Choose your Data Wisely: A Framework for Semantic Counterfactuals

Author Name
Affiliation
email@example.com

A Additional Experiments: CLEVR-Hans3

The experimental objective most aligned with counterfactual explanations is uncovering the biases of a trained classifier. To test our system on this task, we start our experimental procedure with very controlled datasets and move toward more real-world use cases. Our first experiment was to cross-check the validity of our conceptual counterfactuals system with an existing black-box counterfactual method that performs only on pixel-edits, namely FACE: Feasible and Actionable Counterfactual Explanations, on a controlled dataset. For this, we chose CLEVR-Hans3[1], which generates images of colored 3D geometrical object sets with predetermined biases. The goal was to see if our system would be able to detect the foreknown bias of class A (in the training set, all images contain a "gray cube").

Our objective for experimenting with this dataset was twofold. Firstly, we wanted to evaluate the suggested edits of our knowledge-inclusive system with the suggested pixel-level edits of the FACE algorithm. Secondly, we needed to test the theory that by accumulating local explanations, we could produce global explanations that highlight the foreknown biases.

Setting

CLEVR-Hans3 is a dataset that contains images of colored 3D geometrical object sets, which are further split into three classes. For each image, information is provided regarding all present objects concerning their shape (Sphere, Cube, $\mathsf{Cylinder}$), size (Large, Small), material ($\mathsf{Metallic}$, Rubber), and color (Blue, Yellow, Brown, Grey, Green, Purple, Cyan, Red). The three classes are separated by the objects the images contain, according to their respective rules: A) a Large Cube and a Large Cylinder, B) a Small Metal Cube and a Small Sphere, and C) a Large Blue Sphere and a Small Yellow Sphere. Furthermore, the first two classes are confounded in the training set with an intentional bias. For class A, in the training set, the Large Cube is always Grey while in the test set the color of the Large Cube is random. For class B, the material of the Small Sphere is Metal in the training set, while in the test set the material of the Small Sphere is random. This means that we expect classifiers trained on the training set, to be biased toward the confounding factor. For our classifier, we trained a resnet34 model which achieved 99% accuracy on the (confounded) training set, while for the test set the per-class F1 scores were class A: 0.27, class B: 0.54, class C: 0.92. As expected, its performance on the confounded classes was poor.

We created two explanation datasets, one including exclusively training set images so that we can compare our system to FACE (which is intended to run only on the training set images), and one including test set images in an attempt to detect the biases that were acquired in training. As a set of concept names CN, we defined a concept for every combination of shape, size, material, and color (including the absence of any of the above), leading to $|\mathsf{CN}| = 4 \times 3 \times 3 \times 9 = 324$. As a TBox, we added an inclusion axiom from each concept in CN to any other concept with the same description where one element is missing. For example $\mathsf{GrayCube} \sqsubseteq \mathsf{Gray}$ and $\mathsf{GrayCube} \sqsubseteq \mathsf{Cube}$. This way, we assigned sets of concepts to each element in the dataset, based on the descriptions provided by the creators of the dataset in the corresponding json files.

Local Counterfactuals

In fig.1 we show local counterfactual explanations generated for three randomly selected images (first column), which were classified in class B (Small Metal Cube and Small Sphere - where the Small Sphere is always Metal in the train set) and with the target class being class A (Large Cube, Large Cylinder, where the Large Cube is always Grey in the train set). The second column shows the suggestions of the FACE algorithm and the third column shows the suggestions of our algorithm. At first glance, neither results are very intuitive, and we argue that the form of the explanations (sequence of samples from the training set) is the reason. A more thorough observation reveals that our approach tends to keep the number of objects in an image constant, which is due to the high cost of adding and deleting concepts rather than replacing them, while FACE, which relies on the distribution of the dataset, operating on pixel-level and having no knowledge of the objects depicted, tends to transition to images containing a greater number of objects.

Global Counterfactuals

For our first global counterfactual explanation, we take the images classified as class B from CLEVR-Hans3 test-set and examine the edits that our system performs on those images, for them to be classified in class A. We consider a large num-

[1]https://github.com/ml-research/CLEVR-Hans

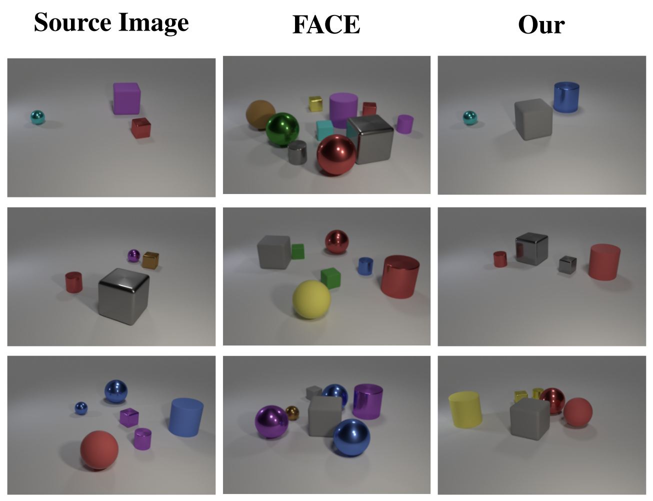


Figure 1: Counterfactuals for 3 images (first column) which classified in class B with target class A, using FACE (second column) and our proposed method (third column)

