# This Looks Like That... Does it? Shortcomings of Latent Space Prototype Interpretability in Deep Networks

**Adrian Hoffmann** [*1] **Claudio Fanconi** [*2] **Rahul Rade** [*2] **Jonas Kohler** [1]


## Abstract

Deep neural networks that yield human interpretable decisions by architectural design have lately become an increasingly popular alternative to post hoc interpretation of traditional black-box models. Among these networks, the arguably most widespread approach is so-called prototype learning, where similarities to learned latent prototypes serve as the basis of classifying an unseen data point. In this work, we point to an important shortcoming of such approaches. Namely, there is a semantic gap between similarity in latent space and similarity in input space, which can corrupt interpretability. We design two experiments that exemplify this issue on the so-called ProtoPNet. Specifically, we find that this network's interpretability mechanism can be led astray by intentionally crafted or even JPEG compression artefacts, which can produce incomprehensible decisions. We argue that practitioners ought to have this shortcoming in mind when deploying prototype-based models in practice.


## 1. Introduction

Due to the increasingly widespread application of artificial neural networks in science, industry and society, explaining decisions made by these models is becoming more and more important (Samek & Müller, 2019; Rudin, 2019). While many works in this regard have focused on ex-post interpretation of black-box machine learning models (e.g. Bach et al. (2015); Kim et al. (2018); Bau et al. (2017); Lapuschkin et al. (2019)), a recent line of research breaks with this paradigm by proposing to develop network architectures that are inherently interpretable by design (e.g. Girdhar & Ramanan (2017); Chen et al. (2019); Li et al. (2018); Brendel & Bethge (2019)). These models aim at basing their decisions on a small number of human-understandable features that allow users to comprehend how predictions come about while keeping compromises in terms of predictive performance low (Chen et al., 2019).

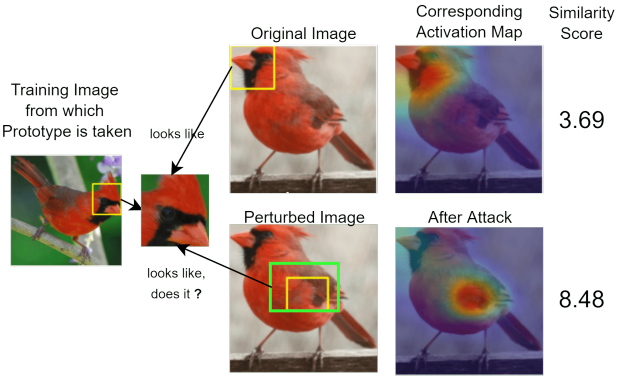



*Figure 1.* Example where adding crafted, human-imperceptible noise to an image (location indicated by the green box) breaks the human-interpretability of a properly trained ProtoPNet in the sense that it assigns prototypes to (for human eyes) arbitrary regions.

In this work, we focus on prototype-based reasoning which, unlike traditional deep classification networks, labels datapoints based on distances to learned prototypes of each class in latent space (Li et al., 2018; Chen et al., 2019; Gee et al., 2019; Ming et al., 2019; Xu et al., 2020). While these methods undoubtedly produce interpretable decisions in many cases, we here point to an important limitation of such approaches:

> The fact that two datapoints have a similar latent representation does not necessarily entail that they share similarities in terms of human-interpretable features in the input space.

In the remainder of this paper, we substantiate this claim by closely investigating the performance of an image classification network termed *Prototypical Part Network* (ProtoPNet). This network is proclaimed to yield a transparent and human-like decision process by classifying images based on similarity scores between (parts of) the test image and (parts of) the training images, called prototypes, in latent space (Chen et al., 2019). To assess these claims, we first argue that,

[*]Equal contribution [1]Department of Computer Science, ETH Zurich, Switzerland [2]Department of Information Technology and Electrical Engineering, ETH Zurich, Switzerland. Correspondence to: Adrian Hoffmann <adriahof@ethz.ch>.

for the reasoning of ProtoPNet to be qualitatively similar to human reasoning, the following statement must hold.

**Equivalence 1.**

*Two image patches look similar to a ProtoPNet* $\iff$ *Two image patches look similar to a human*

In the following, we design two experiments, which demonstrate that neither of the two directions always holds: (i) We first show that small, human imperceptible perturbations in a given test image can make a properly trained ProtoPNet identify prototypes in arbitrary regions (for the human eye) of the input (Section 3). (ii) Secondly, we disprove the backward direction by revealing that simple JPEG compression can drastically lower similarity scores between image patches that look similar to a human before and after compression (Section 4).

## 2. Preliminaries

### 2.1. ProtoPNets: Intuition and Inner Workings

A ProtoPNet is comprised of three stages: (i) First, similar to classical deep neural architectures deployed in image classification (Krizhevsky et al., 2012), ProtoPNets make use of a convolutional backbone $f$ (e.g. ResNet (He et al., 2016)) to extract meaningful latent representations of the input images. Yet, these representations are not directly fed through a linear classification layer followed by a softmax. (ii) Instead, Chen et al. (2019) employ a prototype layer which computes $L^2$ norm-based similarities to learned, class-specific prototype patches in the latent space and then (iii) classify an unseen image by passing these similarity scores through a softmax classification layer.

After classification, image patches and prototypes with the highest similarity score can be visualized in input space in order to represent the network's decision in a human-perceptible way. As a result, ProtoPNets have some level of interpretability built directly into the architecture and the authors in Chen et al. (2019) even go so far as to attest their network a transparent, human-like reasoning process.

**Prototype layers.** Given an input image $\mathbf{x}$ and its latent representation $\mathbf{z} = f(\mathbf{x})$ of size $H \times W \times D$, ProtoPNet computes the squared $L^2$ distance between all pixels in $\mathbf{z}$ and all $m$ learnable prototypes $\mathbf{P} = \{\mathbf{p}_l\}_{l=1}^m$ of size $1 \times 1 \times D$. For each pixel $\mathbf{z}_k \in \mathbb{R}^D$ and prototype $\mathbf{p}_l \in \mathbb{R}^D$, these distances are converted into a monotonically decreasing similarity score

$$s(\mathbf{z}_k, \mathbf{p}_l) = \log\left(\frac{\|\mathbf{z}_k - \mathbf{p}_l\|_2 + 1}{\|\mathbf{z}_k - \mathbf{p}_l\|_2 + \epsilon}\right), \quad \epsilon > 0. \tag{1}$$

This results in $m$ heatmaps of the same spatial size as $\mathbf{z}$ which indicate where and how strongly a given prototype is present in $\mathbf{z}$. Since spatial relations are preserved in the convolutional backbone $f$, the heatmap can be up-sampled and visualized on the input image. For each prototype $\mathbf{p}_l$, simple max-pooling over the spatial dimensions yields the final similarity score $s_{\mathbf{p}_l}(\mathbf{z})$. Hence, the output of a Prototype layer is a vector of $m$ values, indicating the highest similarity of each prototype to *some* patch in the image.

