Supplementary Material
SIC: Similarity-Based Interpretable Image Classification with Neural Networks

Tom Nuno Wolf Emre Kavak Fabian Bongratz Christian Wachinger
Lab for Artificial Intelligence in Medical Imaging, Technical University of Munich, Germany
Munich Center for Machine Learning (MCML), Germany
tn.wolf@tum.de

A. Appendix

A.1. On the Faithfulness of B-cos Explanations

B-cos modles are piece-wise linear models that allows to summarize each input as an input-dependent linear transform $f(x) = \mathbf{W}_{1\to L}(x)x$, derived in Eq. 1. Explanations are computed in terms of the weight matrix $\mathbf{W}_{1\to L}(x)$ by multiplying each feature with its matrix weight $\phi(x)_i = [\mathbf{W}_{1\to L}]_j^T \odot x]_{(ch,i,j)}$. To show that computing explanations for a forward pass is faithful (i.e., satisfies below six axioms introduced by Sundararajan et al. [30]), we need to show that they hold considering the transformation matrix $\mathbf{W}_{1\to L}(x)$. While $\mathbf{W}_{1\to L}(x)$ is indeed input-dependent, it is effectively fixed during the computation of the explanation for its specific input x. This is because the explanations $\phi(f, X)$ aim to attribute the output $f(x)$ to the input features of x using the weights at that point. Therefore, we assume a fixed $\mathbf{W}_{1\to L}(x)$ when explaining the forward pass of an input x, and acknowledge that while $\mathbf{W}_{1\to L}(x)$ varies across different inputs, it remains constant within the context of computing local explanations $\phi(f, x)$ for a particular x. In addition, we consider the case of single-class prediction, where the weight matrix simplifies to a single-column form ($\mathbf{W}_{1\to L}(X) \in \mathbb{R}$) for in input vector $X \in \mathbb{R}$. This allows us to focus on the core properties of the explanation mechanism without loss of generality since explaining the prediction of a class in the multiclass setting is analogous to explaining its corresponding column vector of the matrix $\mathbf{W}_{1\to L}(x)$.

We abbreviate $\mathbf{W}_{1\to L}(x)$ with $W(X)$ for ease of notation in the following. Let SIC be defined as $f(X) = W(X) \cdot X$ and its contributions similarily by $\phi(f, X) = W(X) \odot X = (W(X)_1 \times X_1, \dots, W(X)_N \times X_N)$ (remark that $\cdot$ denotes the scalar product and $\odot$ the element-wise multiplication, $\times$ the multiplication of real numbers).

Now, we reformulate $f(X)$:

$$f(X) = W(X) \cdot X = \sum_{i=1}^{N} W(X)_i \times X_i = \sum_{i=1}^{N} \phi(f, X)_i. \tag{2}$$

Completeness: *The sum of feature attributions should add up to the model output.*

This follows trivially from the reformulation in Eq.2.

Sensitivity: *If changing only one feature's value changes the prediction of the model, this feature's attribution should be non-zero.*

We reformulate: $f(X) = \sum_{j=1}^{N} \phi(f, X)_j = \phi(f, X)_i + \sum_{j\neq i}^{N} \phi(f, X)_j$.

Modifying X only in i, denoted by $\hat{X}$, and $f(X) \neq f(\hat{X})$ yields:

$f(X) = \phi(f, X)_i + \sum_{j\neq i}^{N} \phi(f, X)_j \neq f(\hat{X}) = \phi(f, \hat{X})_i + \sum_{j\neq i}^{N} \phi(f, X)_j$.

and allows to substitute $\sum_{j\neq i}^{N} \phi(f, X)_j$.

$\implies \phi(f, X)_i \neq \phi(f, \hat{X})_i$, which means that at least one of the two attributions in non-zero.

Implementation Invariance *If two models are functionally equivalent, i.e. the outputs are equal for all inputs, despite having different implementations, their contributions should always be identical.*

Let f_1, f_2 be implementations.

Then: $f_1 = W_1(X) \cdot X$, $f_2 = W_2(X) \cdot X$, $\forall X \implies W_1(X) = W_2(X)$, $\forall X$.

It follows that $W_1(X) \odot X = W_2(X) \odot X$, $\forall X \implies \phi(f_1, X) = \phi(f_2, X)$, $\forall X$.

Dummy *If a model f does not depend on some feature X_i, its attribution should always be zero.*

If $f(X)$ is does not depend on some feature X_i, it follows that $W(X)_i \times X_i = 0$, $\forall X_i \in \mathbb{R}$

$\implies W(X)_i = 0 \,\forall, X_i \in \mathbb{R}$

$\implies \phi(f, X)_i = 0$, $\forall X_i \in \mathbb{R}$.

Linearity *If the output of a model is a linear combination of two models, the attribution of the combined model should be the weighted sum of the contributions of the original models.*

We need to show that $f(X) = \alpha f_1(X) + \beta f_2(X) \implies \phi(f, X) = \alpha\phi(f_1, X) + \beta\phi(f_2, X)$.

$f(X) = \alpha f_1(X) + \beta f_2(X) = \alpha(\sum_{i=1}^{N} W_1(X)_i \times X_i) + \beta(\sum_{i=1}^{N} W_2(X)_i \times X_i) = \alpha(\sum_{i=1}^{N} \phi(f_1, X)_i) + \beta(\sum_{i=1}^{N} \phi(f_2, X)_i) = \alpha\phi(f_1, X) + \beta\phi(f_2, X)$.

Symmetry-Preserving *If swapping two features does not change the model output for all possible values, they should have identical attributions.*

Let X_i, X_j be two features.

If swapping the two does not change the model output for all possible values, it follows that

$W(X)_i \times X_i = W(X)_j \times X_j$, $\forall X_i, X_j \in \mathbb{R}$

$\implies \phi(f, X)_i = \phi(f, X)_j$, $\forall X_i, X_j \in \mathbb{R}$.

