. The keypoint definitions from the CUB dataset are used. We did not perform a quantitative evaluation of other supercategories, as these are considerably more challenging to annotate. Specifically, we identified the following challenges: (i) some supercategories do not have identifiable parts (e.g., fungi), and (ii) some supercategories are too diverse, and do not support common keypoint definitions across all sub-categories (e.g., plants, mammals, insects, etc.). We do provide qualitative results on several of these supercategories in Sec. C.
- **StanfordDogs**: The StanfordDogs dataset contains images of 120 dog breeds taken from the ImageNet [19] dataset. The keypoint annotations are provided by [10]. Again, we treat left-right instances of parts as the same part, i.e., left ear and right ear as just ear.

Classifier. The training follows the typical VGG-16 and ResNet-50 implementation in PyTorch [37] with 100 epochs. All models use pre-trained ImageNet [19] weights. The training uses stochastic gradient descent with momentum 0.9 and weight decay of 0.0001. We use batches of size 32 for the CUB and Stanford Dogs datasets, and batches of size 256 for the iNaturalist-2021 dataset. The initial learning rate is selected via grid search and decreased by 10 at the 70-th and 90-th percentile of training.

Self-supervised models. We used the publicly available weights that were provided by the authors of the respective works [13,23]. For DeepCluster and SWAV, we adopt the models trained via the multi-crop augmentation from [13]. We pre-trained all models on ImageNet. Different pre-training schemes could be used when considering more specialized domains like medical images.

Parts detector. We trained a parts detector on CUB. The parts detector consists of a ResNet-50 backbone followed by a 1×1 convolutional layer. The input images are 224×224 pixels and the output has spatial dimensions 7×7. We project the ground-truth bird keypoint annotations onto a grid of shape 7×7 and train the parts detector to predict keypoint presence via a multi-class cross-entropy loss. The loss is only applied to cells that contain at least one keypoint. Training follows the classifier implementation but we decrease the number of epochs to 50. The initial learning rate is 0.001. We evaluate the predictions via the mean AP metric (excluding cells that contain no keypoints). The best model obtains a mean AP of 92.3 on the validation set.

B Additional results

We report additional results that complement the ablation studies in the main paper.

Selection of query-distractor classes. The experiments in Sec. 4 examine counterfactual examples for query-distractor classes obtained via the confusion matrix - for a query class c, we select the distractor class c' as the class with which images from c are most often confused. This procedure differs from the approach in [22] which uses attribute annotations to select the most confusing classes. In particular, the authors select c' as the nearest neighbor class of c in terms of the average attribute annotations provided with the dataset. We argue that our setup is more generic as it does not use additional annotations. For completeness, we provide results with both selection procedures in Table S1. We observe that there are no significant differences in the results when adjusting the selection procedure. In practice, the two selection procedures often generate the same query-distractor class pairs. In conclusion, our selection procedure provides a viable and more generic alternative to the method from [22].

Background as a metric. The CUB dataset contains mask annotations that segment the foreground object. Prior work [22] used the object segmentation to measure how often the counterfactuals select cells belonging to the foreground object. Like the Same-KP metric, this *foreground metric* is a proxy for how often the counterfactuals select discriminative cells, i.e., cells that explain the class differences. For completeness, we report the results of the foreground metric in Table S2. Our method outperforms the baseline [22] in terms of the foreground metric, which indicates that we select more discriminative cells in the image. This

Table S1: **Comparison of different methods to select the query-distractor classes.** We select the distractor class as the most confusing class in the confusion matrix, or as the nearest neighbor class in terms of the average attribute annotations. The results are reported for a VGG-16 classifier on CUB.

Method	Selection Procedure	Near KP	Same KP	#Edits
Goyal *et al.* [22]	Confusion Matrix	54.6	8.3	5.5
	Attributes	55.0	8.6	5.4
Ours	Confusion Matrix	68.5	35.3	3.9
	Attributes	68.6	35.6	3.9

Table S2: **Ablation study of the foreground metric.** The results are with VGG-16 on CUB. We compare the baseline [22] against our method.

	Foreground	Near KP	Same KP	#Edits
Goyal *et al.* [22]	94.2	54.6	8.3	5.5
Ours	99.1 (+4.9)	68.5 (+13.9)	35.3 (+23.0)	3.9

observation aligns with our conclusions in the paper based upon the Near-KP metric. Note that the other metrics were discussed in Section 4.3.

Soft versus hard semantic constraint. We have modeled the semantic consistency constraint in a *soft* way (see Eq. 4). That is, we select replacements that balance the increase of $g_{c'}(\cdot)$ with the semantic similarity of the image regions. Alternatively, the constraint could be implemented in a *hard* way. That is, we cluster the auxiliary spatial features first, e.g., K=50, and only replace query cells with distractor cells from the same cluster. Table S3 compares the two mechanisms. The hard constraint selects considerably less discriminative cells (lower Near-KP and more edits) as it's more restrictive of the cells that can be replaced. In contrast, the soft constraint achieves better results, as it balances the $\mathcal{L}_s$ and $\mathcal{L}_c$ losses in Eq. 4.