Importantly, the prototypes are class-specific and learned employing a contrastive loss as a regularizer that incentivizes a semantically meaningful clustering structure in the latent space (see Chen et al. (2019), Section 2.2). Furthermore, as the final step of training, the prototypes $\mathbf{p}_l$ (learned via backpropagation) are projected onto the nearest neighbour $\mathbf{z}_k^i$ of all inputs $x^i$ of the prototype-specific class. This last step is crucial for the proclaimed interpretability, as every prototype is thereby being equated with the latent representation of a patch of one of the training images, in order to visualize prototypes in a human-perceptible way.

### 2.2. Projected Gradient Descent

We use an adapted version of projected gradient descent (PGD) (Madry et al., 2018) for our experiments in Section 3. PGD is commonly used in the context of adversarial attacks (Akhtar & Mian, 2018) where it generates a perturbed image from a correctly classified sample. The ultimate goal is that the resulting image is misclassified by the given model while keeping the perturbation below a threshold in a given norm (hence the use of *Projected* Gradient Descent). We here reuse the idea behind PGD but we do not share the goal of adversarial attacks i.e. the generated perturbation is not intended to change the classification of a sample. Instead, it aims to make the network produce absurd interpretations.

### 2.3. Compression and other artefacts

In the experiments of Section 4, we investigate the robustness of ProtoPNet's interpretation mechanism against common file format artefacts. Specifically, we study JPEG compression, which is a lossy image compression format that introduces compression artefacts (Pennebaker & Mitchell, 1992). We choose these artefacts for our experiment for their convenience. However, we suspect that our results, which indicate that ProtoPNets can fail to convey their reasoning, also apply to other variants of noise (like Rayleigh in MRI- or speckle in ultrasound imaging (Dietrich et al., 2008; Goyal et al., 2018)). While it has been noted before that neural networks are not robust to compression artefacts in classification settings (Hendrycks & Dietterich, 2019) we are unaware of any prior results on their impacts on prototype-based interpretability mechanisms.

### 2.4. Definition of Interpretability

Lastly, we briefly define the notion of interpretability used in this work, since we are unaware of a universally agreed-upon definition. The following wording taken from Gilpin et al. (2018) suits our understanding: "*for a system to be interpretable, it must produce descriptions that are simple enough for a person to understand using a vocabulary that is meaningful to the user.*" Additionally, we require that the description also has to be credible. A description can be simple and understandable, but if it does not match the model's internal reasoning there is no value in it for our purposes. We also believe that this definition mirrors the exposition that Chen et al. (2019) had in mind when arguing that ProtoPNet was human-interpretable.

### 2.5. Deriving the Equivalence

In this subsection, we motivate our claim that Equivalence 1 is necessary for a ProtoPNet to be human interpretable.

Chen et al. (2019) suggest that we can interpret a classification decision by looking at the prototypes which the ProtoPNet found to be most strongly present in the input. For a particular prototype, this means that a user looks at the prototypes visualisation (i.e. a patch of a training image) and the patch in the input image which the ProtoPNet finds most similar to the given prototype. This is indeed similar to how humans communicate their decisions. However, it is only meaningful *if* the other person finds two parts of an object similar for the same reasons. Otherwise, the decision of one person is not understandable to the other. Hence, without such a mutual understanding between a human and a ProtoPNet, displaying prototype similarities does not perse yield interpretability. Consequently, Equivalence 1 needs to hold for a ProtoPNet to be interpretable.
Indeed, if a human and a ProtoPNet find two image patches similar for different reasons, interpretability would require the human to make sense of the latent representations which are the ProtoPNet's basis for scoring similarity. Yet, these representations are the outputs of a convolutional backbone and can hence not be considered interpretable.

## 3. Head on Stomach Experiment

We begin by designing a simple experiment which shows that the left-to-right implication in Equivalence 1 does not always hold in ProtoPNets. Since these networks are (like most networks) not perfect classifiers, we restrict the evaluation to test images that are correctly classified before and even after the experiment.

The basic idea is that we take a correctly classified test image $\mathbf{x}$, choose one of the most activated prototypes, say $\mathbf{p}_l$, that the ProtoPNet identifies in a sensible location in $\mathbf{x}$, and then perturb $\mathbf{x}$ such that the network finds $\mathbf{p}_l$ with high similarity but in a different region than before (and hence in a nonsensical place). All the while keeping the perturbation noise very small such that humans cannot detect it. An illustrative example is depicted in Figure 1 where a ProtoPNet accurately reports a similarity between the head of the bird in the test image and a prototype that depicts a head of a bird of the same species. However, after adding human-imperceptible noise the ProtoPNet reports an even higher similarity between the prototype and the stomach of the bird (hence the name of this experiment).

### 3.1. Methodology

For a single run of this experiment, we consider a ProtoPNet with convolutional backbone $f$, a correctly classified test image $\mathbf{x}$, and a chosen prototype $\mathbf{p}_l$. Furthermore, $S$ denotes the set of pixels in the latent representation of $\mathbf{x}$ with the highest similarities to $\mathbf{p}_l$ and $S_{\text{noisy}}$ is the set of latent pixels we want to move the highest similarity to. Lastly, $g_{\mathbf{p}_l}(\mathbf{z})$ denotes the similarity map between prototype $\mathbf{p}_l$ and latent representation $\mathbf{z}$ where each entry in $g_{\mathbf{p}_l}(\mathbf{z})$ is computed with the similarity score in Equation 1. To achieve our goal, we formulate the following objective function:

$$\begin{aligned}\mathcal{L}(\mathbf{x}; S, S_{\text{noisy}}) = &\frac{1}{|S_{\text{noisy}}|} \sum_{(i,j) \in S_{\text{noisy}}} g_{\mathbf{p}_l}(f(\mathbf{x}))_{ij} \\ &- \frac{1}{|S|} \sum_{(i,j) \in S} g_{\mathbf{p}_l}(f(\mathbf{x}))_{ij}\end{aligned} \tag{2}$$

The objective encourages high similarity between the latent pixels at the positions in $S_{\text{noisy}}$ and prototype $\mathbf{p}_l$ while reducing similarity between the latent pixels at positions in $S$ and $\mathbf{p}_l$. This allows us to state the full optimization problem:

$$\min_{\delta \in \mathbb{R}^{\dim(\mathbf{x})}} -\mathcal{L}\left(\mathbf{x} + \delta; S, S_{\text{noisy}}\right) \quad \text{s.t. } \|\delta\|_\infty \leqslant \varepsilon \tag{3}$$

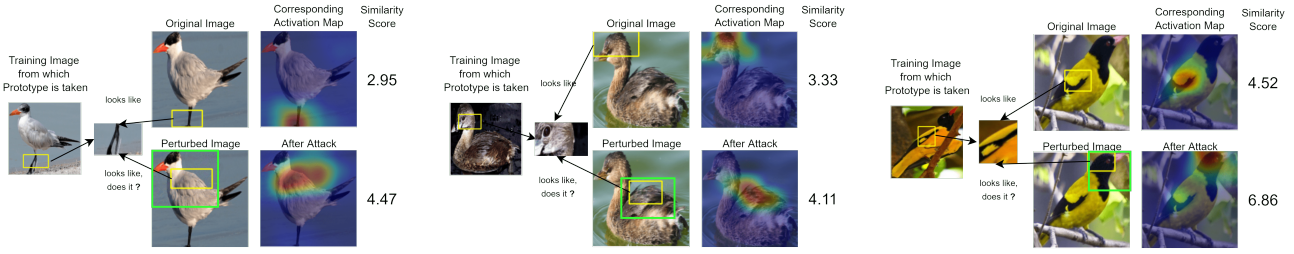
where the constraint makes sure that the noise stays small.

