Appendix for "Grounding Counterfactual Explanation of Image Classifiers to Textual Concept Space"

A. Details for Experimental Settings

A.1. Target Model Training

For the AwA2 dataset, we froze the feature extractor of pre-trained ResNet18 and ResNet50 provided by torchvision library [3] and trained only the top linear layer. For CUB dataset, ResNet18 and ResNet50 were fine-tuned without freezing from the pre-trained model. The final test accuracy of target models is shown in Table A1.

	ResNet18	ResNet50
AwA2	0.8108	0.9121
CUB	0.6288	0.7404

Table A1. Target model accuracy

A.2. Concept Prompting for C_{AwA2} and C_{CUB}

AwA2 C_{AwA2} has three more concept categories in addition to the six categories of C_{BRODEN}, which we named "size", "food", and "abstract". "Small" and "big" belong to category "size" and we prompted the concept as "`A photo of {size} object`". For "food", we prompted five food-relevant concepts using the template "`A photo of object eating {food}`". Category "abstract" includes abstract properties of animals rather than their visual properties, such as "fierce" or "domestic". Instead of excluding them, we used them by prompting them as "`A photo of {abstract} object`".

CUB Images and attribute names from the CUB dataset are different from BRODEN and AwA2 in two-fold: 1) The images in the CUB dataset are all images of bird, so we can use a more specific template using "bird" instead of "object". 2) The attribute names provided by CUB indicate fine-grained appearances of birds and are formatted as "attribute_type:attribute_value", e.g., "has_back_color::green". Taking the two properties into account, we fixed the source text to "`A photo of a bird`" and the target text to "`A photo of a bird with {attribute_value} {attribute_type}`". Please note that the prompting for the concepts can be done automatically with the specified templates.

B. Reproduce CCE [1] for Quantitative Evaluation

Original CCE paper, which is our primary competitor, did not experiment with the AwA2 and CUB datasets. Therefore, we reproduced the CAVs for C_{AwA2} and C_{CUB} to conduct the quantitative comparison. To obtain CAVs for a target model, two steps are required: 1) Collecting concept-positive/negative images, 2) Train linear classifiers that classify the embeddings of collected positive/negative images. Below illustrates how we collected images and learned CAVs.

B.1. Positive Dataset Collection

[2] suggested that the image can be crawled from the Internet to reduce manual dataset collection. Therefore, we crawled images from the Internet with GoogleImageCrawler provided by open source python library, Icrawler `https://icrawler.readthedocs.io/en/latest/`. To collect the positive images for a concept, we searched for images with the target text prompting the concept. For example, to collect positive images for the concept "blue", the search query was specified as "A photo of blue object". We crawled maximum of 50 images per concept. If the number of searched images did not exceed 50, fewer images were used.

B.2. Negative Dataset Construction

The negative dataset for $c \in C_{\text{AwA2}}$ was constructed by random sampling from positive datasets of other concepts. On the other hand, the negative dataset for $c \in C_{\text{CUB}}$ is constructed considering the attribute type to ensure that the concept is excluded. For a concept c, we randomly sampled positive images of different concepts that have the same attribute_type, but different attribute_value.

Due to the copyright issue, we could not attach the collected images, but we manually confirmed that images in positive datasets are all exhibit the corresponding concepts and the images comprising negative dataset are highly divergent. In few cases, we found some erroneous cases where negative images contained concepts. This can also be seen as re-proving that collecting "good" images of a

concept is nontrivial and requires significant amount of effort.

B.3. Learn CAVs

Following the original paper and code provided by the authors of CCE https://github.com/mertyg/debug-mistakes-cce, we trained support vector machine (SVM) for the newly crawled concepts. A number of classifiers corresponding to the number of concepts were trained. Note that concept classifiers need to be re-trained when that target model changes. The average test accuracy of the concept classifiers is shown in Table A2.

	CLIP+linear	**ResNet18**	**ResNet50**
AwA2	0.9882	0.9727	0.9804
CUB	0.7184	0.6980	0.7293

Table A2. Average accuracy of concept classifiers

C. Additional Counterfactual Explanations

C.1. Additional Results for Debugging Misclassification

Figure A2 shows additional counterfactual explanations generated with respect to images misclassified by CLIP+linear. Here, the target classes were specified as the correct classes. Similar to the main manuscript, target classes are denoted with example images. Red and blue colors denote positive and negative concept importance scores, respectively.