Table S3: **Comparison of the hard and soft constraint mechanism**. Results are obtained with a VGG-16 classifier on CUB. We use a single distractor image.

Constraint	Near KP	Same KP	#Edits
Hard	40.1	13.3	9.3
Soft	**52.1**	**22.0**	**6.8**

Clustering. We studied the part clustering accuracy via different auxiliary models in Table 4. In this way, we verified whether the spatial feature represen-

tations of the auxiliary models are capable of disentangling parts. In this section, we provide qualitative results of this experiment. Figure S1 visualizes clusters found via a CUB classifier and DeepCluster model. We select several clusters and highlight cells assigned to the same cluster. The DeepCluster features better disentangle parts, i.e., cells assigned to the same cluster refer to the same part.

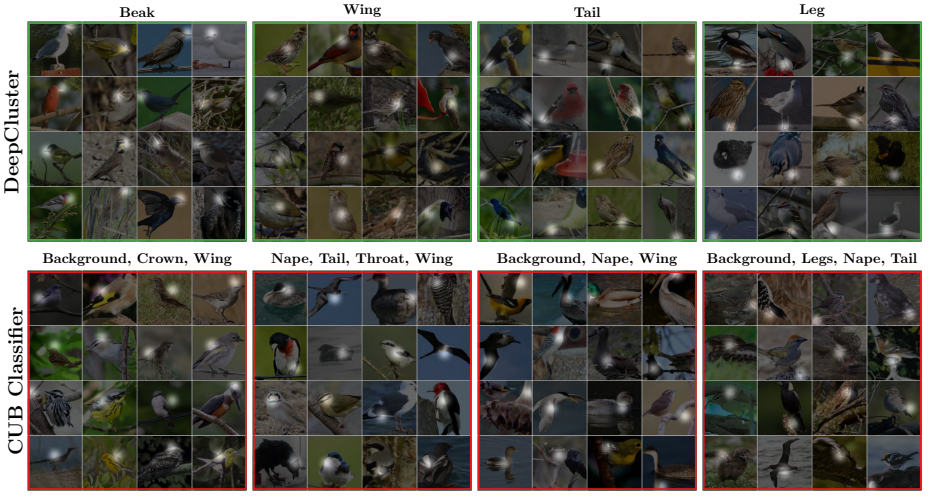

Fig. S1: **Clustering visualizations.** We study part disentanglement when clustering the spatial features obtained with different auxiliary models. We highlight image regions assigned to the same cluster (best viewed in color digitally). We indicate the parts found in each cluster for the purpose of visualization.

Ablation of pre-filtering operation. We study the influence of the pre-filtering step from Sec. 3.3 in Table S4. Namely, we vary k in the selected top-k% permutations to be used in the multi-distractor setup. As it can be observed, our method is very robust to the choice of the value k, so we selected k in order to achieve around x10 speedup over the vanilla multi-distractor approach.

Effect of receptive field. We discuss how the receptive field size of the classification network effects the quality of the counterfactuals. Large receptive fields can lead to poorer localization and reduce the counterfactual's quality. For example, the ResNet-50 models with receptive field size 299 yield lower numbers in Table 2 compared to their VGG-16 counterparts with receptive field size 212. We tried to mitigate this behavior by using features from the earlier `conv5_1` layer instead of `conv5_3` for ResNet and found it improves results. For example, on CUB, this change led to a reduction in the number of edits from 8.0 to 3.2.

Table S4: **Ablation of pre-filtering operation** that selects $k\%$ of permutations via the semantic similarity loss. We study the influence of varying $k\%$.

k%	Near KP	Same KP	#Edits	Time (s)
0.01	61.2	34.1	4.8	0.18
0.05	66.8	34.9	4.2	0.64
0.10	68.5	35.3	3.9	1.15
0.15	68.9	35.5	3.9	1.71
0.20	69.1	35.9	3.8	2.20
1.00 (no-prefiltering)	69.2	36.0	3.8	10.82

In the future, we could further address this issue by using features from earlier layers with better localization. Alternatively, we could explore techniques which control the receptive field size of the network.

Interpretation of Same-KP. Visualization of our counterfactual explanations shows that we consistently identify class-specific and semantically matched parts (see Figure 3). However, the absolute values of the Same-KP metric in Table 2 might still seem low ($< 40\%$). There are two main reasons for this. First, we project the keypoint annotations from the query and distractor images onto the discrete spatial cells during evaluation, associating each cell with a set of keypoints. Now, keypoints lying near the cell boundaries are assigned to only one cell. At the time of evaluation, for such a keypoint, even if a neighboring cell is chosen for replacement, Same-KP metric is penalized. Secondly, a keypoint represents just a near center point of a semantic part, rather than the whole part. Hence, a wing of the bird may actually belong to two adjoining spatial cells, but the keypoint is only in one cell. Thus we also report Near-KP metric which does not suffer form these issues. In conclusion, our counterfactuals are faithful, which is also reflected in our qualitative results.