A.2. On the Faithfulness of support vectors

On a theoretical level, the global explanations are the explanations of an intermediate B-cos neuron. Thus, they satisfy the above axioms as well and can be interpreted as the information encoded in the respective support vector.

When providing local explanations, we view support vectors as the weights of a B-cos linear transform. Hence, computing the explanation of the prediction of the class in terms of the output yields the input features compressed in

any of the support vectors; and the log-probability scores of each support vector indicate to which extend each was found.

Justification for ReLU in SIC In contrast to other functions that could be used to implement $\oplus$, the ReLU activation does not alter the range of the input to potentially huge or tiny numbers like, e.g. the exponential. However, ReLU activations are commonly scrutinized for their role in setting negative activations to zero, which can cause deep neural networks to interpret this suppression as the absence of a feature. Thus, many XAI methods (e.g. gradient-based) yield misleading attribution maps, as they cannot account for this type of contribution. However, this phenomenon is tightly controlled in SIC, where the sole operation following the ReLU activation is a similarity computation between the latent vectors of two images. As a result, the absence of a feature does not, by design, affect the probability of class logits, rendering it valid to disregard any input and its explanation from neurons corresponding to such features.

A.3. Implementation and Training Details

Optimization All models were optimized using the AdamW optimizer with fixed hyperparameters: weight decay set to 0.0, betas configured to (0.9, 0.999), and an epsilon value of 1e-08. For FunnyBirds, we selected the default parameters of the framework: a learning rate of 0.001, and weight decay of 0.0, trained for 100 epochs with a temperature of 30 and a batch size of 8. For all other experiments, we conducted an extensive grid search over the hyperparameter space (see Tab. A.2) to identify the optimal settings based on accuracy. The training employed a custom learning rate scheduling policy, which began with a linear warm-up phase over the first two epochs, increasing the learning rate from 10% to the initial learning rate. This was followed by maintaining the initial learning rate until 50% of the total training iterations were completed. Subsequently, the learning rate was decayed by a factor of 0.5 at every subsequent 10% milestone of the total iterations. Batch sizes were set to 32 for all datasets except RSNA, which utilized a batch size of 64. The number of training epochs was standardized to 50 across all datasets, with RSNA trained for 100 epochs. Pretrained feature extractor weights were sourced from torchvision `https://pytorch.org/vision/stable/index.html` and the official B-cos repository `https://github.com/B-cos/B-cos-v2` for all experiments except those involving RSNA. We train ProtoPNet with the default training recipe provided on the official GitHub implementation (`https://github.com/cfchen-duke/ProtoPNet`) and adapted the code for VOC for multi-label classification. Note that the NW-Head is not applicable to multi-label classification tasks as it relies on softmax for normalization, which can not be adapted with minor modifications as done with ProtoPNet.

Model-specific parameters included setting B-cos $B = 2$ and using three support samples. We train the NW-Head in the replace cluster mode. For SIC, the grid search explored learning rates of 0.01, 0.003, 0.001, and 0.0003 for the RSNA dataset, and 0.0022, 0.001, 0.00022, 0.0001, 0.000022, and 0.00001 for other datasets. Temperature parameters tested were 10, 30, and 50 for RSNA, and 10 and 30 for the remaining datasets. For BlackBox models, the grid search included the same learning rates as SIC and additionally varied the weight decay parameter, evaluating values of 0.0 and 0.0001. The best-performing models were selected based on their accuracy metrics.

Architecture All backbones are extended by a linear projection layer to extract feature vectors $\in \mathbb{R}^d = 128$. The $\oplus$ function is implemented as a ReLU activation in the evidence predictor $\mathcal{E}$, and the similarity function "sim" is implemented by a B-cos linear layer. In multi-label classification settings, only training samples with a single class label are considered for support labels.

Datasets, Data Pre-Processing and Augmentation In our study, we adhered to standard dataset splits to ensure consistency and comparability with existing research. Specifically, for Stanford Dogs, and Pascal VOC, we utilized the official training and testing partitions as provided. Regarding the RSNA dataset, we employed the training set from Stage 2 of the RSNA challenge and further subdivided it by randomly sampling 25% of the training data to form a separate test set, ensuring that the splits were stratified by class label. Additionally, we verified that the distributions across sex remained consistent following the splitting process, as illustrated in Tab. A.3.

For non-Bcos models, we applied standard normalization using the mean and standard deviation calculated from the training set. Specifically, images were standardized with mean values $[0.485, 0.456, 0.406]$ and standard deviations $[0.229, 0.224, 0.225]$.

Data pre-processing for the Pascal VOC and Stanford Dogs datasets included a `RandomResizedCrop` of torchvision to a target size of 224 pixels using bilinear interpolation. We implemented a `RandomHorizontalFlip` with a probability of 0.5. For non-Bcos models, standardization was performed as mentioned above. Additionally, we employed `RandomErasing` with a probability of 0.5 and a scale range of 2% to 33% of the image area.

We follow the FunnyBirds framework and use the default data processing pipeline and compute the metrics with the unsmoothed explanations ϕ, i.e. the alpha channel of RGBA explanations.

