A APPENDIX

BROADER IMPACT

Our work has the potential of gaining end users' trust in deep neural networks and making it possible to answer "why" by creating human-like explanations. Future implications could include sensitive fields where practitioners are desperate to understand how black-box models decide on a specific prediction before their deployment. A prominent example is medical imaging where it is sine qua non to see how DNNs make decisions. Our technique could help domain experts trust the automated system they get help from. This is achieved differently from currently available techniques that can only highlight the part of an image that DNNs seem to rely on. We argue that self-explainable DNNs are the future of machine learning applications. As DNNs are now currently the most preferred techniques and their most apparent limitation is the complicated decision process, we bring about a novel and cheap technique that, to the best of our knowledge, has never been proposed before.

B APPENDIX

All the experiments were implemented on a personal laptop with the following specifications:

- i7-8750H CPU
- GeForce GTX 1060 GPU
- 16GB RAM

Training of $\mathrm{MLP_L}$ takes around 15 minutes. Training of X-MLP and X-CNN takes around 30 minutes and 80 minutes, respectively. The pre-trained feature extraction models (i.e., ψ) ResNet101 and VGG16 are downloaded from Keras' website[2].

B.1 GENERATION OF THE TRAINING DATASET $\mathcal{T}$ FOR $\mathrm{MLP_L}$

Recall that we have the attribute-class matrix $\mathbf{A} \in \mathbb{R}^{C\times K}$, where C and K represent the number of classes and the number of attributes per class, respectively. Table 4 shows the full matrix $\mathbf{A}$ for the AwA2 dataset. The original matrix $\mathbf{A}$ can directly form a dataset, i.e.,

$$\mathcal{T}_0 = \left\{ (\tilde{\boldsymbol{y}}_k, y_k) \mid \tilde{\boldsymbol{y}}_k \in \tilde{\mathcal{Y}}, y_k \in \mathcal{Y}, k = 1, 2, \ldots, C \right\}, \tag{4}$$

which is too small to train $\mathrm{MLP_L}$; note that $\tilde{\boldsymbol{y}}_k = (\tilde{y}_k^1, \cdots, \tilde{y}_k^K)$ is the k-th row of $\mathbf{A}$ for class k, and $\tilde{y}_k^i$ is the i-th attribute of $\tilde{\boldsymbol{y}}_k$.

We augment $\mathcal{T}_0$ by upsampling each sample $(\tilde{\boldsymbol{y}}_k, y_k) \in \mathcal{T}_0$ to $\hat{C}$ number of samples by randomly manipulating $\hat{K}$ number of attributes of $\tilde{\boldsymbol{y}}_k$ among the total K, with the aim of perturbing the original samples. The values of the selected attributes for manipulation can be conducted randomly. In our setting, we use two values $\beta_0 > 0$ and $\beta_1 < 0$, and change the positive values of the selected attributes to β_1, otherwise, to β_0. Note that this setting is an arbitrary choice (here we use $\beta_0 = 1.5$ and $\beta_1 = -0.5$), which can be replaced by other ways appropriate. We finally generate a training dataset $\mathcal{T}$ with $C\hat{C}$ number of samples. In our experiments, $\hat{C}$ and $\hat{K}$ are set to 100 and 8, respectively.

The way of generating dataset $\mathcal{T}$ finally used to train $\mathrm{MLP_L}$ is summarised in Algorithm 1 below.

B.2 CLASSIFICATION ACCURACY PERFORMANCE OF $\mathrm{MLP_L}$

Although our main focus in this article is XAI rather than the classification accuracy of $\mathrm{MLP_L}$, it is worth evaluating the classification accuracy performance of $\mathrm{MLP_L}$ under the training dataset $\mathcal{T}$ generated using Algorithm 1, which will demonstrate Algorithm 1's effectiveness. To do so, we first generate different training datasets $\mathcal{T}$ using the upsampling rate $\hat{C}$ fixed to 100 and different values

[2]Pre-trained ResNet101 and VGG16: https://keras.io/api/applications/

Algorithm 1 Generation of the training dataset $\mathcal{T}$ for $\mathrm{MLP_L}$

Input: Dataset $\mathcal{T}_0$, empty dataset $\mathcal{T}$, upsampling rate $\hat{C} \in \mathbb{N}$, the number of attributes manipulated $\hat{K}$, $\beta_0 > 0$ and $\beta_1 < 0$.
Output: Training dataset $\mathcal{T}$ ▷ Generated training dataset to train $\mathrm{MLP_L}$
Get the values of C and K from the dataset $\mathcal{T}_0$
for $i = 1$ to $\hat{C}$ **do**
  for $k = 1$ to C **do**
    `Get the current sample` $(\tilde{\boldsymbol{y}}_k, y_k) \in \mathcal{T}_0$
    Set $\tilde{\boldsymbol{y}}_{k,i} = \tilde{\boldsymbol{y}}_k$
    for $j = 1$ to $\hat{K}$ **do**
      `Generate a random number` $t \in \{1, \cdots, K\}$ ▷ Attribute value to change
      if $\tilde{y}^t_{k,i} \leq 0$ **then**
        $\tilde{y}^t_{k,i} = \beta_0$
      else
        $\tilde{y}^t_{k,i} = \beta_1$
      end if
    end for
    Add $(\tilde{\boldsymbol{y}}_{k,i}, y_k)$ into $\mathcal{T}$
  end for
end for
return Dataset $\mathcal{T}$