We would like to point out two subtle differences between the usual application of PGD (Madry et al., 2018) and our approach: First, as alluded to before, our goal is not to create an adversarial example for misclassification but we rather focus on how the similarity of the chosen prototype changes.[1] Secondly, we do not add noise to the entire image but only to those local patches relevant for the similarity scores.

### 3.2. Results

In total, we trained three ProtoPNets, one for each of ResNet-18, ResNet-34, and VGG-19 as backbone. For training we

[1] In fact, all noisy images were still correctly classified.




(a) ResNet-18 (b) ResNet-34 VGG-19

*Figure 2.* Three examples (one per backbone) for the Head on Stomach experiment. The first image is the source of the prototype we consider in the example. The next image shows the prototype itself. The top row of the remaining four images shows where the respective ProtoPNet finds the shown prototype in the input image (yellow rectangle and heat map). The bottom row shows where it recognises the prototype after we add the noise. The position of the added noise is shown by the green rectangle.

used the same dataset (CUB-200-2011 (Wah et al., 2011)), hyper-parameters, and data-augmentation as in the original ProtoPNet paper (Chen et al., 2019). We employ projected gradient descent with $\varepsilon = 8/255$, a fixed step size of $2/255$, and 10-40 iterations to solve the optimization problem.

Figure 2 depicts three examples of the Head on Stomach experiment (one for each backbone). As can be seen, the provided interpretations seem plausible on the clean images. However, when we perturb the image, the network's reasoning becomes clearly incomprehensible to humans. Additionally, the similarity scores for the noisy pictures are remarkably high (a similarity score $\geq 4$ is empirically very high, see plots in Appendix C.2). Additional examples can be found in Appendix B. Moreover, we report the accuracy of the models in Appendix D.

## 4. JPEG Experiment

We next show that the right-to-left implication in Equivalence 1 can break down in fairly realistic scenarios. Interestingly, our experiment also uncovers a potential gap between the interpretability and decision mechanism in ProtoPNets, which is important for practitioners to keep in mind.

JPEG compression is a lossy compression algorithm, which leaves artefacts behind. While this noise is usually barely perceptible for humans, it may still confuse neural networks in detrimental ways (Hendrycks & Dietterich, 2019). In the following, we examine changes in interpretability when training a ProtoPNet on a dataset where the training images of a few (but not all) classes are JPEG compressed, which might occur e.g. when collecting training data from different sources. In particular, we assess if the interpretation provided by the model differs depending on whether the compressed- or clean version of a test image is considered. As the noise is almost unrecognisable to humans, ProtoPNet would have to provide similar interpretations in both cases for the right-to-left implication in Equivalence 1 to hold.

### 4.1. Methodology

For this experiment, we again train on the CUB-200-2011 dataset. But first, we choose 100 classes at random (i.e. half of the classes) and apply JPEG compression with 20% quality to the images of these classes (see Fig. 7).

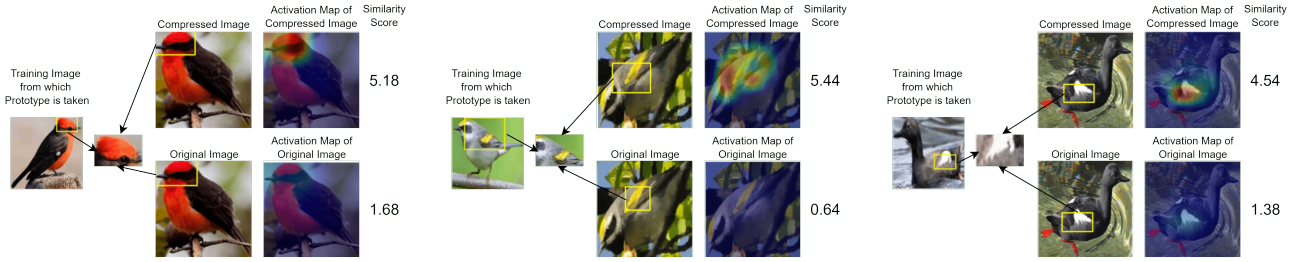
To assess whether the interpretations stay consistent for images with and without compression artefacts, we consider a given test image $\mathbf{x}$ of a compressed class and check if the compressed version of $\mathbf{x}$ (denoted $\bar{\mathbf{x}}$) is classified correctly. If so, we record the similarity score for the most activated prototype (denoted $\mathbf{p}_l$) for $\bar{\mathbf{x}}$ and track how the score of this prototype changes when we instead feed $\mathbf{x}$ into the network. If the scores for $\mathbf{p}_l$ differ significantly, then the ProtoPNet does not consider $\mathbf{x}$ and $\bar{\mathbf{x}}$ similar in the same manner as humans would, which in turn suggests that the right-to-left implication does not always hold.

### 4.2. Results

We train one ProtoPNet for each of the ResNet-18, ResNet-34, and VGG-19 convolutional backbones with the same training setup as in the Head on Stomach experiment, besides the aforementioned changes to the dataset.

In Figure 3, we display three instances of the JPEG experiment. As can be seen in the top row, the ProtoPNets detect strong similarities between prototypes and image patches in accordance with human reasoning when considering compressed images. However, the similarity scores drop significantly when we remove the compression artefacts (bottom row). This suggests that a) images that humans find very similar need not look similar to ProtoPNets and b) the compression artefacts themselves are important for the decision making of the network[2]. More examples can be found in App. C.1. Importantly, we note that switching from the compressed to the original test image does not simply uniformly scale down all similarity scores (see App. C.2).

[2]Which is supported by a drop in accuracy (see App. D)




(a) ResNet-18 (b) ResNet-34 (c) VGG-19

*Figure 3.* Three examples (one per backbone) for the JPEG experiment. In each example, the first image is the source of the considered prototype. The next image shows the prototype itself. To the right, the top (bottom) row shows a compressed (original) test image and its activation map. Similarity scores drop significantly if no JPEG compression is applied, despite the fact that the images look similar to the human eye, which highlights that ProtoPNet no longer considers the image patches similar.

## 5. Discussion

The results from the previous sections show that both directions of Equivalence 1 do not hold in general. Importantly, we only consider correctly classified images for each ProtoPNet and focus on prototypes with high similarity scores. Still, in the Head on Stomach experiment (Section 3) the trained models found strong similarities between image patches that humans cannot comprehend from looking at the images in input space. Furthermore, in the JPEG experiment (Section 4) we demonstrated that compressed images, which look very much like their originals to humans, are not considered alike by ProtoPNets. Finally, the JPEG experiment suggests that while compression artefacts can be important to a model for classification at test time, the interpretability mechanism in ProtoPNets cannot convey this to the user. Since practitioners might not be aware of the type and level of noise that is present in their data, it is important to keep in mind that there can be a difference between what the interpretability mechanism is able to convey and what information the model actually uses for its decision.