C Qualitative results

Additional qualitative examples. Figures S2-S3 show counterfactual examples generated with different methods on the CUB [58], iNaturalist-2021 Birds [55] and StanfordDogs [28] datasets. The model is ResNet-50. In each case, we highlight the single best edit. We observe that our counterfactual explanations consistently identify class-specific and semantically consistent image regions. This is opposed to other counterfactual explanations [22,59] which often replace regions of different parts.

Qualitative examples for other iNaturalist-2021 supercategories. The paper only studied the birds supercategory on iNaturalist-2021, as other supercategories are considerably harder to annotate with keypoint information (see discussion in Sec. A). In this subsection, we demonstrate that our method can

be applied to other supercategories too via qualitative results. We train separate image classifiers on the supercategories of 'Mammals', 'Insects', and 'Ray-finned Fishes', and generate counterfactual explanations using our approach. The image classifiers use a ResNet-50 model, and training follows the iNaturalist-2021 implementation detailed in Sec. A. Figure S5 shows the results for the single best edit. Again, we find that our counterfactual explanations highlight class-specific and semantically consistent image regions in the query and distractor images. In conclusion, our method applies to a broad variety of fine-grained image classification tasks.

Visualization of multiple edits. Recall that our method iteratively replaces cells between the query image and distractor image(s) until the model's decision changes. So far, we have only showed visualizations of the first cell edit. Figure S6 shows counterfactual explanations where we visualize all edits until the model's decision changes. For the purpose of visualization, we randomly selected counterfactual explanations that require three cell replacements. In each case, we observe that all edits select class-specific and semantically consistent image regions. In conclusion, our method achieves the desired result, not only for the first edit, but across all edits.

Visualization of failure cases. Figure S7 shows three failure cases, where our counterfactual explanations replace regions referring to different bird parts. The examples were generated for a ResNet-50 classifier trained on the CUB dataset. We make the following observations. First, the failure cases seem to occur for (i) odd looking birds; e.g., the query bird in Example 1 looks quite different from other birds in the CUB dataset, and (ii) for query images where certain bird parts fall outside the image; e.g., the discriminative parts of the query bird in Example 2 and Example 3 fall outside the image. Second, we observe that other methods make similar mistakes. In conclusion, the failure cases seem caused by the challenging nature of the examples, rather than being a pitfall in our method.