of $\hat{K}$ (i.e., the number of attributes whose values are manipulated) ranging from 0 to 28 (i.e., up to a third of the total 85 attributes per class in the AwA2 dataset). Then we train $\mathrm{MLP_L}$ using the generated training datasets individually. In detail, after the upsampling process in Algorithm 1, the obtained $\mathcal{T}$ is the size of $5,000$ (*cf.* the original given number of samples is 50 in the AwA2 dataset). To evaluate the accuracy performance of $\mathrm{MLP_L}$, we split each dataset $\mathcal{T}$ into two parts with the ratio of 70/30 for training and validation, respectively. The original dataset $\mathcal{T}_0$ is used for test.

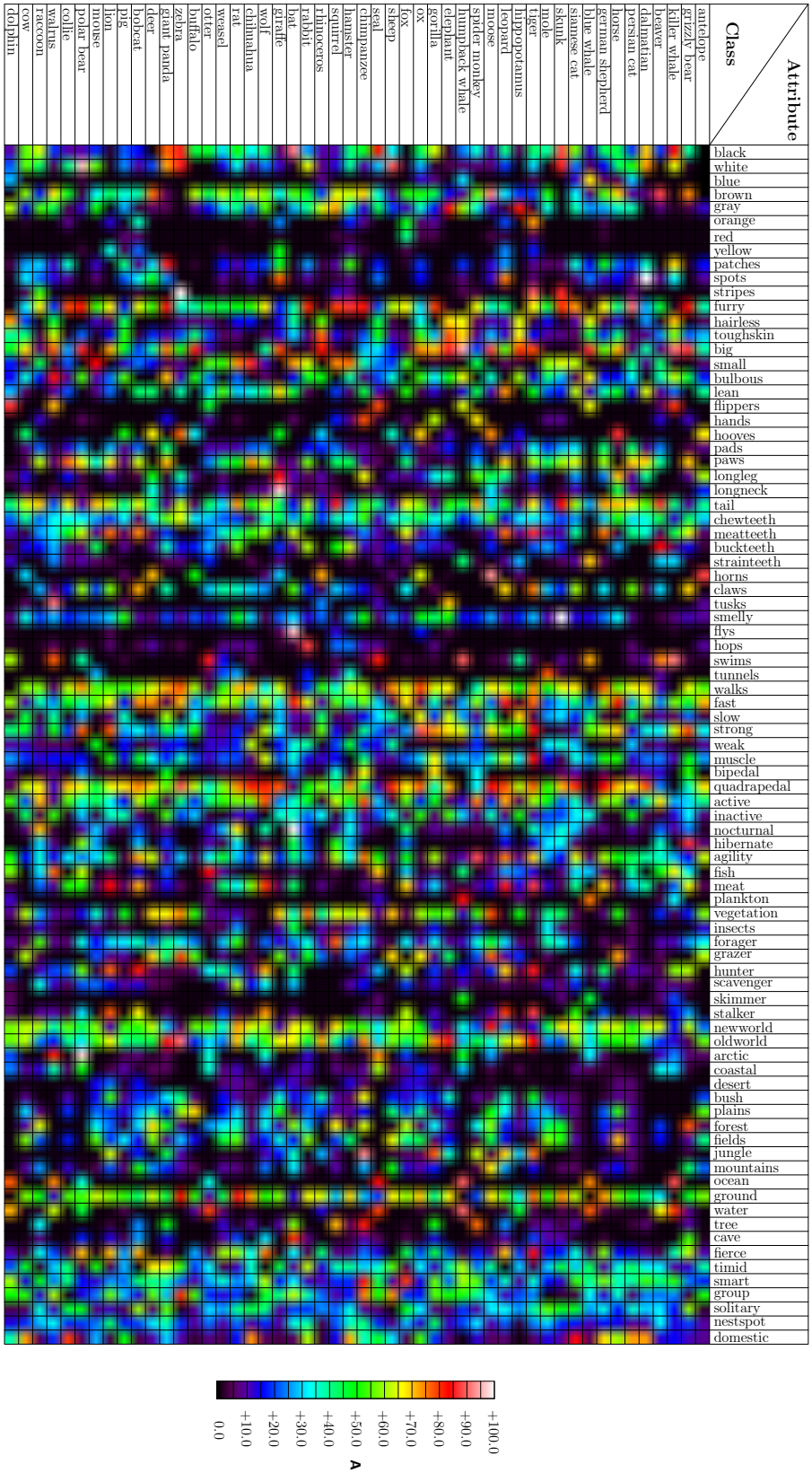
Table 3: Classification accuracy of $\mathrm{MLP_L}$ using the generated training datasets $\mathcal{T}$ by Algorithm 1 with different $\hat{K}$ on the AwA2 dataset. Note again that $\hat{K}$ represents the number of attributes whose values are manipulated per class against the ground-truth attribute-class matrix.