In summary, our findings suggest a gap between ProtoPNet's understanding of similarity and that of humans. While this is not severe in the case of bird images, where any flaw in interpretability is easily detected, we consider this lack of robustness to be much more critical in settings that humans are less well suited for evolutionarily, such as in medical imaging, where one might e.g. be interested in gaining insights about diseases via the neural network's decision process. Here, it is important for the network to accurately report the patterns it picks up on. While it is arguably true, that both adversarial and compression noise are known weak spots of neural networks, we want to emphasize that their application to interpretability mechanisms is novel. Furthermore, we consider our uncoverings as important because (a) none of the current interpretability definitions in the literature exempt the presence of noise and (b) this scenario is relevant as practitioners might not always be aware of the type and level of noise in their data, especially when collecting data from multiple sources.

Finally, we note that we do by no means dispute the general usefulness of prototype-based interpretation mechanisms but rather this paper is to be understood as a call for caution. In fact, as we detail in Appendix E, classical counter measures such as adversarial training and data augmentation do indeed increase the robustness of ProtoPNets albeit at the cost of reduced accuracy.

## 6. Summary and Future Work

In this work, we investigated potential shortcomings of the interpretability mechanism of so-called ProtoPNets, which base their decisions on similarities between image patches and visualisable prototypes. We showed that, while ProtoPNets usually bring about human-understandable decisions, there are certain cases, namely in the presence of adversarial and compression noise, where the analogy between human and network reasoning breaks down. While the first scenario is not overly surprising, the latter is of particular relevance to practitioners who might have different sources of noise in their data (compression, lighting, device artefacts, zoom and motion artefacts in cameras, etc.). We thus caution practitioners to be mindful of the aforementioned shortcomings when using prototypical networks, especially in settings where possible flaws in the interpretability mechanism are less obvious, like for example in medical imaging.

Going forward, we consider a more comprehensive study of the robustness of interpretability mechanisms to be an interesting direction of future research, including both the examination of interpretable architectures different to ProtoPNets (e.g. (Nakka & Salzmann, 2020)) as well as the identification of further sources of artefacts/noise that may lead to a break down of interpretability in real-world settings. Finally, building upon our initial results in Appendix E, further improving the robustness of interpretable models via architectural enhancement as well as advanced training techniques constitutes an interesting line of future work.

## References


Akhtar, N. and Mian, A. Threat of adversarial attacks on deep learning in computer vision: A survey. *IEEE Access*, 6:14410–14430, 2018.

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R., and Samek, W. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PLOS ONE*, 10(7):e0130140, 2015.

Bau, D., Zhou, B., Khosla, A., Oliva, A., and Torralba, A. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6541–6549, 2017.

Brendel, W. and Bethge, M. Approximating CNNs with bag-of-local-features models works surprisingly well on imagenet. In *International Conference on Learning Representations*, 2019.

Chen, C., Li, O., Tao, D., Barnett, A., Rudin, C., and Su, J. K. This looks like that: Deep learning for interpretable image recognition. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Dietrich, O., Raya, J., Reeder, S., Ingrisch, M., Reiser, M., and Schoenberg, S. Influence of multichannel combination, parallel imaging and other reconstruction techniques on mri noise characteristics. *Magnetic resonance imaging*, 26:754–62, 08 2008. doi: 10.1016/j.mri.2008.02.001.

Gee, A. H., García-Olano, D., Ghosh, J., and Paydarfar, D. Explaining deep classification of time-series data with learned prototypes. In *Proceedings of the 4th International Workshop on Knowledge Discovery in Healthcare Data co-located with the 28th International Joint Conference on Artificial Intelligence, KDH@IJCAI*, volume 2429 of *CEUR Workshop Proceedings*, pp. 15–22, 2019.

Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., and Kagal, L. Explaining explanations: An overview of interpretability of machine learning. In *2018 IEEE 5th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 80–89, 2018. doi: 10.1109/DSAA.2018.00018.

Girdhar, R. and Ramanan, D. Attentional pooling for action recognition. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Goyal, B., Dogra, A., Agrawal, S., and Sohi, B. Noise issues prevailing in various types of medical images. *Biomedical and Pharmacology Journal*, 11:1227–1237, 09 2018. doi: 10.13005/bpj/1484.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. *Proceedings of the International Conference on Learning Representations (ICLR)*, 2019.

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 *International Conference on Machine Learning*, pp. 2668–2677. PMLR, 2018.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25: 1097–1105, 2012.

Lapuschkin, S., Wäldchen, S., Binder, A., Montavon, G., Samek, W., and Müller, K.-R. Unmasking clever hans predictors and assessing what machines really learn. *Nature Communications*, 10(1):1–8, 2019.

Li, O., Liu, H., Chen, C., and Rudin, C. Deep learning for case-based reasoning through prototypes: A neural network that explains its predictions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

Madry, A., Makelov, A., Schmidt, L., Tsipras, D., and Vladu, A. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations (ICLR)*, 2018.

Ming, Y., Xu, P., Qu, H., and Ren, L. Interpretable and steerable sequence learning via prototypes. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 903–913, 2019.

Nakka, K. K. and Salzmann, M. Towards robust fine-grained recognition by maximal separation of discriminative features. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, November 2020.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, 2019.

Pennebaker, W. B. and Mitchell, J. L. *JPEG Still Image Data Compression Standard*. Van Nostrand Reinhold, New York, 1992.

Rudin, C. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1(5):206–215, 2019.

Samek, W. and Müller, K.-R. Towards explainable artificial intelligence. In *Explainable AI: interpreting, explaining and visualizing deep learning*, pp. 5–22. Springer, 2019.

Tsipras, D., Santurkar, S., Engstrom, L., Turner, A., and Madry, A. Robustness may be at odds with accuracy. In *International Conference on Learning Representations*, 2019.

Wah, C., Branson, S., Welinder, P., Perona, P., and Belongie, S. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Wong, E., Rice, L., and Kolter, J. Z. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations*, 2020.

Xu, W., Xian, Y., Wang, J., Schiele, B., and Akata, Z. Attribute prototype network for zero-shot learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21969–21980, 2020.

Zhang, H., Yu, Y., Jiao, J., Xing, E., Ghaoui, L. E., and Jordan, M. Theoretically principled trade-off between robustness and accuracy. In *Proceedings of the 36th International Conference on Machine Learning*, 2019.

## A. Implementation Details

We implemented the ProtoPNets, as well as our adversarial attacks using PyTorch (Paszke et al., 2019). We use exactly the same experimental setup as described in Chen et al. (2019) for our experiments. Particularly, we use 40x augmentation for each training image in the CUB-200-2011 dataset[3](Wah et al., 2011).

Following Chen et al. (2019), we set the number of channels in a prototype to 128 for ResNet-18 and VGG-19; and 256 for ResNet-34. Additionally, we set the prototype size to $1 \times 1$ and use 10 prototypes for each class. Finally, we simply reuse their training setup for ease of implementation and reproducibility. We set the weights of cluster and separation costs to 0.8 and -0.08 respectively, and choose the coefficient of the $L_1$ regularization as $0.0001$. For training, we set the number of warmup epochs to 5 and use Adam optimizer with an initial learning rate of $0.003$. Subsequently, we train the entire model for 6 more epochs with a learning rate of $0.0001$ for the backbone network and $0.003$ for the prototype layer. Lastly, we push the prototypes to the most similar latent training image patches and fine-tune the fully-connected layer for 20 iterations using Adam optimizer with a learning rate of $0.0001$. We select the best performing model on the test set for our experiments. The code to our experiments is openly available on GitHub.