Method	Dataset	Backbone	Best LR	Best WD	Best Temp.	Acc (%)
BlackBox	VOC	DenseNet121	2.2×10^{-5}	0.0	-	96.42
	VOC	ResNet50	1.0×10^{-5}	0.0	-	96.71
	VOC	VitC	1.0×10^{-4}	0.0	-	96.35
	StanfordDogs	ResNet50	1.0×10^{-5}	0.0	-	87.16
	RSNA	DenseNet121	3.0×10^{-4}	1.0×10^{-4}	-	79.78
NW-Head [31]	StanfordDogs	ResNet50	1.0×10^{-5}	0.0	-	85.16
	RSNA	DenseNet121	3.0×10^{-4}	1.0×10^{-4}	-	79.30
ProtoPNet [8]	VOC	DenseNet121	-	-	-	91.19
	VOC	ResNet50	-	-	-	89.31
	StanfordDogs	ResNet50	-	-	-	49.24
	StanfordDogs	ResNet50*	-	-	-	74.70
	RSNA	DenseNet121	-	-	-	78.68
BagNet [6]	VOC	BagNet17	2.2×10^{-5}	1.0×10^{-4}	-	94.56
	StanfordDogs	BagNet17	2.2×10^{-5}	0.0	-	62.32
	RSNA	BagNet17	3.0×10^{-4}	1.0×10^{-4}	-	77.70
B-cos [5]	VOC	DenseNet121	1.0×10^{-4}	-	-	96.48
	VOC	ResNet50	1.0×10^{-4}	-	-	96.85
	VOC	VitC	2.2×10^{-5}	-	-	96.79
	StanfordDogs	ResNet50	1.0×10^{-4}	-	-	85.54
	RSNA	DenseNet121	1.0×10^{-2}	-	-	78.73
SIC	VOC	DenseNet121	2.2×10^{-4}	-	30	96.73
	VOC	ResNet50	1.0×10^{-4}	-	30	97.00
	VOC	VitC	2.2×10^{-5}	-	10	96.72
	StanfordDogs	ResNet50	1.0×10^{-4}	-	10	83.46
	RSNA	DenseNet121	3.0×10^{-4}	-	50	79.13

Table A.2. Best configurations and performance of each model.
$*$: optimized with training recipe published in [9] instead of [8].

Split	Sex	Target	Num. Samples
Train	F	No Opacity	6,770
	F	Opacity	1,873
	M	No Opacity	8,734
	M	Opacity	2,636
Test	F	No Opacity	2,246
	F	Opacity	629
	M	No Opacity	2,922
	M	Opacity	874

Table A.3. Dataset statistics of the RSNA dataset.

For the RSNA dataset, intensity rescaling was conducted to map the maximum Hounsfield Units to a range between 0 and 1. Data augmentation techniques included a `RandomHorizontalFlip` with a probability of 0.5 and a `RandomAffine` transformation. The `RandomAffine` parameters consisted of rotations up to 45 degrees, translations of $\pm 15\%$ in each direction, and scaling factors ranging from 0.85 to 1.15. We also applied `RandomErasing` with a probability of 0.5, using patch sizes between 5% and 20% of the image area.

A.3.1. FunnyBirds - ResNet50

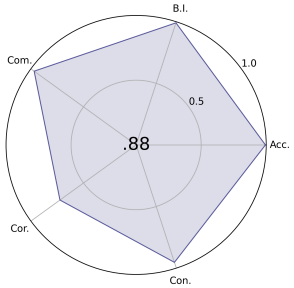


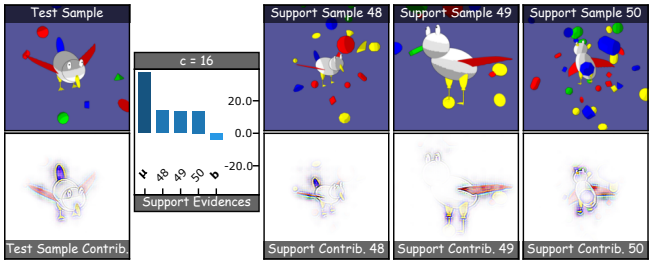
Figure A.6. FunnyBirds [12] result of main metrics Completeness (Com.), Background Independence (B.I.), Accuracy, Contrastivity (Con.), Correctness (Cor.).

A.3.2. CUB-200-2011 - ResNet50

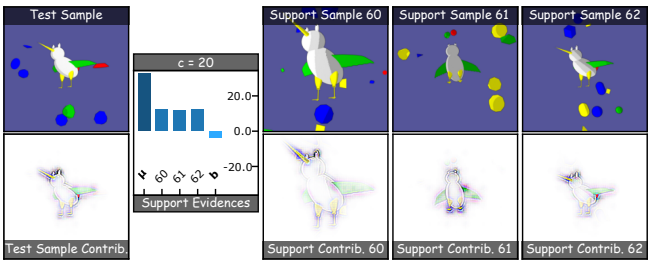
We trained SIC on the official splits of CUB-200-2011 [32] (cropped ROI), achieving an accuracy of 79.8% (cmp. ProtoPNet 76.1-80.2% in Tab. 1 (top) [8]).

Method	A	BI	CSDC	PC	DC	D	SD	TS	Com.	Cor.	Con.
ProtoPNet [8]	0.94	**1.00**	0.93	0.91	0.92	0.58	0.24	0.46	0.75	0.24	0.46
BagNet [6]	**1.00**	**1.00**	0.95	**0.98**	0.91	0.91	**0.76**	**0.99**	0.93	0.76	0.99
B-cos [5]	0.96	0.87	0.93	0.88	0.94	0.86	0.69	0.89	0.89	0.69	0.89
SIC (ours)	0.99	0.99	**0.96**	**0.98**	**0.96**	**0.97**	0.72	0.95	**0.97**	0.72	0.95

Table A.4. Results on the FunnyBirds Framework with a ResNet50 backbone. Note that *Completeness* (Com.) is the mean across Controlled Synthetic Data Check (CSDC), Preservation Check (PC), Deletion Check (DC), and Distractability (D); *Correctness* (Cor.) equals Single Deletion (SD), and *Contrastivity* (Con.) equals Target Sensitivity (TS).



(a) Prediction of the representative FunnyBirds [12] test sample. The primary location of relevance is the tail of the bird, while the wings, beak, eyes, and legs show a lower contribution. We observe a variety of poses of representative support samples, e.g. a strong focus on the tail and wings in the first support sample, a focus on the legs, eyes, and wings in the second support sample, and a focus on all visible concepts in the last support sample.