$\hat{K}$	Training	Validation	Test
0	100%	100%	100%
4	99%	99%	100%
8	97%	97%	98%
⋮	⋮	⋮	⋮
20	95%	93%	95%
28	93%	91%	90%

Table 3 shows the classification accuracy of $\mathrm{MLP_L}$ corresponding to different $\hat{K}$ in terms of training, validation and test. We see that the accuracy decreases when $\hat{K}$ becomes larger, which is quite reasonable, since the larger the $\hat{K}$, the higher the perturbation of the ground-truth attribute-class samples. Overall, satisfying accuracy (above 90%) is obtained by all different $\hat{K} \leq 28$. It is worth mentioning that 100% accuracy is achieved for $\hat{K} = 0$, which is because in this case the samples in the training set are the same as the ones in the test, i.e., the original dataset $\mathcal{T}_0$.

In our XAI experiments, $\hat{K}$ is fixed to 8.

Table 4: Full attribute-class matrix **A** for the AwA2 dataset.



C APPENDIX

C.1 FURTHER EXAMPLES FOR LEARNT ATTRIBUTES

As discussed in Section 4, some attributes can be well captured by DNNs with examples shown in Figures 1, 3 and 5 in the main text. To further demonstrate this property, we below present more images from a variety of classes, showing that the presented attributes are indeed learnt rather than special to the given images or classes.

For example, the attribute-wise saliency maps in Figure 7 show that the attribute *horns* is clearly learnt for the Ox, Antelope and Buffalo classes; for more results see Figures 8 and 9.