## B. More Examples for The Head on Stomach Experiment

In this section, we present further examples from the Head on Stomach experiment which were successful. Figures 4, 5 and 6 illustrate the results for ResNet-18, ResNet-34 and VGG-19 respectively.

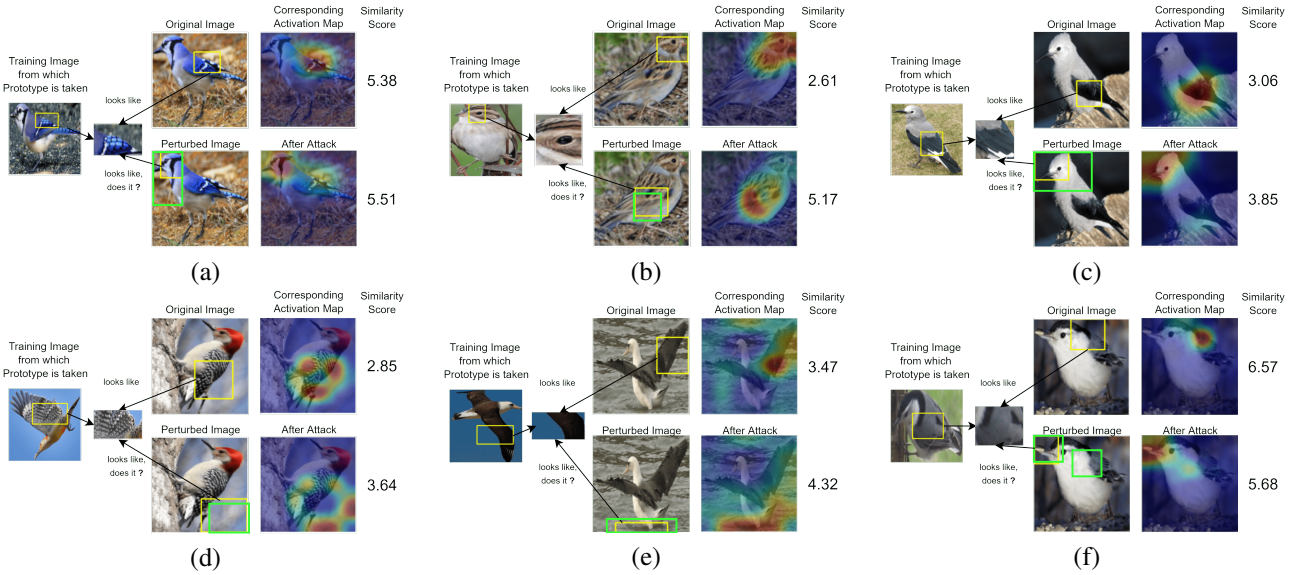



*Figure 4.* The Head on Stomach experiment - ResNet-18

[3]The CUB-200-2011 dataset can be downloaded from http://www.vision.caltech.edu/visipedia/CUB-200-2011.html.

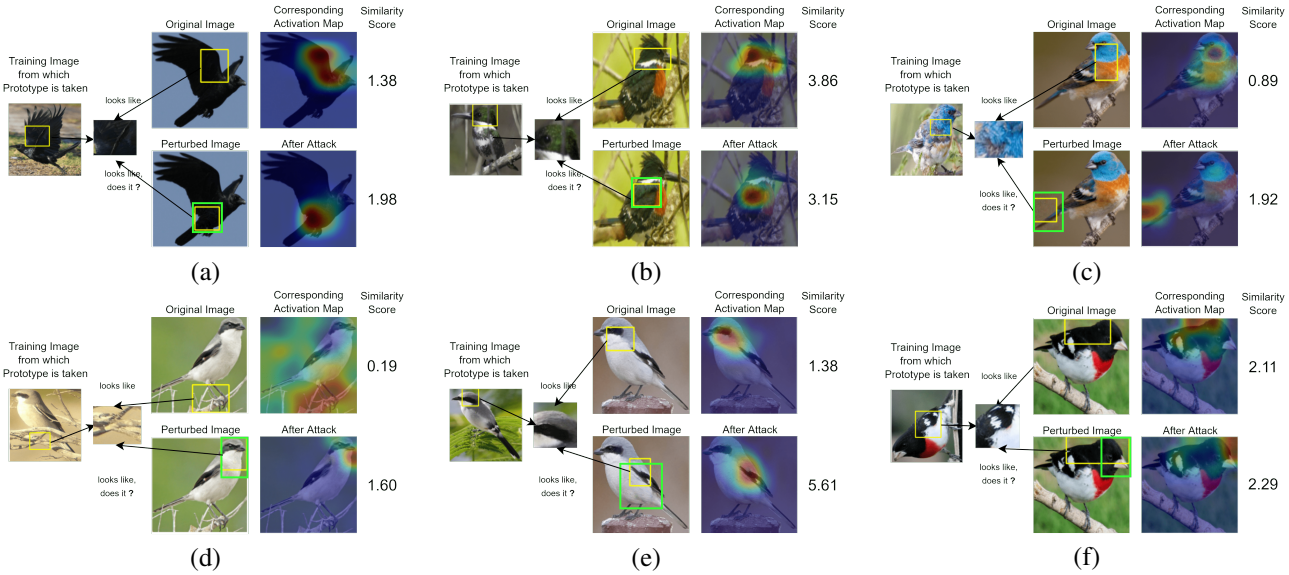



*Figure 5.* The Head on Stomach experiment - ResNet-34

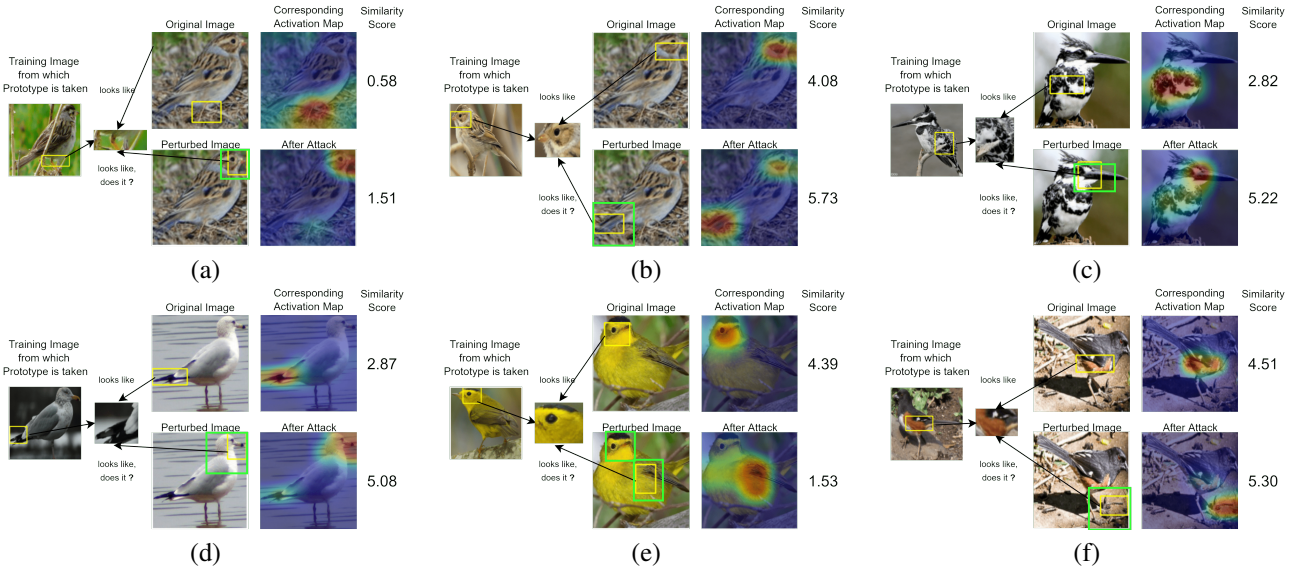



*Figure 6.* The Head on Stomach experiment - VGG-19

## C. Details on JPEG Experiment

original compressed

*Figure 7.* A random image from the the CUB-200-2011 dataset without and with compression artefacts respectively (JPEG compression quality is set to 20%)