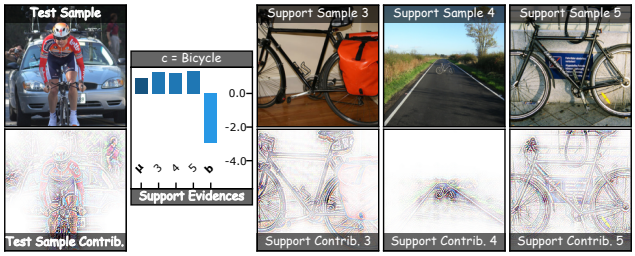


(b) Prediction of another test sample. Similar to (a), all concepts are relevant for the decision making. While the tail appears to be the most important concept for the classification in (a), the eyes seem more important in the specific example. Additionally, the long beak contributes over its whole length, suggesting that the model is sensitive to the appearance of the concept rather than encoding color only.

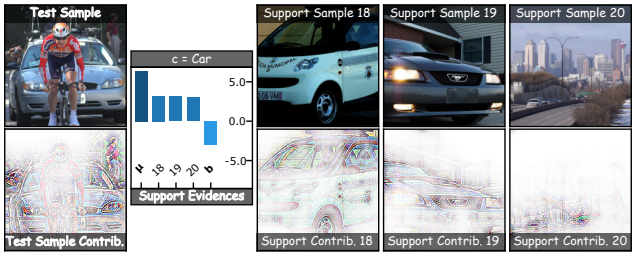
Figure A.7. Comparison of Explanations.

A.4. Additional Figures

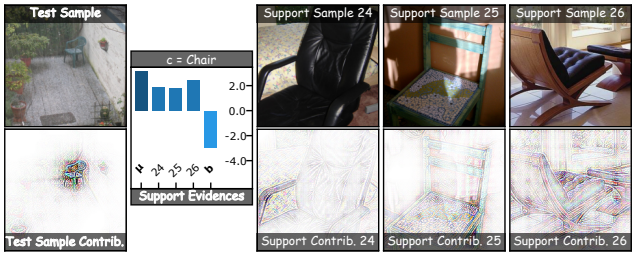
A.4.1. VOC - ResNet50



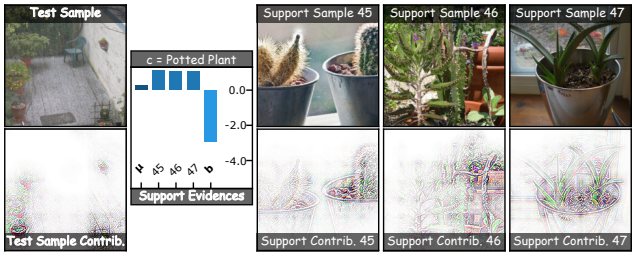
(a) Prediction for the bicycle class.



(b) Prediction of the car class. Note the shift in contributions compared to (a).

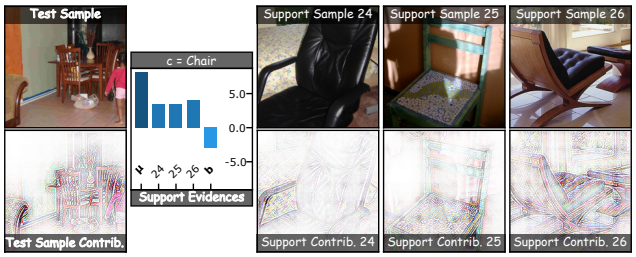


(c) The model correctly found the chair.

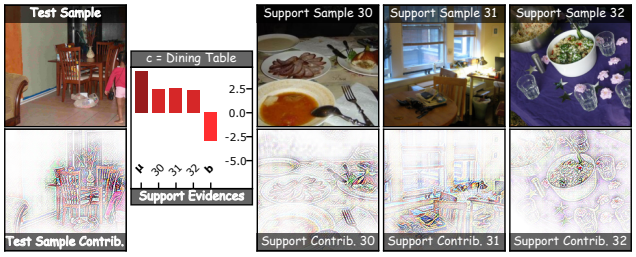


(d) And also the potted plants. Note the shift in contributions compared to (c).

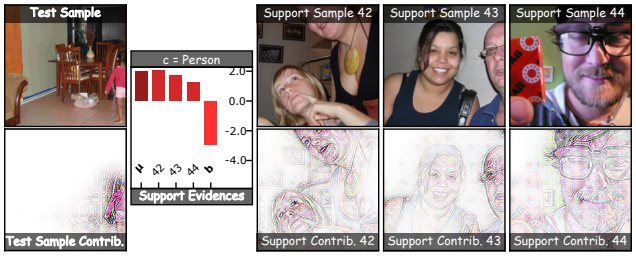
Figure A.8. In this and the following figures, they layout introduces marginal changes compared to the main figure. Refer to (a) for the respective explanations. This model was trained with a ResNet50 backbone on VOC.



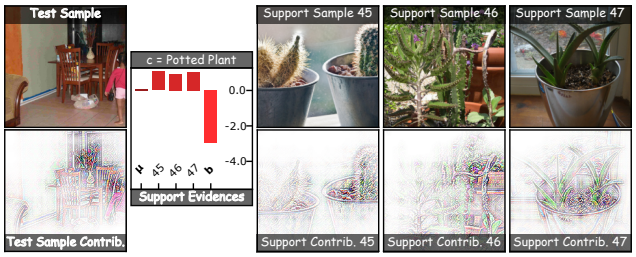
(a) Prediction for the chair class.



(b) Wrong prediction according to the label of the dining table.