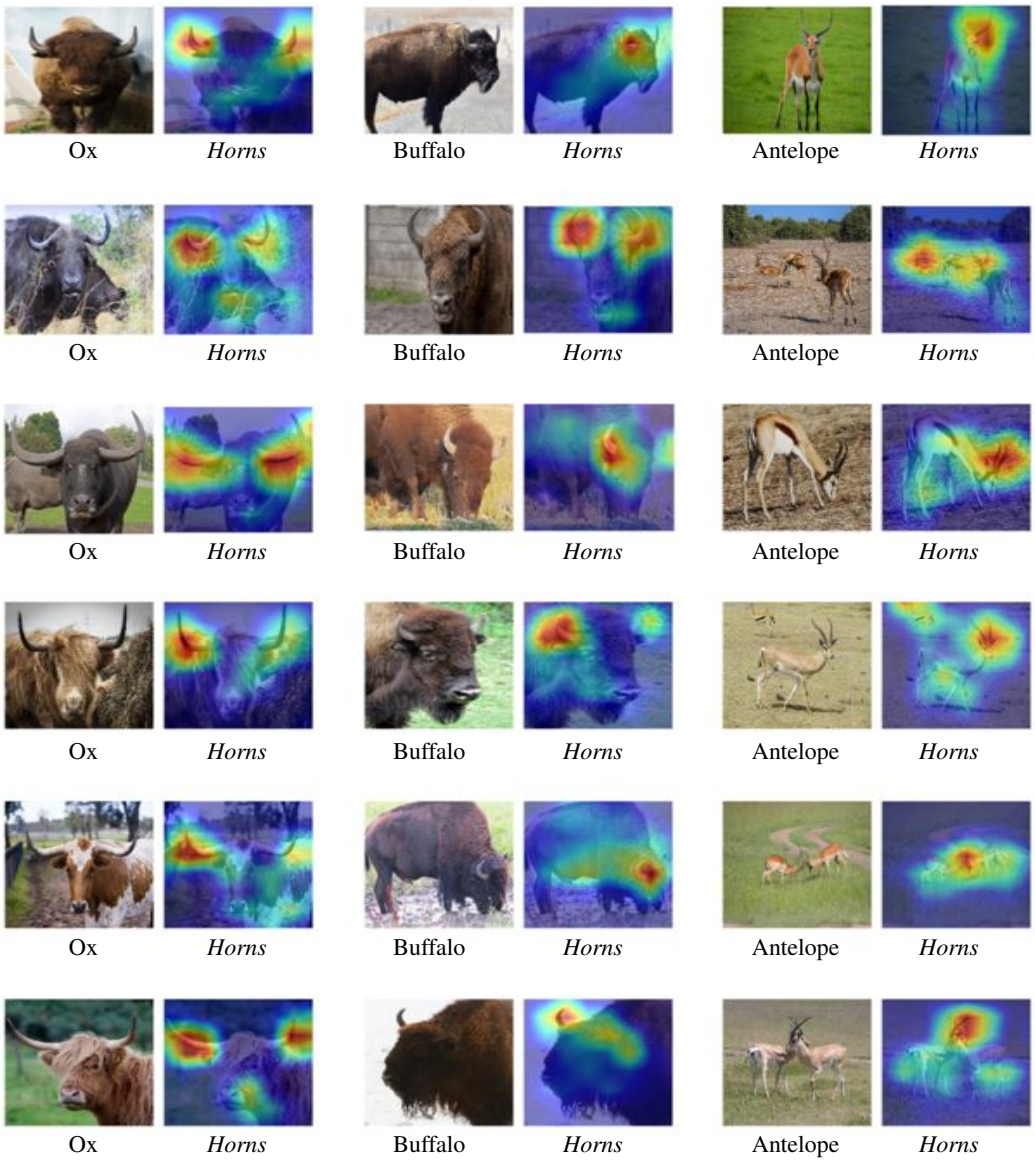


Figure 7: Images from different classes in the AwA2 dataset with their obtained attribute-wise saliency maps by our approach. Examples from Ox, Buffalo and Antelope classes show that the attribute *horns* is well captured by X-CNN.

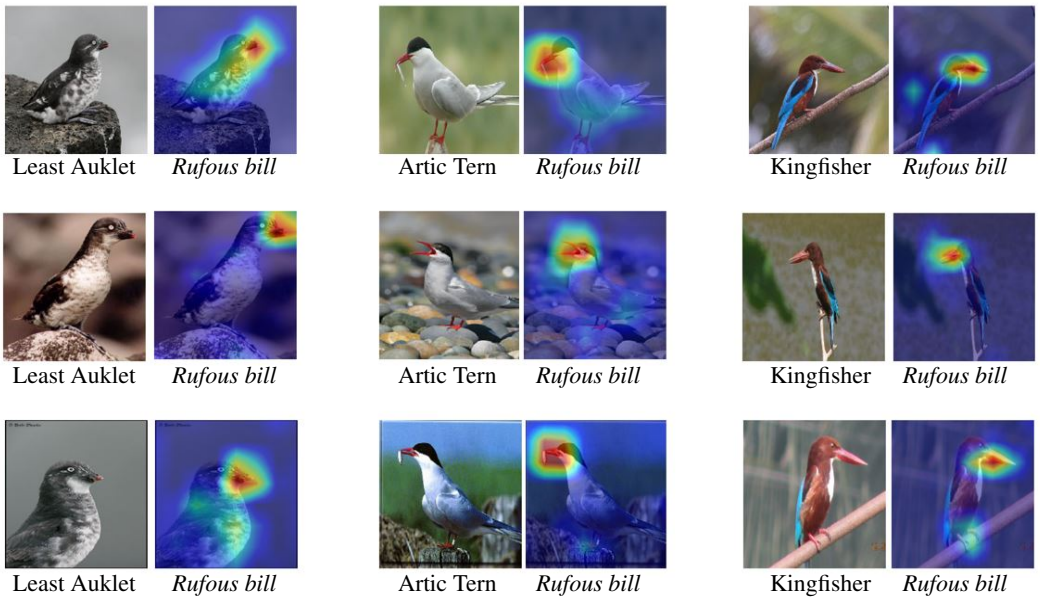


Figure 8: Images from different classes in the CUB dataset with their obtained attribute-wise saliency maps by our approach. Examples from Least Auklet, Artic Tern and Kingfisher classes show that the attribute *rufous bill* is well captured by X-CNN.