### C.1. More Examples for The JPEG Experiment

In this subsection, we show further examples from the JPEG experiment which were successful. Figures 8, 9 and 10 present the results for ResNet-18, ResNet-34 and VGG-19 respectively.

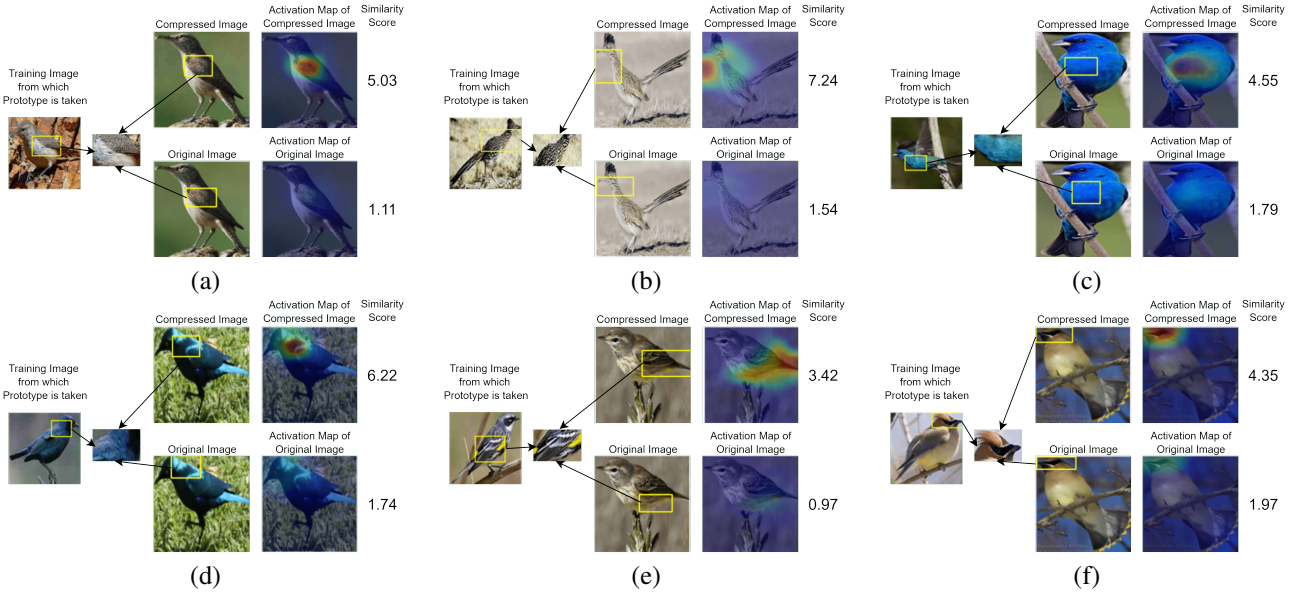



*Figure 8.* The JPEG experiment - ResNet-18

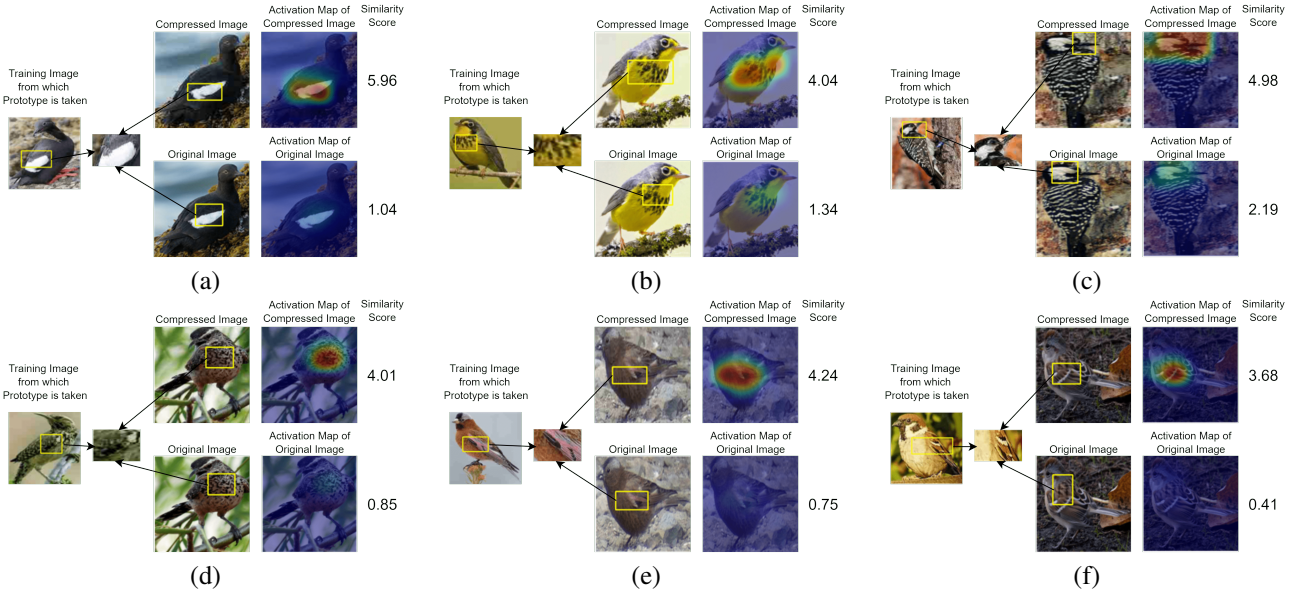



*Figure 9.* The JPEG experiment - ResNet-34

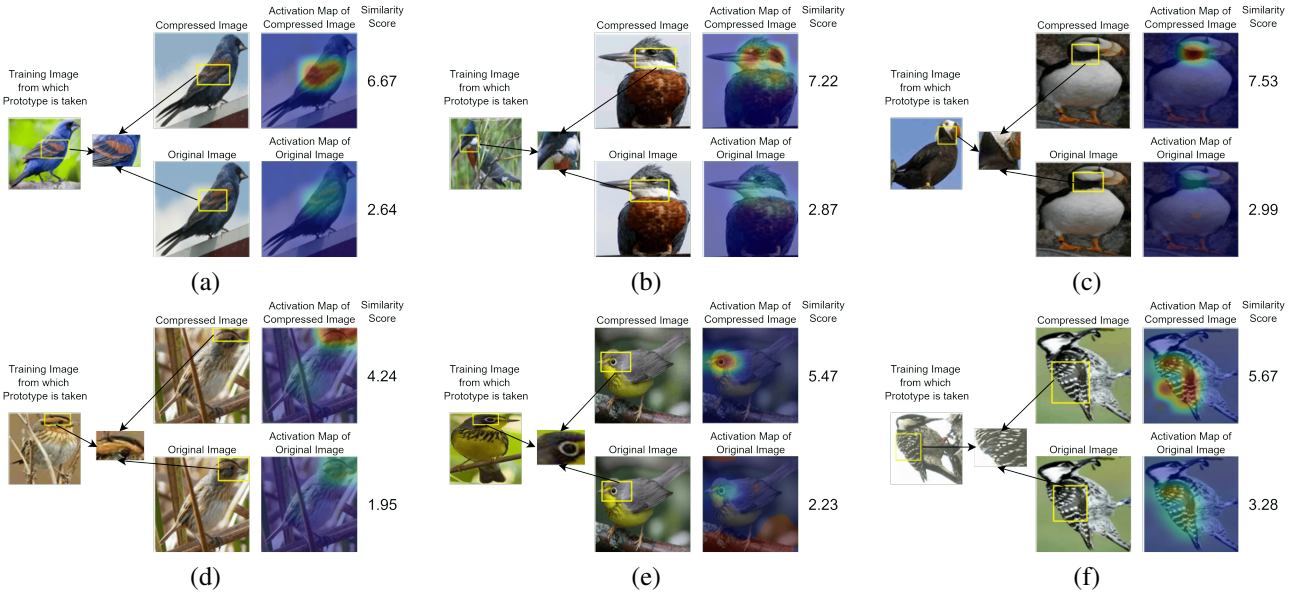



*Figure 10.* The JPEG experiment - VGG-19

### C.2. Histograms in JPEG Experiment

In Section 4 we showed that if $\mathbf{p}_l$ is the prototype with the highest similarity reported by a ProtoPNet for some test image $\bar{\mathbf{x}}$ with JPEG compression artefacts then the ProtoPNet might find a much lower similarity score for $\mathbf{p}_l$ if we remove the artefacts from $\bar{\mathbf{x}}$. This shows that ProtoPNets don't generally find compressed and clean versions of an image similar in the same manner as humans do. However, a ProtoPNet might uniformly scale down the similarity scores when switching from compressed to clean versions of an image. For example, if a ProtoPNet reports the highest similarity scores between a compressed test image $\bar{\mathbf{x}}$ and prototypes $\mathbf{p}_q$, $\mathbf{p}_r$, and $\mathbf{p}_s$ and does so again if we remove the noise then the ProtoPNet would consistently