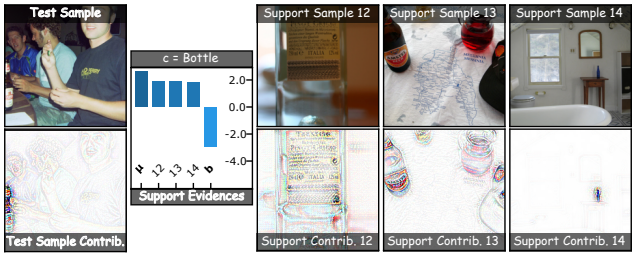
(c) Wrong prediction according to the label of a person.



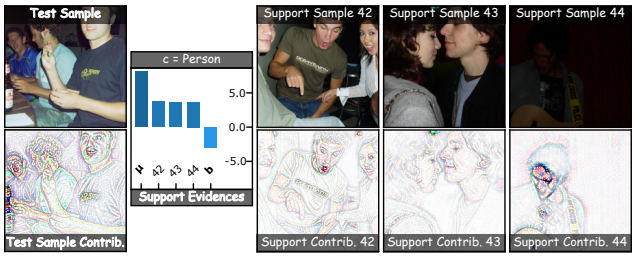
(d) And a wrong prediction, as the flower on the table was mistaken with a potted plant.

Figure A.9. VOC ResNet50

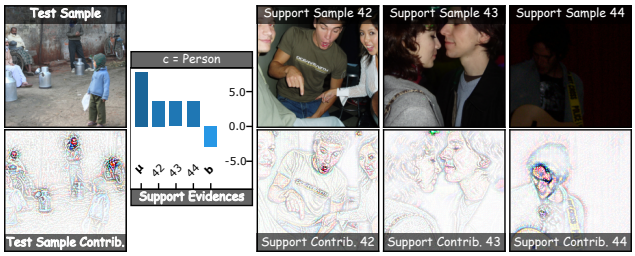
A.4.2. VOC - VitC



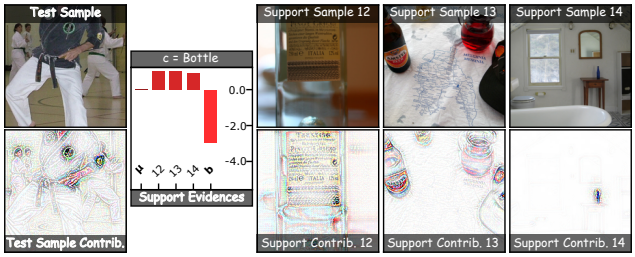
(a) Prediction for the bottle class.



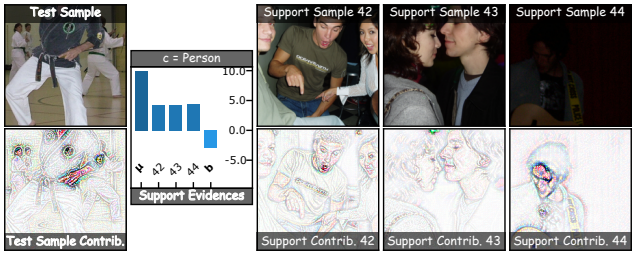
(b) Prediction of the person class.



(c) For this test sample, this backbone was the only one not to mistake the milk jugs with bottles.



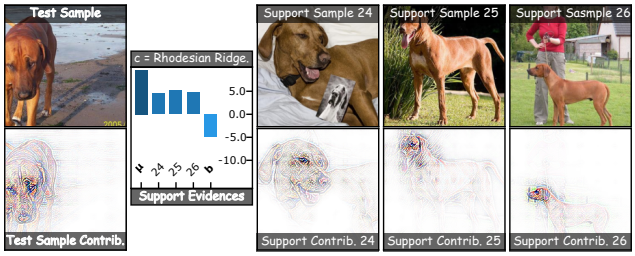
(d) Wrong prediction: The martial arts logo was mistaken with the labels of the first two support samples.



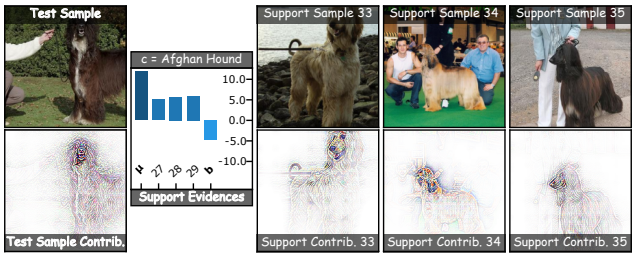
(e) Person was correctly classified.

Figure A.10. VOC VitC

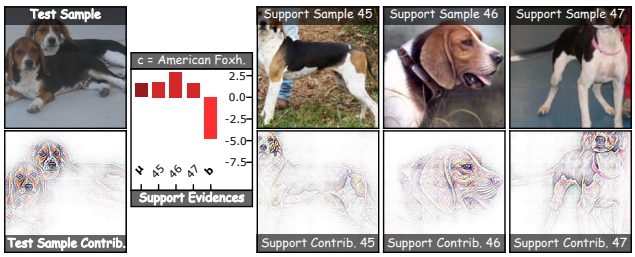
A.4.3. Dogs - ResNet50



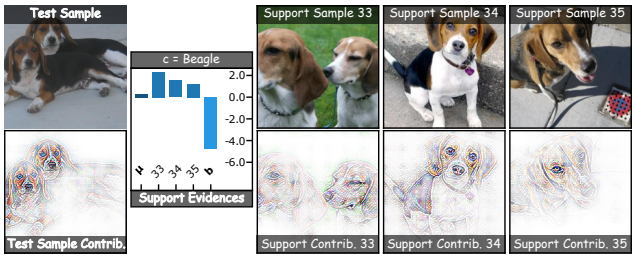
(a) Correct classification. Note the insensitivity to the background.



(b) Correct classification. Note the insensitivity to the background.