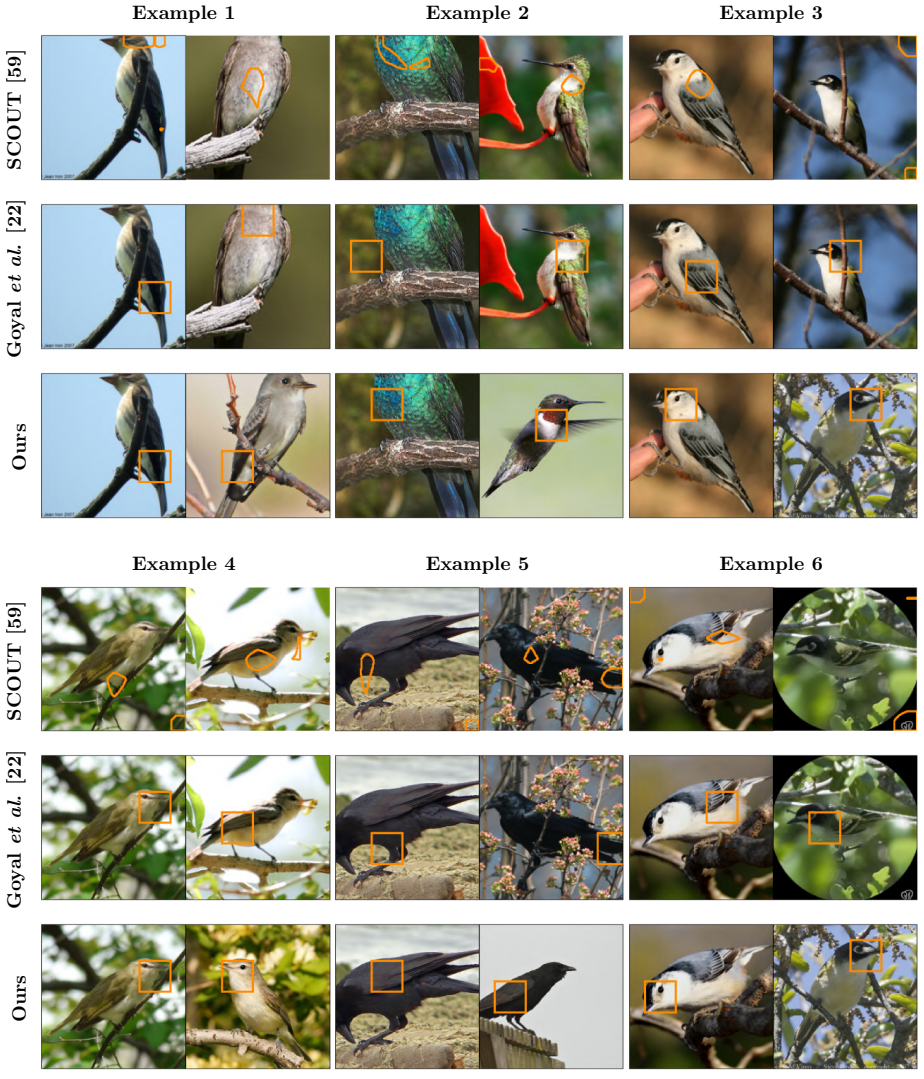

Fig. S2: **Additional qualitative results on CUB [58].** We highlight the best edit in the query image (left) and distractor image (right).

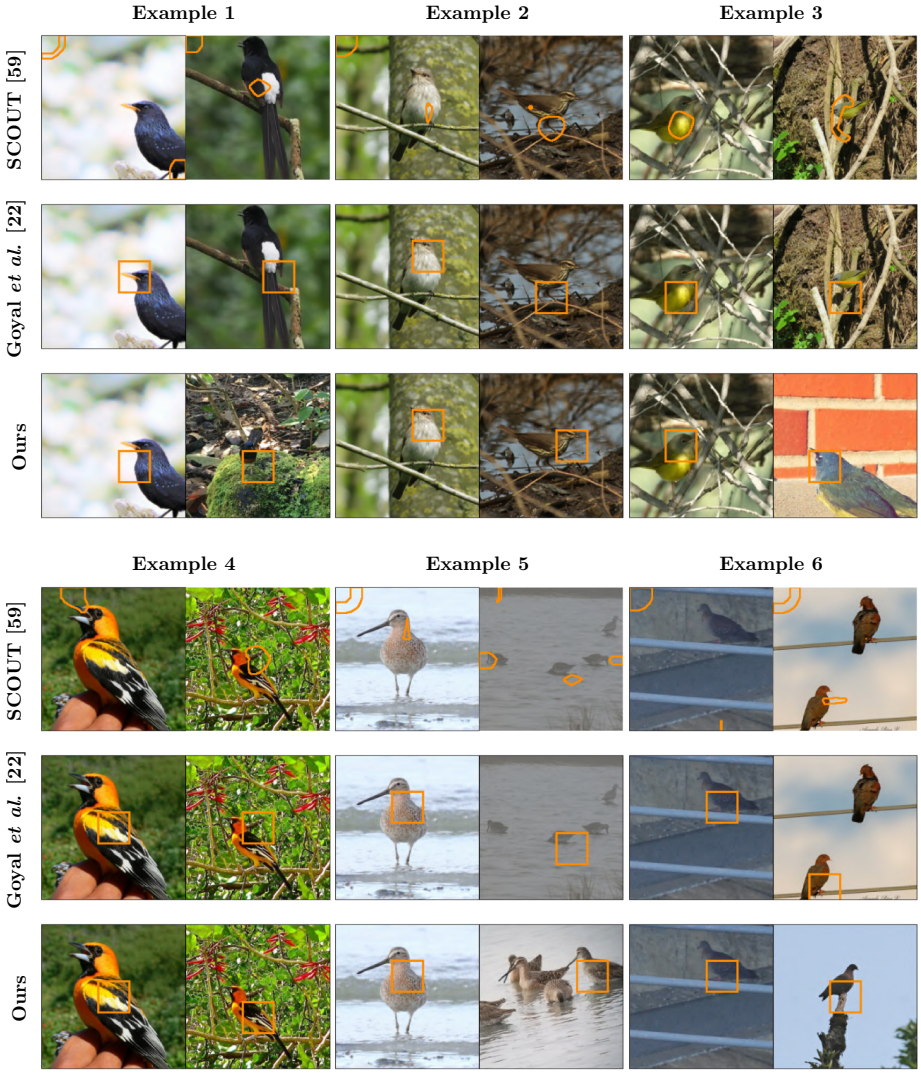

Fig. S3: **Additional qualitative results on iNaturalist-2021 Birds [55].** We highlight the best edit in the query image (left) and distractor image (right).

Fig. S4: **Additional qualitative results on StanfordDogs [28].** We highlight the best edit in the query image (left) and distractor image (right).