find the same prototypes to be most important. Which would be a very valuable characteristic of ProtoPNets. In Figures 11, 12, and 13 we investigate this. Each figure shows on the left the 75 similarity scores of the prototypes that are most strongly found by the ProtoPNet in an image $\bar{\mathbf{x}}$ (i.e. the JPEG compressed image) and overlaid the similarity scores of the respective prototypes but for image $\mathbf{x}$ (i.e. the clean version of $\bar{\mathbf{x}}$). The right parts of the plots show the other way round (i.e. the prototypes with the highest similarities to $\mathbf{x}$ are considered). The images and ProtoPNets we consider here are the same as in Figure 3.

As can clearly be seen in the figures, the top few prototypes do not generally remain the prototypes with the highest similarity scores once the artefacts are removed and vice versa. This suggests that ProtoPNets do not consider the same prototypes to be the most important for compressed and clean versions of an image.

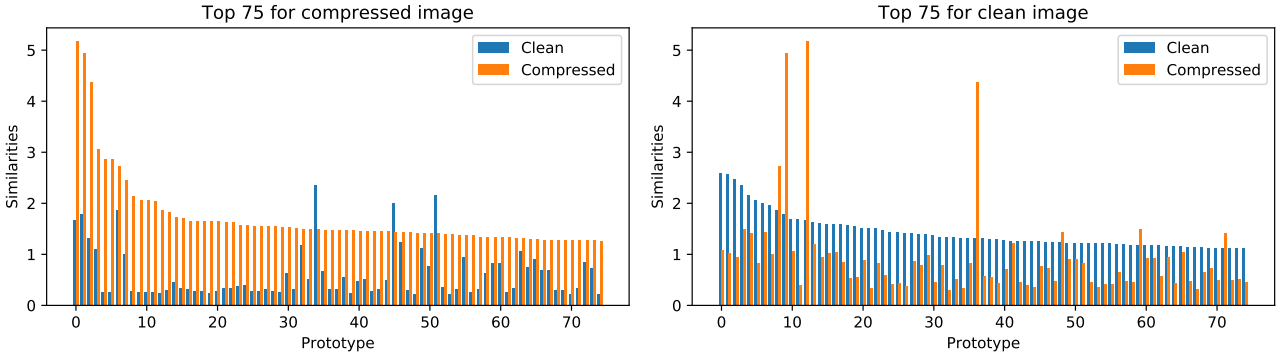



*Figure 11.* Highest similarities for the test image depicted in Figure 3 (a) on the ProtoPNet with the ResNet-18 backbone from Section 4.

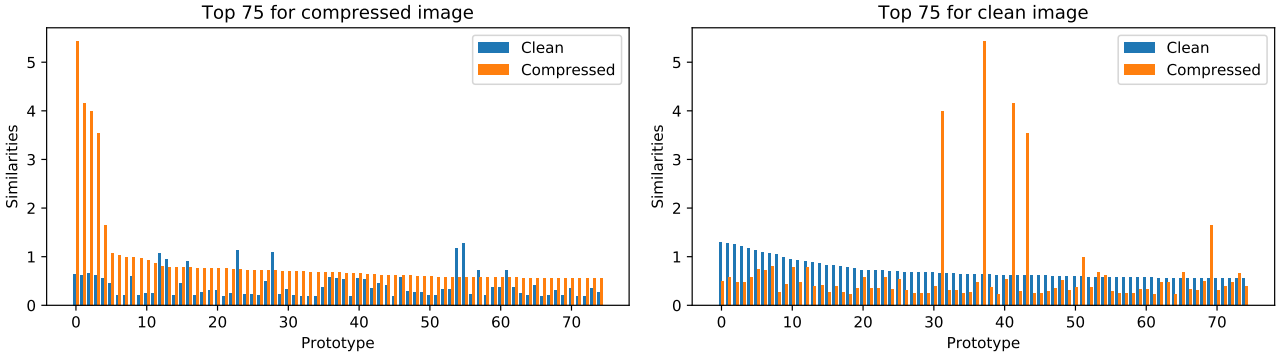



*Figure 12.* Highest similarities for the test image depicted in Figure 3 (b) on the ProtoPNet with the ResNet-34 backbone from Section 4.