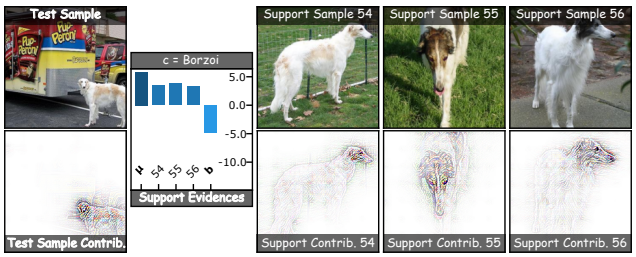


(c) Incorrect classification. The model was very unsure about the class...

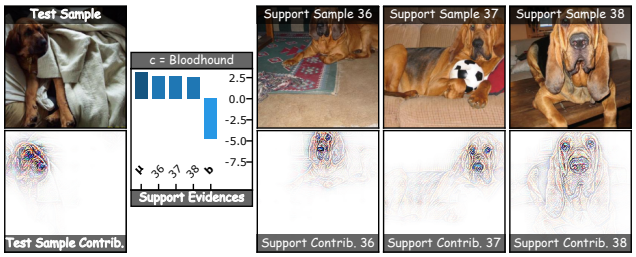


(d) The model found very little evidence for two of the three support samples.

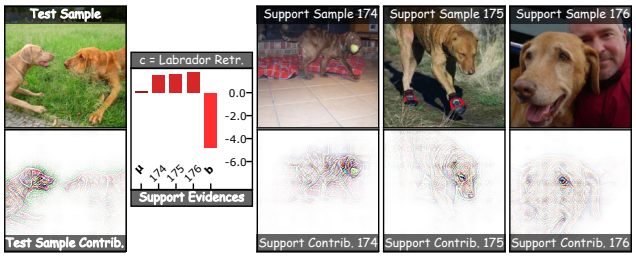
Figure A.11. ...and the correst class probability suggests that the two breeds are indeed pretty similar.



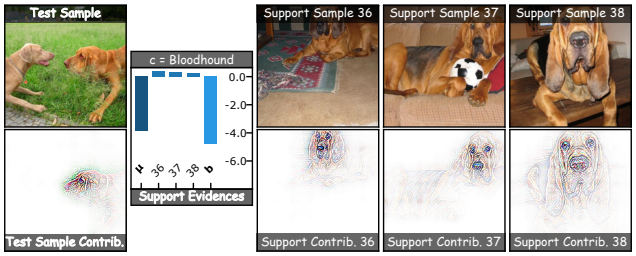
(a) Correct classification. Note the insensitivity to the background.



(b) Correct classification. Note the insensitivity to the background.

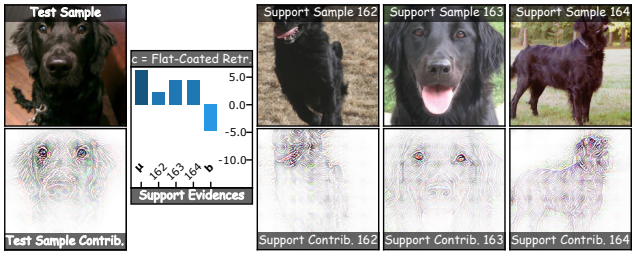


(c) Incorrect classification. The explanations suggest that the dog on the left was prioritized by the network...

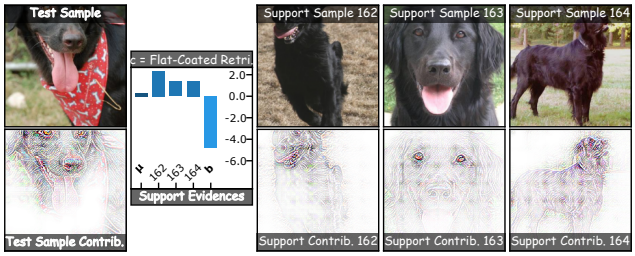


(d) ...in contrast to the (arguable) true label. Compared to support samples, the tip of the nose is relatively bright.

Figure A.12. Dogs Dataset ResNet50



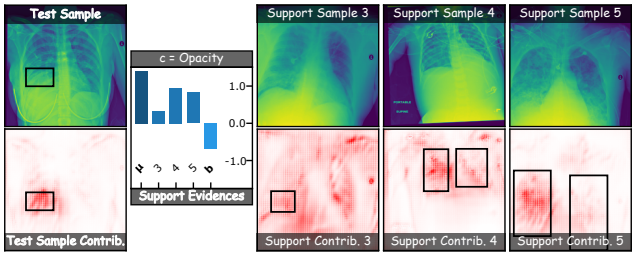
(a) Correct classification. Note how different poses of the support samples lead to different contributions, which is also evident in ...



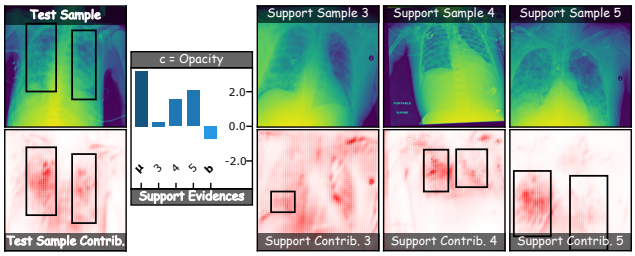
(b) ...this figure. Here, the first support sample is most similar to the test image, which is also cropped along the eyeline.