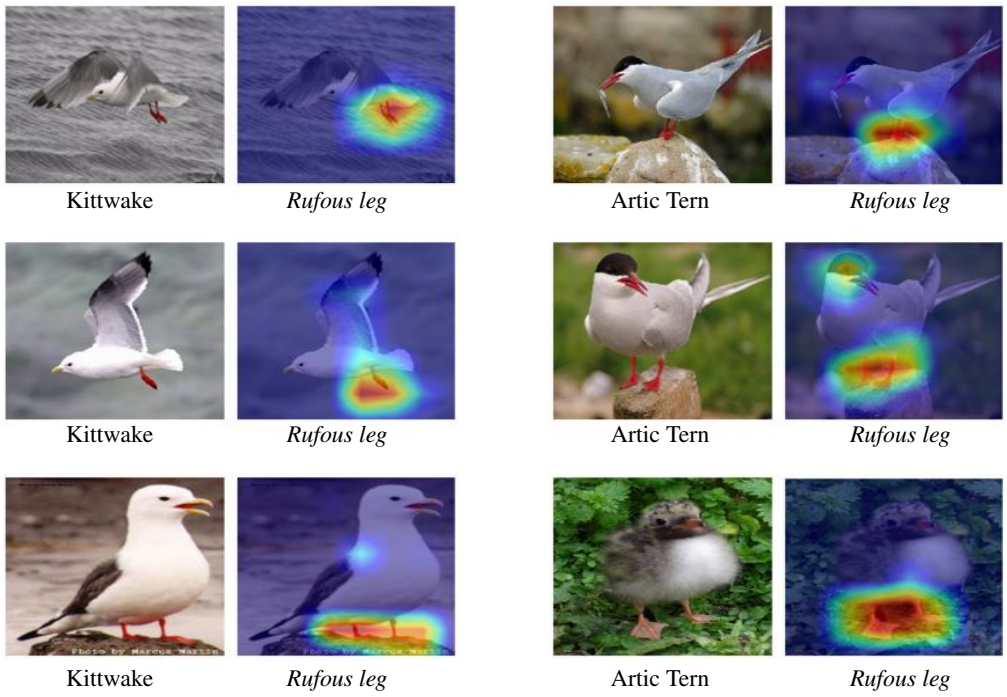


Figure 9: Images from different classes in the CUB dataset with their obtained attribute-wise saliency maps by our approach. Examples from Kittiwake and Artic Tern classes show that the attribute *rufous leg* is well captured by X-CNN.

C.2 EXPLAINABILITY FOR CORRECT CLASS PREDICTION

Further examples for correct class prediction with the multilevel explainability by our approach are shown in Figures 10 and 11. For abstract attributes, their saliency maps could give us an idea of what part of the image activates that specific attributes, e.g. active or weak. Moreover, the attribute values given by experts in Table 4 for the predicted classes indicate whether experts think these attributes are helpful to discriminate one class from the others. After checking, we can see these salient attributes obtained by our approach for the predicted classes are indeed meaningful.

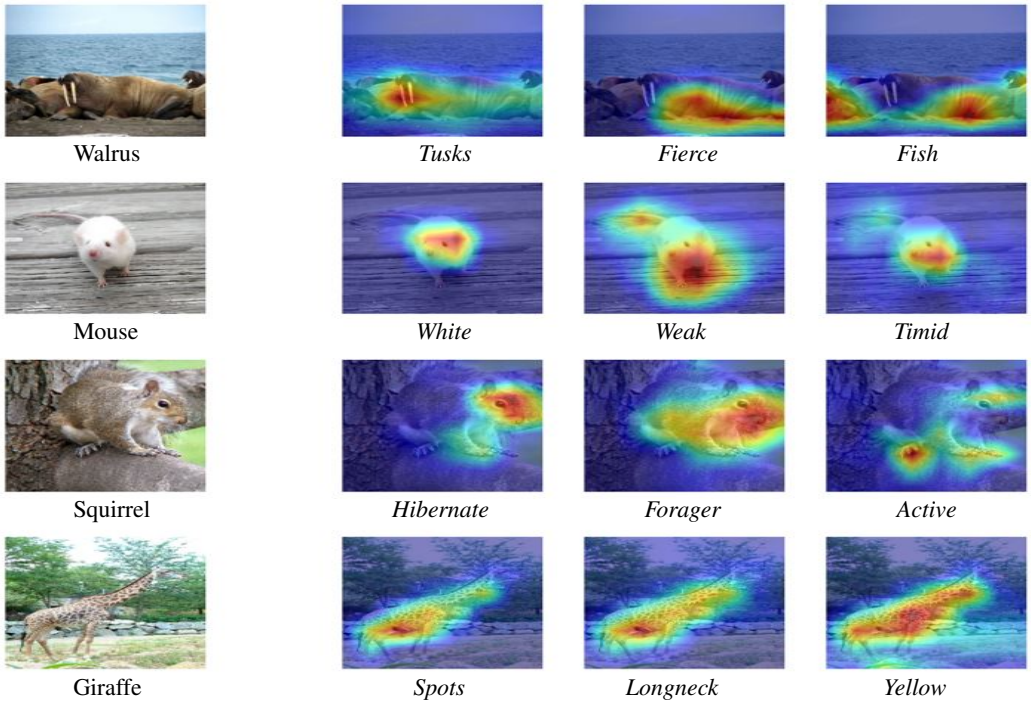


Figure 10: Explainability of the proposed approach for correct class prediction. Left: randomly selected test images in the AwA2 dataset. Right: the top three most salient attributes helping the neural network make the correct classification, and the corresponding attribute-wise saliency maps.