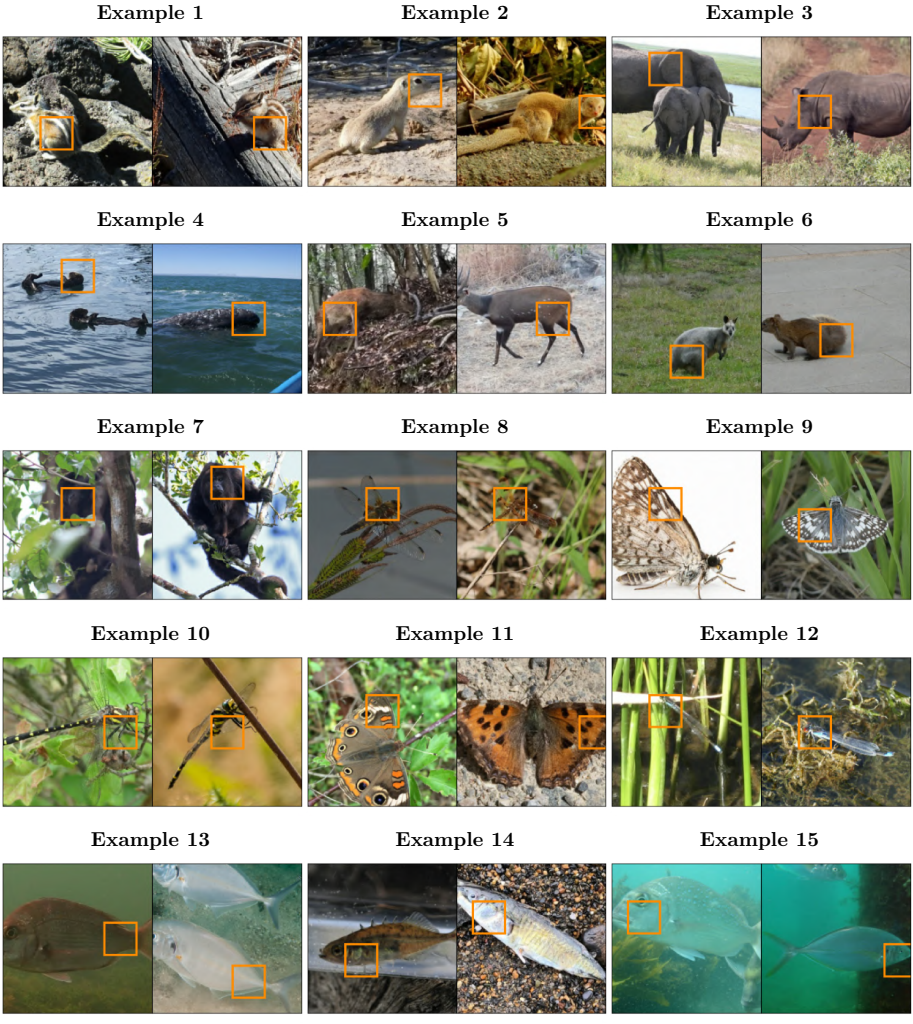

Fig. S5: **Qualitative results on other iNaturalist-2021 supercategories**. We show counterfactual explanations from our method on the following iNaturalist-2021 supercategories: 'Mammals', 'Ray-finned Fishes' and 'Insects'. We highlight the best edit in the query image (left) and distractor image (right).