Figure A.13. Dogs Dataset ResNet50

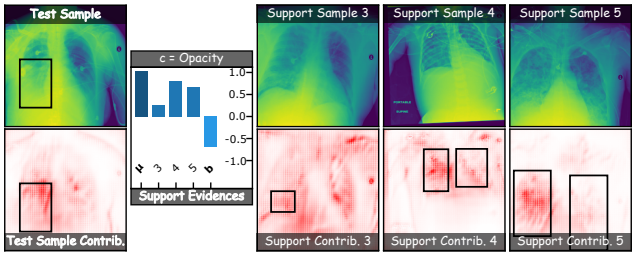
A.4.4. RSNA - DenseNet121



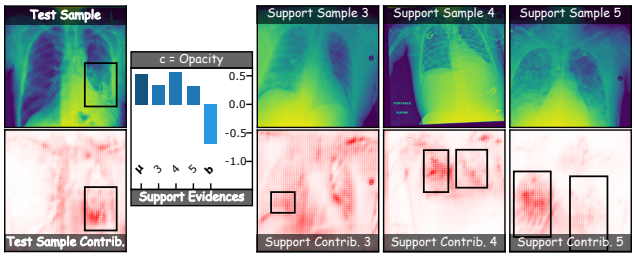
(a) Correct classification of an occlusion present sample.



(b) Correct classification of an occlusion present sample.



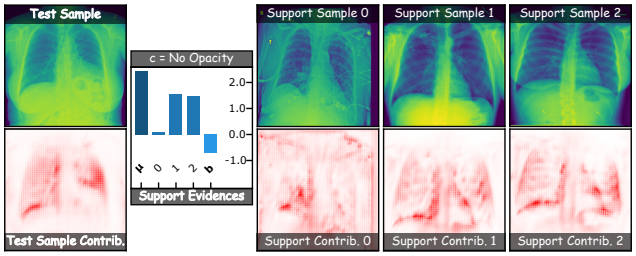
(c) Correct classification of occlusion present sample.



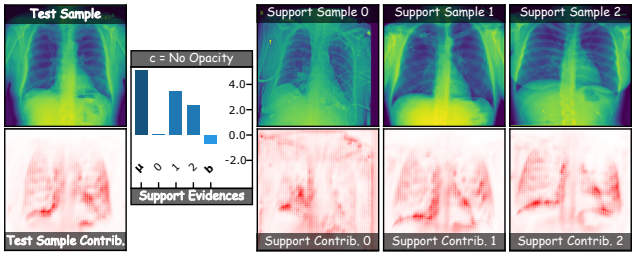
(d) Correct classification of an occlusion present sample.

Figure A.14

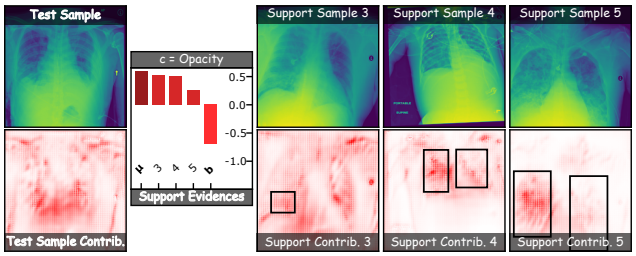
A.4.5. VOC - DenseNet121: Prototypes and B-cos Similarities



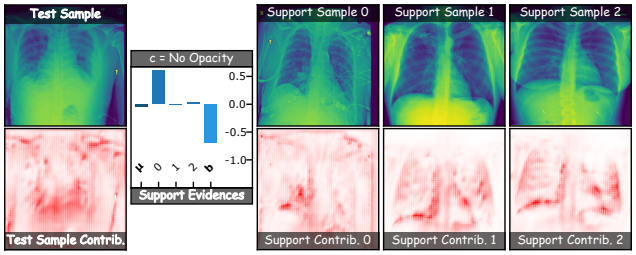
(a) Correct classification of a control sample.



(b) Correct classification of a control sample.



(c) Incorrect classification of a control sample. The model likely mistook the relatively large heart for pathology.



(d) The model found very little evidence for two of the three support samples.

Figure A.15

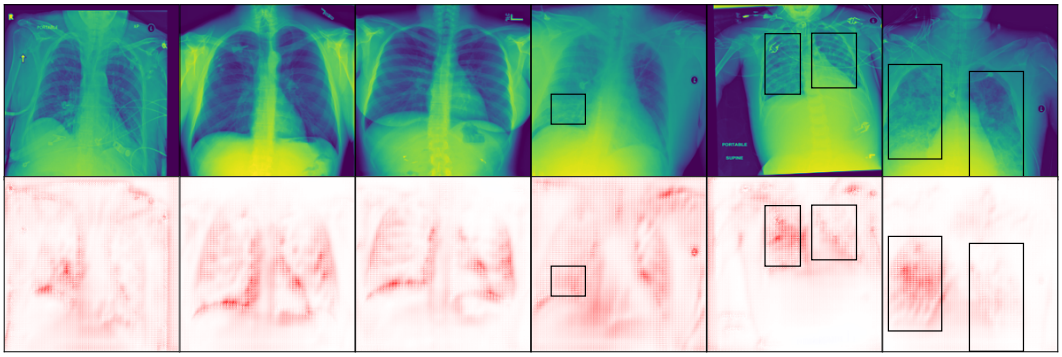


Figure A.16. Prototypes of SIC trained with a DenseNet121 backbone on RSNA. Bounding boxes indicate where medical staff found occlusion of the lungs. The first three columns are the support samples of controls, the next three are support samples with occlusion present.