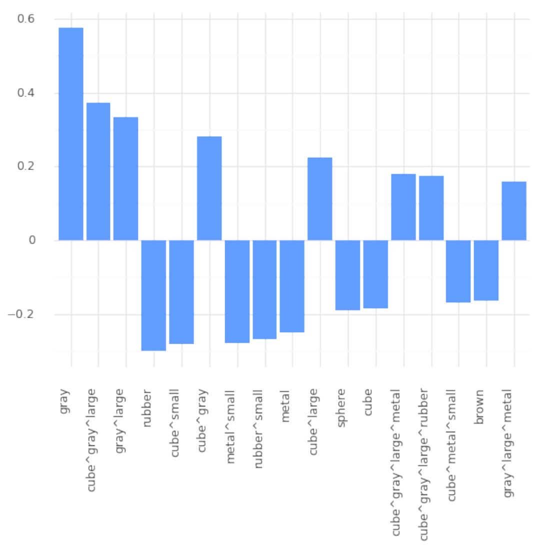


Figure 2: Global explanation for the subset of CLEVR-Hans3 which is classified in class B, with target class A

ber of repeating identical changes to be of high importance when transitioning between those classes and an indicator of the most essential class features. Negative importance indicates that a concept tends to be removed while positive importance indicates that it tends to be inserted.

We immediately notice that the classifier's bias for the confounded class A, is clearly detected. As mentioned previously, the confounding factor for class A is that the Large Cube is always Grey in the train set. This is apparent from the first three bars of fig.2, where the most important insertions seem to be the concepts: (Gray, GrayLargeCube, GrayLarge), instead of just (LargeCube).

B Algorithm Complexity

Regarding complexity, when using the graph representation, as mentioned before graph edit distance computation is NP-Hard and it has to be done $|\mathsf{EN}|^2$ times.

For the case of *concept set descriptions*, first we need to find the connected components of exemplars on the ABox graph which requires time $O(|\mathcal{A}|)$. Then we need to add $\exists r.C$ concepts to the labels of nodes a for which $r(a,b)C(b)$ is in the ABox, which also requires time $O(|\mathcal{A}|)$.

To compute the set edit distance between two labels of nodes ℓ_a, ℓ_b, each of which is a set of concepts (either atomic or of the form $\exists r.C$), we first construct a bipartite graph where each element of ℓ_a is connected to every element of ℓ_b and has a cost based on the TBox $\mathcal{T}$, as defined in section **??**. The computation of the cost of a single edit can be done with Dijkstra's algorithm on the TBox graph, requiring time $O(v + |\mathcal{T}|\log(v))$, where $v = |\mathsf{CN}| + |\mathsf{RN}|$ thus the construction of this bipartite graph requires time $O(|\ell_a||\ell_b|(v + |\mathcal{T}|\log(v)))$. On this bipartite graph we then compute the minimum weight full match using an implementation of Karp's algorithm for the problem with time complexity $O(|\ell_a||\ell_b|\log(|\ell_b|))$ to get the optimal set of edits from one set of concepts to another.

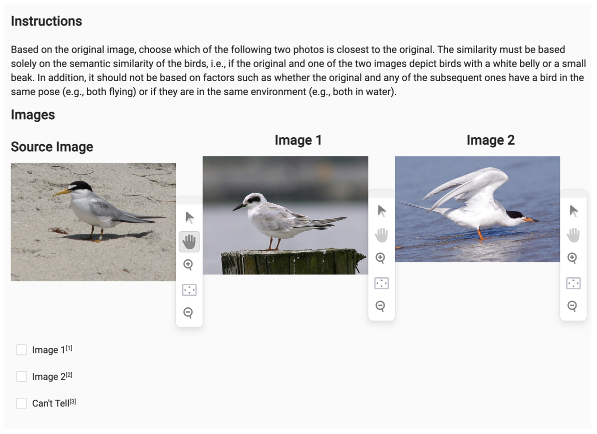
Finally, to compute the edit distance between two sets of labels L_1, L_2, each of which is **a set of sets** of concepts, we first compute the edit distance from each label in L_1 to every label in L_2 by using the procedure described in the previous paragraph for each pair of labels, meaning the set edit distance computation is performed $|L_1||L_2|$ times. Then to find the edit distance between L_1 and L_2 we use the same procedure as with sets of concepts (bipartite graph and full match), but this time the weights of the edges of the bipartite graph are assigned according to set the edit distance. Following from the above, for computing the edit distance between all pairs of exemplars, the total preprocessing time ends up being $O(|\mathsf{EN}|^2(L^2(l^2(v + T\log v + \log l)) + L^2\log L))$ where $|\mathsf{EN}|$ is the number of exemplars, L is the maximum number of nodes in a connected component of an exemplar, l is the maximum cardinality of a label of a node, T is the size of the TBox and v is the number of atomic concepts and roles. Having preprocessed the explanation dataset and saved the edit paths, an explanation can be provided in $O(|\mathsf{EN}|)$.