C.2. Qualitative Comparison with CCE

Figure A2 shows additional explanations of ResNet18 generated using CCE and CounTEX. CCE defined their own library C_{CCE} which includes curated concepts from C_{BRODEN} and the names of corruption types such as "blurred". The total number of concepts is 170, much less than C_{BRODEN}. We also used the above-mentioned official code provided by CCE authors. We displayed top-2 and bottom-2 concepts considering that fewer concepts were used. CounTEX consistently outperforms CCE where CCE assigns large importance scores to concepts that are not relevant to the target class and do not even exist in the input image.

References

[1] Abubakar Abid, Mert Yuksekgonul, and James Zou. Meaningfully debugging model mistakes using conceptual counterfactual explanations. In *International Conference on Machine Learning*, pages 66–88. PMLR, 2022. 1

[2] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pages 2668–2677. PMLR, 2018. 1

[3] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. 1

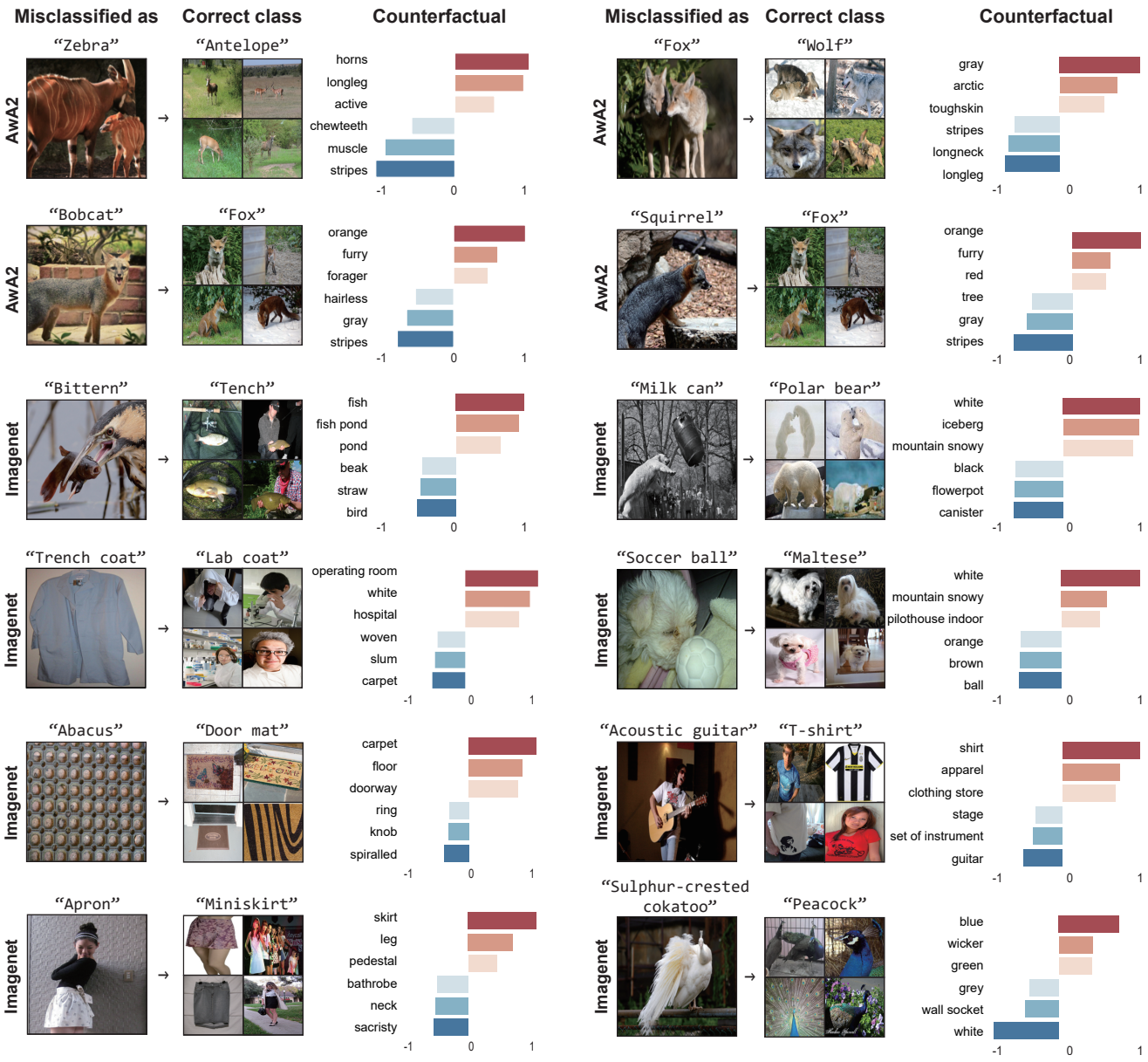
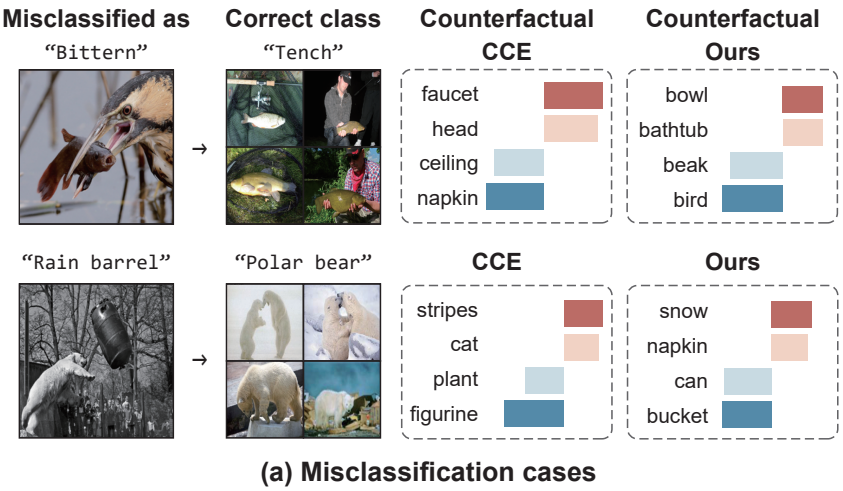
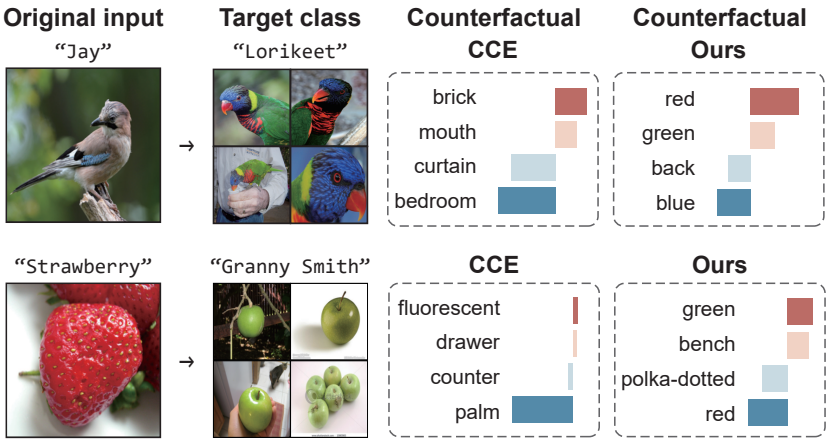


Figure A1. Counterfactual explanation for the samples misclassified by CLIP+linear. C_{BRODEN} is used to generate the explanations. First two rows show the results for AwA2 dataset and remaining bottom rows show the results for ImageNet dataset.



(a) Misclassification cases



(b) Correct classification cases

Figure A2. Qualitative comparison between CCE and CounTEX. The target model is CLIP+linear and the explanations are generated with respect to C_{CCE}. Following CCE, we displayed top-2 and bottom-2 concepts. (a) Misclassification results (b) Correctly classified cases.