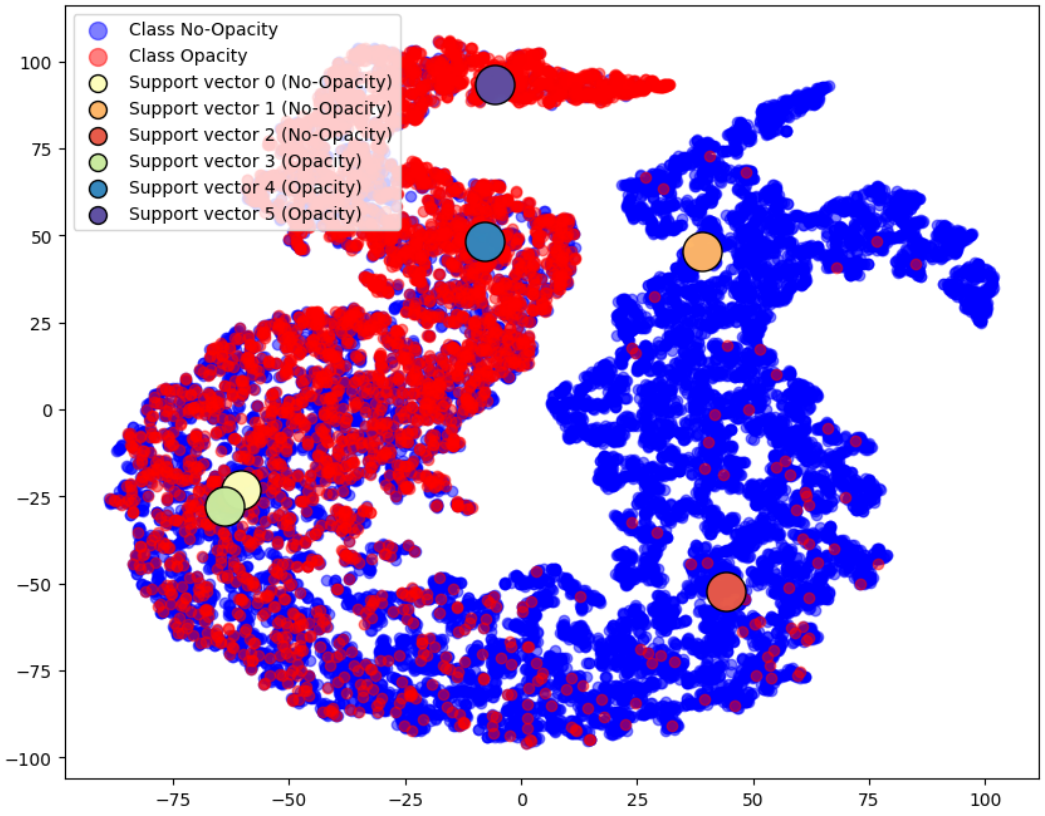


Figure A.17. t-SNE plot of the latent vectors of the RSNA training set. Red denotes samples of the class ”opacity” and blue the class ”no opacity”. Support vector 0 and support vector 3 are very close in the projected latent visualization.

Figure A.18. Global explanations for SIC trained on Pascal VOC with a DenseNet121 backbone. The respective second rows show the raw contribution maps $\phi^L_{j=c}$, in which red denotes positive contributions and blue (absent) negative contributions. The respective third columns present RGBA explanations.

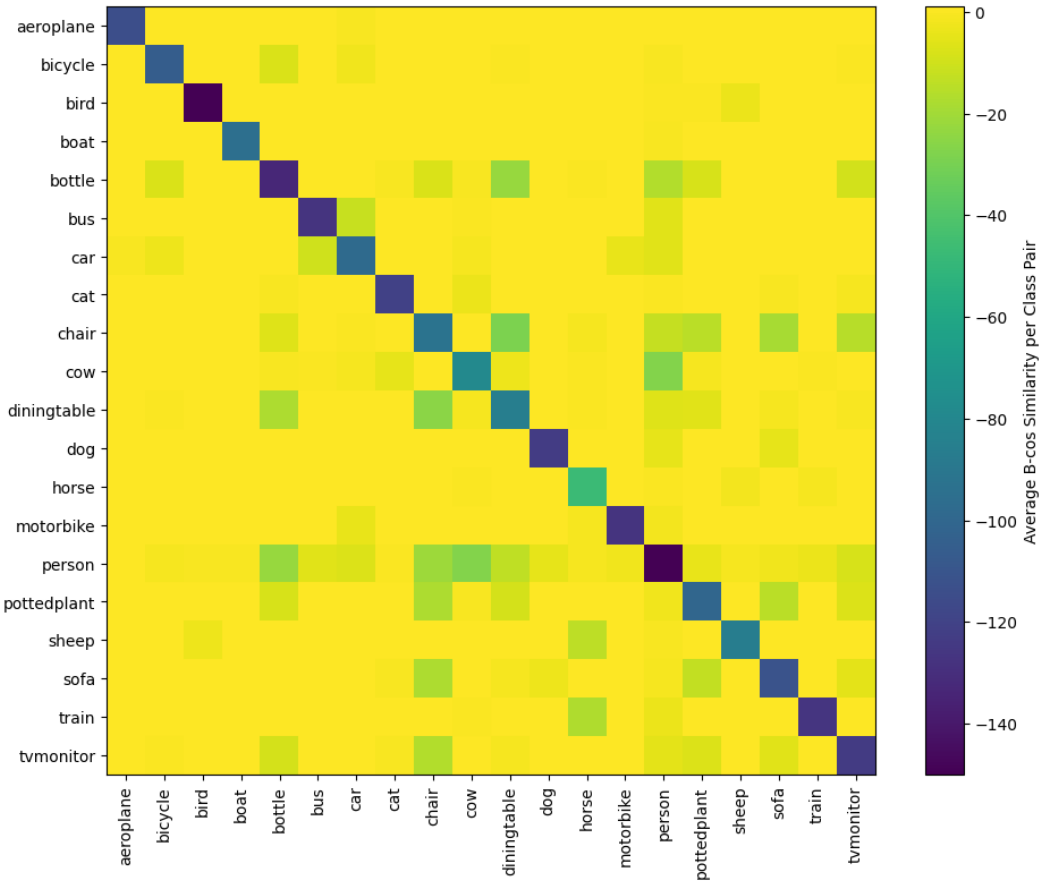


Figure A.19. Heatmap of Intra-class and Inter-class B-cos Similarities for SIC trained on Pascal VOC with a DenseNet121 backbone.