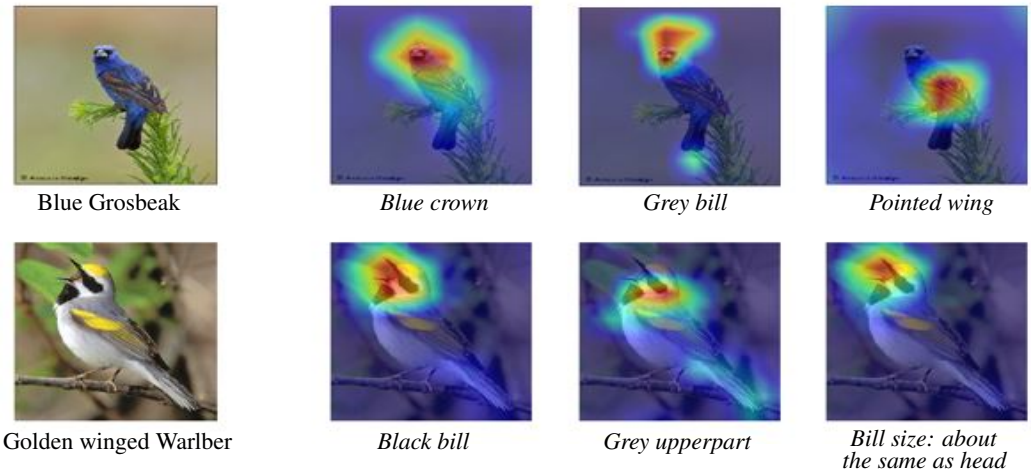


Figure 11: Explainability of the proposed approach for correct class prediction. Left: randomly selected test images in the CUB dataset. Right: the top three most salient attributes helping the neural network make the correct classification, and the corresponding attribute-wise saliency maps.

C.3 EXPLAINABILITY FOR INCORRECT CLASS PREDICTION

Further examples for incorrect class prediction with the multilevel explainability by our approach are shown in Figures 12 and 13. Grizzly bear classified as polar bear and whale classified as dolphin are some of the most frequent misclassification cases detected. After checking the attribute values in Table 4 given by the experts for the predicted classes and the ground-truth classes, we can see these salient attributes obtained by our approach are indeed consistent with the ones given by experts for the predicted classes; see also more discussion in Section 4 for the challenges in e.g. the linguistic alignment and the nature of explainability.

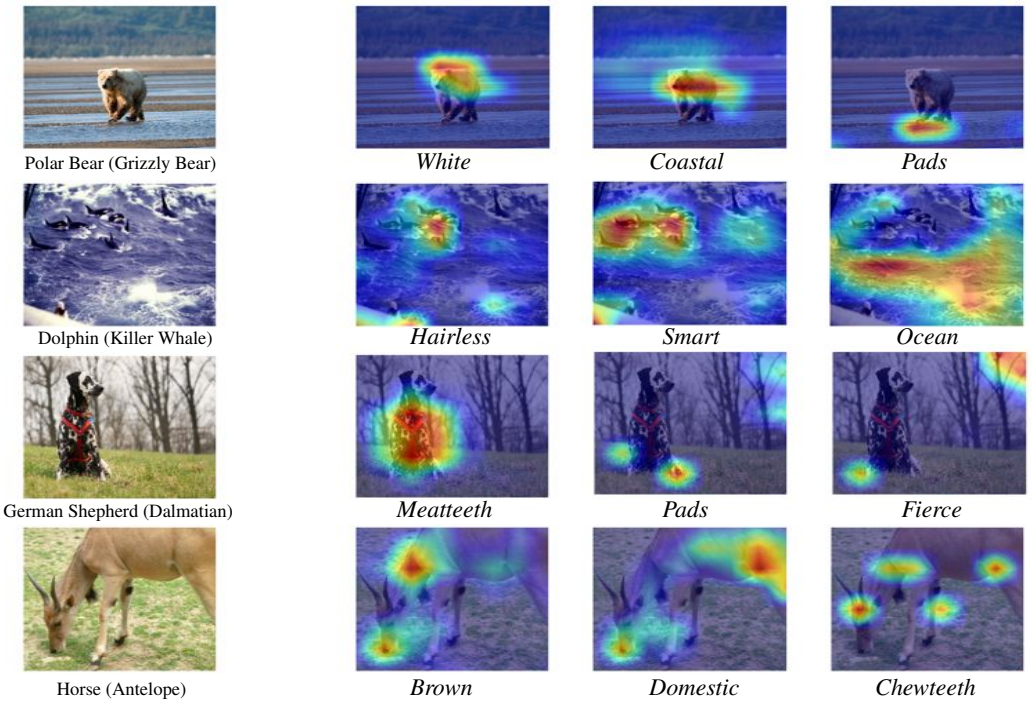


Figure 12: Explainability of the proposed approach for incorrect class prediction. Left: randomly selected test images in dataset AwA2; e.g., Polar Bear (Grizzly Bear) means the Grizzly Bear class is incorrectly predicted to be Polar Bear. Right: the top three most salient attributes helping the neural network make the incorrect classification, and the corresponding attribute-wise saliency maps.