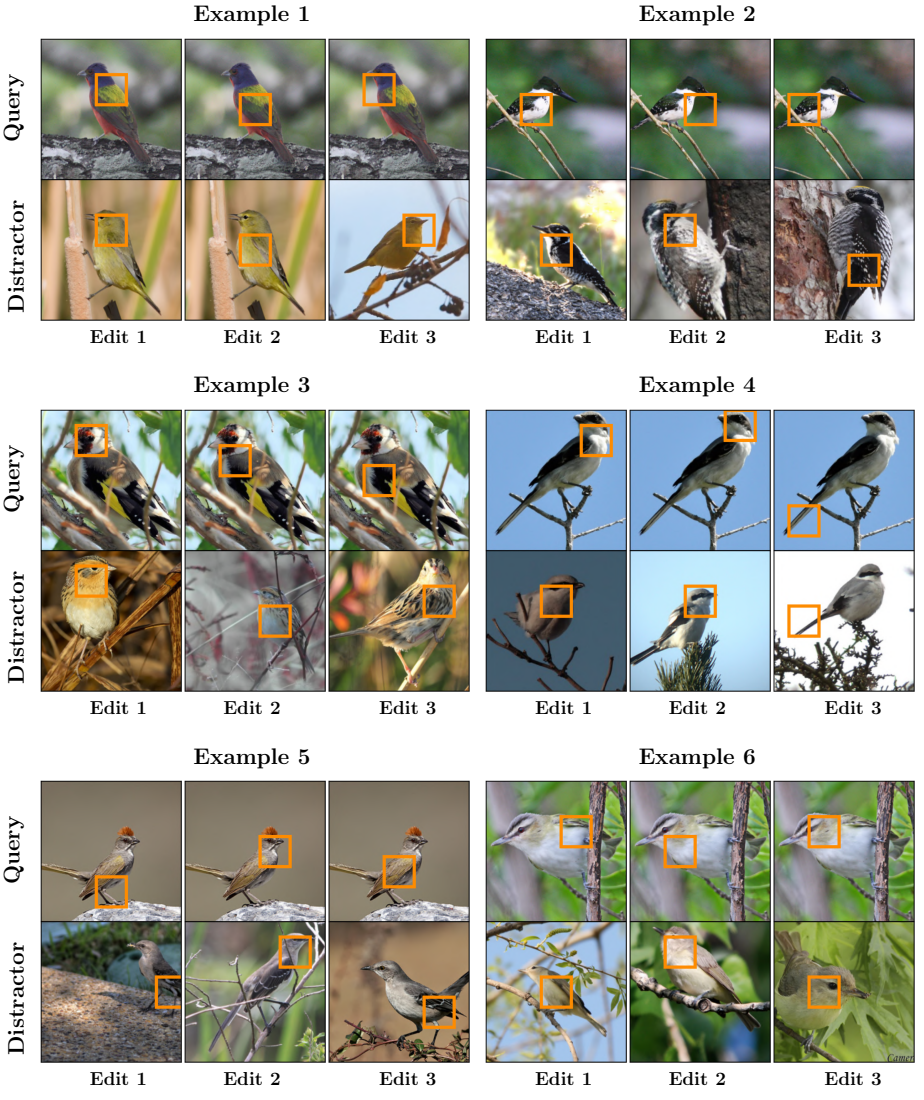

Fig. S6: **Visualization of consecutive edits on CUB.** Our counterfactual explanations iteratively replace single cells until the model's decision changes. The figure highlights these consecutive edits in the query image and distractor image(s). To generate the figure, we select counterfactual explanation that use three edits.

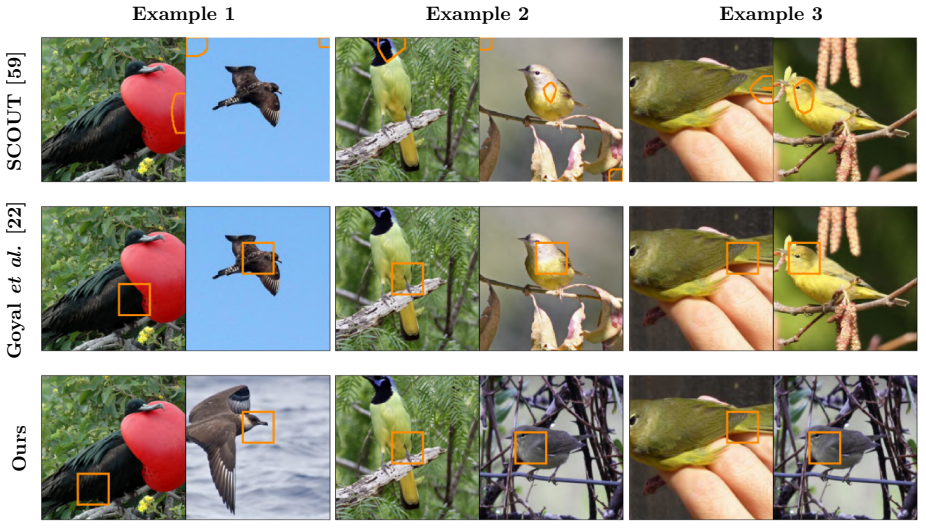

Fig. S7: **Failure cases**. We show some failure cases, where our counterfactual explanations replace regions referring to different parts.

D Computational cost analysis

In this section, we perform a complexity analysis of our approach. Additionally, we include a compute time analysis under the multi-distractor setup.

Computational complexity. We perform a back-of-the-envelope calculation of the number of multiply-add computations (MACs) in our framework. To simplify the analysis, we consider the computational cost of performing a single edit. Recall from Sec. 3 that the computational complexity of the classification loss ($\mathcal{L}_c$ in Eq. 2) and semantic loss ($\mathcal{L}_s$ in Eq. 3) can be summarized as:

$$C_{\mathcal{L}_c} = 2 \cdot C_f + h^2 w^2 \cdot C_g \tag{5}$$

$$C_{\mathcal{L}_s} = 2 \cdot C_u + h^2 w^2 \cdot C_{\text{dot}}. \tag{6}$$

We compute counterfactual explanations using the $7 \times 7 \times 512$ spatial features of the `max_pooling2d_5` layer in VGG-16 [47]. The evaluation of the classification loss $\mathcal{L}_c$ uses 318.8×10^9 MACs ($\approx 2 \cdot 15.4 \times 10^9$ for f + $h^2 w^2 \cdot 1.2 \times 10^8$ for g), while the semantic loss $\mathcal{L}_s$ computation uses 8.2×10^9 MACs ($\approx 2 \cdot 4.1 \times 10^9$ for u + $h^2 w^2 \cdot 2.0 \times 10^3$ for the dot-product in the softmax s). We conclude that the classification loss is expensive to compute due to it's quadratic dependence on the number of cells hw and the relatively high cost of evaluating the decision network $g(\cdot)$. In contrast, the semantic loss does not suffer from it's quadratic term because the dot-product operation is inexpensive to compute. In conclusion, $\mathcal{L}_s$ can be computed more efficiently compared to $\mathcal{L}_c$.

The pre-filtering operation from Sec. 3.3 relies on the fast computation of the semantic similarity loss to realize a speed-up. For example, we select the top-10% most similar cells ($k = 0.1$) according to the semantic loss, and then only consider the classification loss for this subset of cells. This reduces the complexity of $\mathcal{L}_c$ to 59.6×10^9 MACs ($\approx 2 \cdot 15.4 \times 10^9$ for f + $\underline{k} h^2 w^2 \cdot 1.2 \times 10^8$ for g). As a result, the overall computational cost is reduced from 327×10^9 to 67×10^9 MACs, meaning our framework holds a significant speed advantage over methods that compute $\mathcal{L}_c$ exhaustively [22].

This analysis does not consider the memory aspect of computing $\mathcal{L}_c$ and $\mathcal{L}_s$. It's worth noting that the similarity loss also holds an advantage in terms of memory usage compared to the classification loss. Specifically, in order to compute the classification loss for all $h^2 w^2$ permutations, we need to construct all permutations in memory. This involves the allocation of $h^2 w^2$ spatial cell matrices by replacing cells in $f(I)$ with cells from $f(I')$. In contrast, computing the similarity loss does not require to allocate $O(h^2 w^2)$ extra memory as it does not involve replacing cells. Instead, the semantic loss operates directly on the spatial feature matrices of the auxiliary model, i.e., $u(I)$ and $u(I')$.

Compute time analysis in a multi-distractor setup: Figure S8 reports the average computation time per edit (on a single V-100 GPU) as a function of the number of distractor images. We note that our method which includes a pre-filtering operation (Sec. 3.3) scales linearly with the number of distractor images and is about an order of magnitude faster compared to [22].

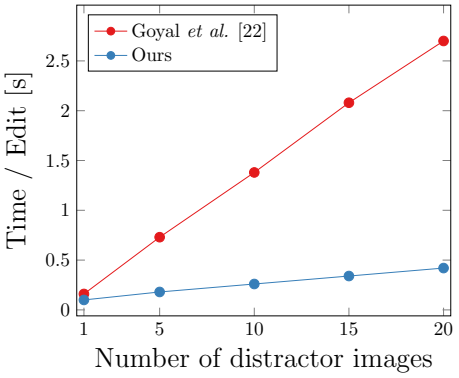

Fig. S8: Time analysis of multi-distractor setup.

E Attributes

E.1 Implementation details

We provide additional implementation details of how we add natural language attribute information to the visual counterfactual explanations in Sec. 5. Recall, the classifier is a ResNet-50 model trained to identify bird species on CUB. The spatial feature extractor f computes the $h \times w \times d$ spatial feature output of the last convolutional layer, and g performs a global average pooling operation followed by a linear classifier.

Parts detector. We reuse the CUB parts detector from Sec. A. The parts predictor is used to select the top-3 parts for the spatial cells that are being replaced in the counterfactual.

Attribute classifiers. We train linear classifiers to predict part-attributes on top of the average-pooled features from $f(\cdot)$. The part-attributes are derived from the attribute annotations used by [30]. Specifically, we only use attributes that refer to parts for which we have keypoint locations. This results in 77 attributes in total. We train linear classifiers to predict the part-attributes via a multi-class cross-entropy loss. The training uses SGD with momentum 0.9 and initial learning rate 0.04. We use batches of size 64 and train for 100 epochs. The learning rate is decayed by 10 after 70 and 90 epochs. We use weight decay 1e-6.

Interpretable basis decomposition. We perform the interpretable basis decomposition as follows. Consider a counterfactual that replaces a cell i in $f(I)$ with a cell i' from $f(I')$. First, we determine the attributes that should be used for the decomposition. To this end, we take the union of detected parts in cell i and i' first, and then select the attributes that are associated with the detected parts, e.g., if one of the parts is 'wing' we select attributes like 'has_wing_color::blue'. We then apply the algorithm from [64] to decompose the weights of the linear

layer in g in terms of the selected attribute classifiers. The decomposition is performed for the query $f(I)$ and counterfactual $f(I^*)$.