C Experimental Details

C.1 Human Survey

For the human survey, we used the Label Studio platform [2], which offers high-level flexibility and functionality. A screenshot of the annotation page is depicted in the following Figure 3. The classifiers that we selected for this experiment had the same pretrained weights that [Vandenhende et al., 2022] used in their work. The 33 participants were mainly graduate students, and PhD candidates, who responded to the call for participation. They were not offered compensation, they were volunteers. No information was provided to the participants, besides the call for participation, and the instructions for the labeling procedure. The study was conducted online.

[2]https://labelstud.io/



Instructions

Based on the original image, choose which of the following two photos is closest to the original. The similarity must be based solely on the semantic similarity of the birds, i.e., if the original and one of the two images depict birds with a white belly or a small beak. In addition, it should not be based on factors such as whether the original and any of the subsequent ones have a bird in the same pose (e.g., both flying) or if they are in the same environment (e.g., both in water).

Images

Source Image

Image 1

Image 2

Image 1[1]

Image 2[2]

Can't Tell[3]

Figure 3: A screenshot from the annotating platform. The first image always depicts a source image, whereas the second and the third are randomly the counterexample produced by [Vandenhende et al., 2022] method and the proposed one.

C.2 COCO setting

For the *restaurant-related* class we gathered all images from COCO that contained the concepts: 1. {dining table, person, pizza} (1000+ images) 2. {dining table, person, wine glass} (1200+ images). For the *bedroom-related* class we gathered all images that contained the label combinations of: 1. {bed, person} (1300+ images) 2. {bed, book} (800+ images) 3. {bed, teddy bear} (300+ images). On top of that, we wanted to make sure that we included some images that might be puzzling for the classifier. Those images were the ones including COCO label combinations of: 1. {bed, fork} (10 images) 2. {bed, spoon} (20 images) 3. {bed, wine glass} (20 images) 4. {bed, pizza} (10 images) 5. {dining table, bed} (170 images).

C.3 Scene Graph Generation

We used ”RelTR: Relation Transformer for Scene Graph Generation” as an SGG and run it on Google Colab, using the default model parameters. The predicted classes were the 150 entity classes of the Visual Genome dataset and its 50 relationship classes. Furthermore, one prediction is considered valid if their confidence is greater than 0.3.

Local Counterfactuals

In fig.4, we see three examples of local counterfactual explanations for the dataset that Scene Graph Generator is used on. The images on the left depict the ‘Pedestrian’ class, while the ones on the right, the ‘Driver’ class. The suggestions we receive for shifting the first image to the ”driver” class, is to add the concept ”rideˆbicycle” (∃ride.bicycle) on two of the men in the background and to change the gender of the person holding the bag from woman to man. Note that the ”rideˆbicycle” notations means For the second row transition, we have to change the node label from dog to man ($e_{\mathsf{Dog}} \rightarrow \mathsf{Man}$) and add the concepts ”rideˆbicycle” to the concepts man and woman. The edits between each pair of images appear sensible and minimal, similar to what we’ve seen on the other datasets.

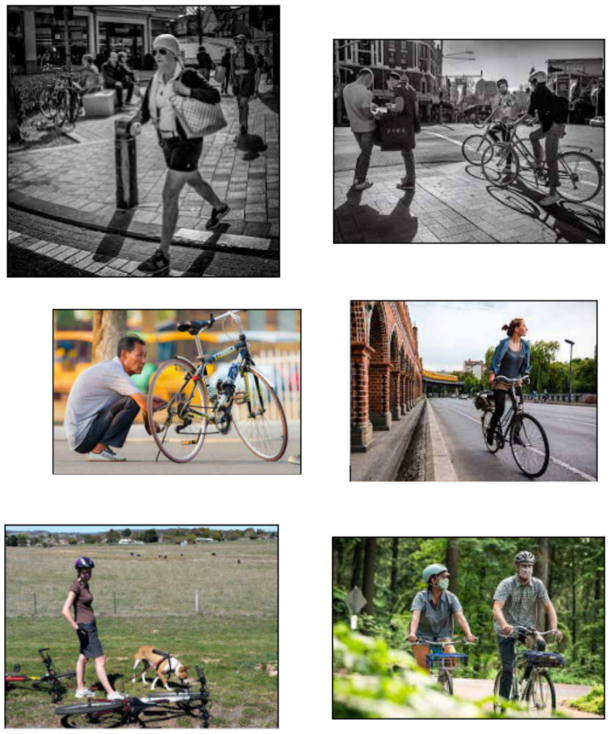


Figure 4: Three examples, shifting from ”pedestrians” (left) to ”drivers” (right). The main edits are additions of ”rideˆbicycle”, along with some gender changes and an edit of a dog to a man ($e_{\mathsf{Dog}} \rightarrow \mathsf{Man}$) in the last row.