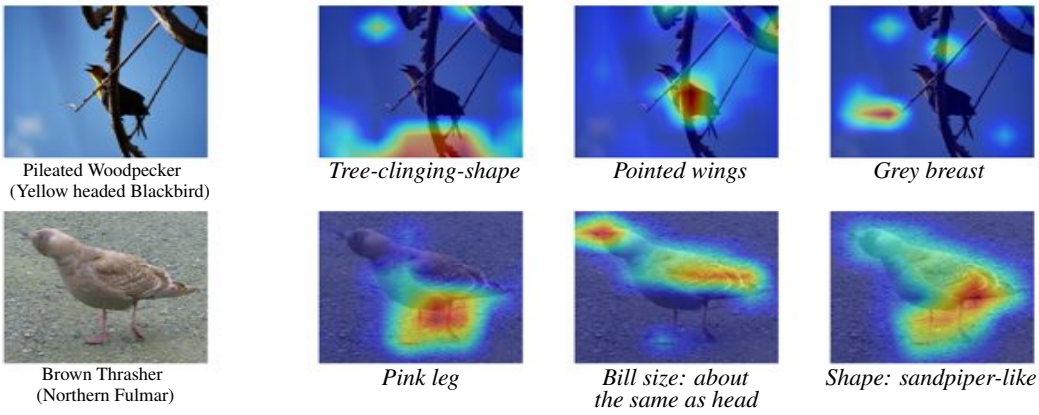


Figure 13: Explainability of the proposed approach for incorrect class prediction. Left: randomly selected test images in dataset CUB. Right: the top three most salient attributes helping the neural network make the incorrect classification, and the corresponding attribute-wise saliency maps.

C.4 INFORMATION OF LINGUISTIC ATTRIBUTES

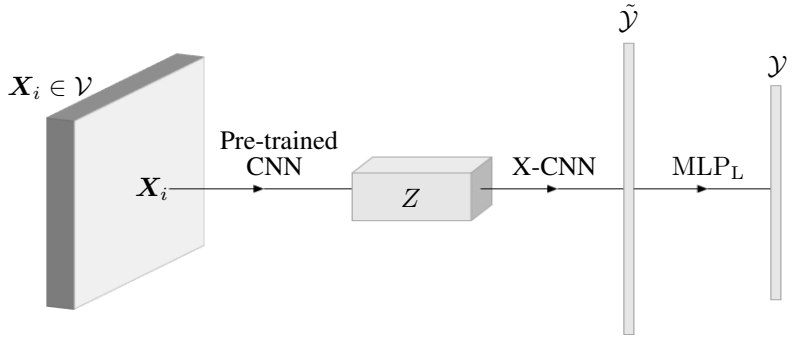


Figure 14: A simplified version of the proposed multilevel XAI architecture. An image $\boldsymbol{X}_i$ with a class label in $\mathcal{Y}$ is drawn from the dataset $\mathcal{V} \subseteq \mathcal{X}$. It is then passed to a pre-trained CNN. After passing through the last CNN layer, Z, it is passed to X-CNN, which produces a linguistic representation in $\tilde{\mathcal{Y}}$. Finally, it is passed to the $\mathrm{MLP_L}$ with a softmax output, giving a predicted label in $\mathcal{Y}$ for $\boldsymbol{X}_i$.

We now study the information of linguistic attributes, which will give us some guidance about the relationship between each class and its attributes.

For a set of classes $\mathcal{Y}$ (in our case the 50 species in the dataset AwA2), let Y be a random variable describing our classes, and define the probability of a class having class label $y \in \mathcal{Y}$ as $\mathbb{P}[Y = y]$. Then the entropy (or uncertainty) in the class label is

$$H_Y = -\sum_{y \in \mathcal{Y}} \mathbb{P}[Y = y] \log(\mathbb{P}[Y = y]). \tag{5}$$

To compute the mutual information for a linguistic attribute, $\tilde{y}^k$, we need to know $\mathbb{P}\left[Y \mid \tilde{y}^k\right]$, which can be estimated from a proper image set $\mathcal{V} \subseteq \mathcal{X}$. If we choose an image $\boldsymbol{X}_i \in \mathcal{V}$, we can feed it in our network (i.e., see Figure 14) to find its attribute value, $\tilde{y}^k_{\boldsymbol{X}_i}, 1 \leq k \leq K$. To simplify the calculation, let us set a threshold on the attribute value so that if it is above the threshold we treat $\tilde{y}^k_{\boldsymbol{X}_i} = 1$, otherwise $\tilde{y}^k_{\boldsymbol{X}_i} = 0$. For each $y \in \mathcal{Y}$ and each $\alpha \in \{0, 1\}$, we can estimate various probabilities, e.g.,

$$\begin{aligned} \mathbb{P}\left[Y = y, \tilde{y}^k = \alpha\right] &\approx \frac{\sum_{\boldsymbol{X}_i \in \mathcal{V}} [\![\boldsymbol{X}_i \text{ is from class } y]\!] \, [\![\tilde{y}^k_{\boldsymbol{X}_i} = \alpha]\!]}{|\mathcal{V}|}, \\ \mathbb{P}\left[\tilde{y}^k = \alpha\right] &\approx \frac{\sum_{\boldsymbol{X}_i \in \mathcal{V}} [\![\tilde{y}^k_{\boldsymbol{X}_i} = \alpha]\!]}{|\mathcal{V}|}, \end{aligned} \tag{6}$$

where $[\![\text{predicate}]\!]$ is an indicator function (equal to 1 if the predicate is true and 0 otherwise). We can then compute

$$\mathbb{P}\left[Y = y \mid \tilde{y}^k = \alpha\right] = \frac{\mathbb{P}\left[Y = y, \tilde{y}^k = \alpha\right]}{\mathbb{P}\left[\tilde{y}^k = \alpha\right]}. \tag{7}$$