Additional examples. Figure 7 shows additional examples, where our method succeeds in adding attribute information to our visual counterfactual explanations. In each case, the returned attribute belongs to class c but not to class c', or vice-versa. Thus, the returned attributes are discriminative of the classes c and c'.

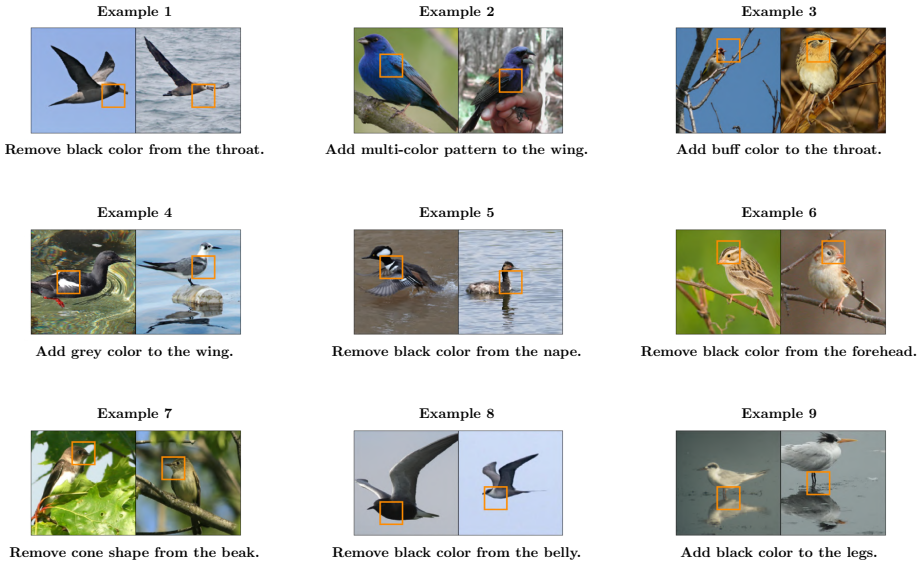

Fig. S9: **Language-based counterfactuals.** We identify the attribute that is most important for chaenging the model's decision.

F Licenses

We include the licenses for images from iNaturalist used in our visualizations.

Table S5: Authors and Creative Commons Copyright notice for images in Figure S3. lauriekoepke: CC BY-NC 4.0, walterflocke: CC BY-NC 4.0, leo_v: CC BY-NC 4.0, Herbert Herbinia: CC BY-NC 4.0, leptim: CC BY-NC 4.0, leptim: CC BY-NC 4.0, nanorca13: CC BY-NC 4.0, toucan55: CC BY-NC 4.0, Jim Brighton: CC BY-NC 4.0, wildmouse3: CC BY-NC 4.0, Will Richardson: CC BY-NC 4.0, Amado: CC BY-NC 4.0, Esteban Munguia: CC BY-NC 4.0, jamesbeat: CC BY-NC 4.0, jamesbeat: CC BY-NC 4.0, N. Mahathi: CC BY-NC 4.0, ddun: CC BY-NC 4.0, John G. Phillips: CC BY-NC 4.0.

Table S6: Author and Creative Commons Copyright notice for images in Figure S5. Daniel George: CC BY-NC 4.0, thehaplesshiker: CC BY-NC 4.0, tiyumq: CC BY-NC 4.0, Rohit Chakravarty: CC BY-NC 4.0, ungerlord: CC BY-NC 4.0, dushenkov: CC BY-NC 4.0, jenhenlo: CC BY-NC 4.0, amniotasmarinos6: CC BY-NC 4.0, Lawrence Troup: CC BY-NC 4.0, Andrew Deacon: CC BY-NC 4.0, Torbjorn von Strokirch: CC BY-NC 4.0, 金翼白眉: CC BY-NC 4.0, Andrés Matos: CC BY-NC 4.0, Christoph Moning: CC BY-NC 4.0, asandlermd: CC BY-NC 4.0, Eric Giles: CC BY-NC 4.0, Paul Cools: CC BY-NC 4.0, Laura Kimberly: CC BY-NC 4.0, sterling: CC BY 4.0, elisabraz: CC BY-NC 4.0, Tom Warnert: CC BY-NC 4.0, Don Loarie: CC BY 4.0, Marlo Perdicas: CC BY 4.0, luca tringali: CC BY-NC 4.0, pmmullins: CC BY-NC 4.0, Erik Schlogl: CC BY-NC 4.0, sea-kangaroo: CC BY-NC-ND 4.0, Jason Grant: CC BY-NC 4.0, Donald Hobern: CC BY 4.0, Richard Ling: CC BY-NC-ND 4.0, BeachBumAgg: CC BY-NC 4.0, Tony Strazzari: CC BY-NC 4.0.