The conditional entropy of the classes given the (binarised) linguistic attribute is given by

$$H_{Y|\tilde{y}^k} = -\sum_{\tilde{y}^k \in \{0,1\}} \sum_{y \in \mathcal{Y}} \mathbb{P}\left[Y = y, \tilde{y}^k = \alpha\right] \log\left(\mathbb{P}\left[Y = y \mid \tilde{y}^k = \alpha\right]\right). \tag{8}$$

The mutual information on the classes due to the linguistic attributes is given by

$$I\left(Y; \tilde{y}^k\right) = H_Y - H_{Y|\tilde{y}^k}. \tag{9}$$

Calculating the mutual information by attributes $\tilde{y}^k$ predicted by our approach and Eqn (9) gives us how important each attribute is in differentiating the classes, see Table 5. Some of them seem useless by themselves as seen in Table 5; however, they could be important in combination with others.

An alternative way to calculate the mutual information is by following the same steps above but obtaining $\tilde{y}^k$ values from the attribute-class matrix given in Table 4 instead of dataset $\mathcal{V}$. These alternative mutual information values represent the importance of each attribute for human experts who created Table 4. The calculated alternative mutual information is also presented in Table 5.

We now have two mutual information values per attribute, i.e., one is by using the dataset $\mathcal{V}$ (here we use the test set) and the other is by using the prior knowledge given in Table 4, see Table 5. Comparison between these two values for each attribute is helpful to see the discrepancy between what attributes people think are important and the ones that our trained DNNs learn. To give an example, "small" reduces the uncertainty by 0.98 bits when using the table values given by experts. However, it takes the value of 0.13 when using images, which suggests that "small" is not one of the best attributes for the trained DNN to differentiate the given classes, although people think it is.

Table 5: Mutual information of attributes calculated by using test images and the prior knowledge given by experts in Table 4, respectively. Symbol *** indicates the value is less than 0.01.

Attributes \ Info. via	*Test images*	*Experts*	Attributes \ Info. via	*Test images*	*Experts*
Black	0.38	0.95	*Muscle*	0.31	0.98
White	0.10	0.99	*Bipedal*	***	0.63
Blue	0.33	0.40	*Quadrupedal*	0.08	0.58
Brown	0.49	0.92	*Active*	***	0.82
Gray	0.02	0.99	*Inactive*	0.49	0.99
Orange	0.09	0.40	*Nocturnal*	0.43	0.88
Red	0.52	0.14	*Hibernate*	0.49	0.82
Yellow	0.40	0.40	*Agility*	0.42	0.92
Patches	0.14	0.88	*Fish*	0.09	0.92
Spots	***	0.79	*Meat*	0.46	0.97
Stripes	0.53	0.40	*Plankton*	0.35	0.32
Furry	***	0.76	*Vegetation*	0.23	0.99
Hairless	0.54	0.82	*Insects*	0.36	0.40
Toughskin	***	0.99	*Forager*	0.30	0.99
Big	0.34	0.95	*Grazer*	0.38	0.92
Small	0.13	0.98	*Hunter*	***	0.92
Bulbous	0.21	0.99	*Scavenger*	0.53	0.52
Lean	0.50	1	*Skimmer*	0.69	0.24
Flippers	0.15	0.58	*Stalker*	***	0.72
Hands	0.15	0.32	*Newworld*	0.30	0.68
Hooves	0.16	0.79	*Oldworld*	0.45	0.52
Pads	0.19	0.88	*Arctic*	0.07	0.68
Paws	0.51	0.99	*Coastal*	0.08	0.63
Longleg	0.50	0.85	*Desert*	0.59	0.14
Longneck	0.29	0.46	*Bush*	0.48	0.76
Tail	0.17	0.76	*Plains*	0.21	0.97
Chewteeth	***	0.76	*Forest*	0.04	0.98
Meatteeth	0.60	0.99	*Fields*	0.37	0.95
Buckteeth	***	0.79	*Jungle*	0.52	0.76
Strainteeth	0.38	0.52	*Mountains*	0.35	0.79
Horns	0.14	0.63	*Ocean*	0.39	0.63
Claws	0.15	0.98	*Ground*	***	0.68
Tusks	0.18	0.32	*Water*	0.14	0.72
Smelly	***	0.99	*Tree*	0.29	0.68
Flys	0.35	0.14	*Cave*	0.27	0.40
Hops	0.36	0.32	*Fierce*	0.25	0.98
Swims	0.32	0.72	*Timid*	***	0.92
Tunnels	0.14	0.46	*Smart*	0.51	0.90
Walks	0.57	0.72	*Group*	0.29	0.97
Fast	0.06	0.63	*Solitary*	0.31	0.98
Slow	0.55	0.97	*Nestspot*	0.20	0.97
Strong	0.24	0.90	*Domestic*	***	0.94
Weak	0.14	0.72			