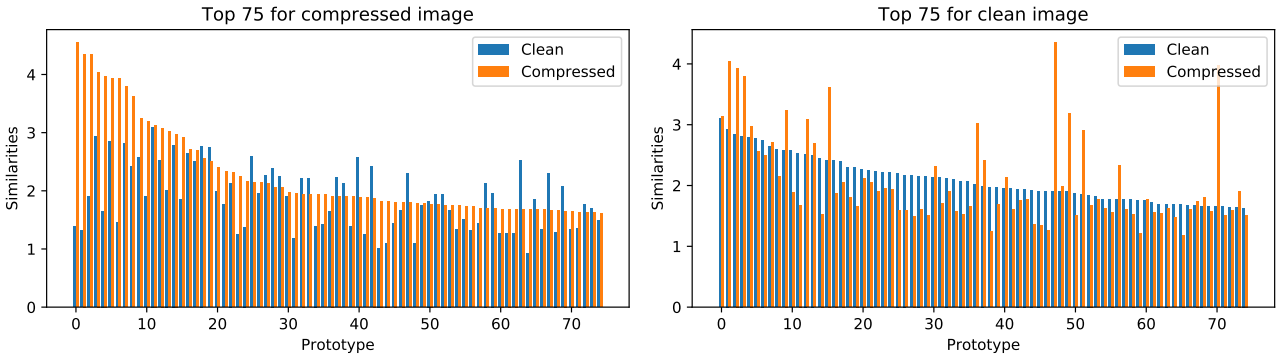



*Figure 13.* Highest similarities for the test image depicted in Figure 3 (c) on the ProtoPNet with the VGG-19 backbone from Section 4.

## D. Performance of ProtoPNets used in our Experiments

Table 1 lists the test accuracy of the trained ProtoPNets used in the Head on Stomach experiment. We obtain similar performance as reported in Chen et al. (2019) for all networks except VGG-19. We suspect that they use VGG-19 with batch normalization instead of vanilla VGG-19, while we use vanilla VGG-19 for our experiments. Notably, we achieve a test accuracy of 79.24% when using VGG-19 with batch normalization. Lastly, the test accuracies for the models used in the JPEG experiment are shown in Table 2.

| Backbone | Chen et al | Ours |
|---|---|---|
| ResNet-18 | - | 75.75 |
| ResNet-34 | $79.2 \pm 0.1$ | 79.58 |
| VGG-19 | $78.0 \pm 0.2$ | 76.39 |

*Table 1.* Test accuracies of the trained ProtoPNets used in the Head on Stomach experiment. We add the corresponding numbers from the *This Looks Like That* paper (Chen et al., 2019) for comparison.

| Backbone | Altered test set | Original test set |
|---|---|---|
| ResNet-18 | 80.19 | 44.75 |
| ResNet-34 | 82.14 | 46.12 |
| VGG-19 | 81.08 | 49.60 |

*Table 2.* Test accuracies of the ProtoPNets used in the JPEG experiment. Specifically, for the altered dataset we use the CUB-200-2011 test set and apply the same compression scheme as during training. For the original test set we simply take the CUB-200-2011 test set.

## E. Countermeasures to improve robustness

### E.1. Remedy Head on Stomach experiment

In Section 3, we showed that the interpretability mechanism of ProtoPNet can be abused to identify prototypes at arbitrary locations in the test image. Since the experiment utilises ideas from adversarial attacks it is natural to test if remedies from that domain work here as well. Specifically, we train the ProtoPNets with the ResNet-18 and ResNet-34 backbones via fast adversarial training (Wong et al., 2020). Note however that it is not clear a priori if this helps against the Head on Stomach problem. This since adversarial training was developed to help classify images correctly even if you add noise to an image. In our case, we don't have the problem of misclassification (see Section 3.1). Instead, we observe the issue that the latent representations admit a change in similarity scores with slight variations in the input space.

We perform adversarial training via fast gradient sign method (FGSM) with random initialization and set step size $\alpha = 10/255$, $\ell_\infty$ radius $\varepsilon = 8/255$. We train the models for 10 epochs and use the same training setup as mentioned in Appendix A. The performance of our trained models is listed in Table 3. The adversarial accuracy is evaluated using PGD with step size $2/255$, $\varepsilon = 8/255$ and 10 iterations.

Lastly, we devise a measure to quantify the susceptibility of ProtoPNet's interpretations to the Head on Stomach problem. For a ProtoPNet and test image $\mathbf{x}$, we consider the top $k$ ($= 5$) prototypes with the highest similarity (denoted $\{\mathbf{p}_{l_i}\}_{i=1}^k$) among all prototypes. Suppose the prototype $\mathbf{p}_{l_i}$ is recognized at the patch $S$ of the latent representation of image $\mathbf{x}$ and $A$ is the set of all patches in the latent representation of $\mathbf{x}$. We then try to perturb $\mathbf{x}$ with the goal that the ProtoPNet finds any arbitrary patch $S_{\text{noisy}} \in A \setminus S$ highly similar to $\mathbf{p}_{l_i}$ by maximizing the objective of Equation 2. Precisely, we solve the optimization problem stated in Equation 3 using PGD with step size $2/255$, $\varepsilon = 8/255$ and apply 40 iterations. Finally, if the similarity of $\mathbf{p}_{l_i}$ with $S_{\text{noisy}}$ is greater than that with $S$, we regard the experiment as being successful. We do

this for each of the top $k$ prototypes and consider the experiment as a success for image $\mathbf{x}$ if it succeeds for any of the $k$ prototypes. We repeat this procedure for a random subset of 200 test images that are correctly classified by the ProtoPNet under consideration and compute the success rate of our algorithm.

In Table 3, we also list the results of our evaluation. As we can see the adversarial training reduces the success rate of our algorithm by more than half but does not alleviate the problem entirely. Additionally, the robustness to our algorithm comes at a significant drop in accuracy on the test set (in line with adversarial training methods (Tsipras et al., 2019; Zhang et al., 2019)).

| Backbone | Standard Training | | | Adversarial Training | | |
|---|---|---|---|---|---|---|
| | Clean Acc. | Adverarial Acc. | Success Rate | Clean Acc. | Adverarial Acc. | Success Rate |
| ResNet-18 | 75.8 | 0.0 | 99.5 | 56.3 | 15.8 | 34.0 |
| ResNet-34 | 79.6 | 0.0 | 93.0 | 58.8 | 24.2 | 27.0 |

*Table 3.* Performance of ProtoPNets trained via standard and adversarial training respectively on the CUB-200-2011 test set. Clean and adversarial accuracy are evaluated on the entire test set while the success rate is computed on a random subset of 200 correctly classified test images.

### E.2. Remedy JPEG experiment

In Section 4 we showed that the interpretations provided by ProtoPNets can behave in a manner that we humans cannot comprehend if JPEG artefacts are introduced in the training data. In this section, we investigate if the problem can be alleviated by augmenting the training data with JPEG compressed images. Specifically, we train ProtoPNets for the same backbones as in Section 4 (i.e. ResNet-18, ResNet-34, VGG-19) but add JPEG compression to the augmentation pipeline. To be exact, every image in a mini-batch has a 50% chance of being JPEG compressed with 20% quality during training. All other augmentations and training parameters are as in Chen et al. (2019) for the CUB-200-2011 dataset. The new test accuracies for the models trained with JPEG random augmentation are shown in Table 4.
Figures 14, 15, and 16 depict the same visualisation of test images as shown in Figures 11, 12, and 13 but for the newly trained ProtoPNets. As can be seen, the similarities of the top prototypes behave much less erratic as in Appendix C.2. This suggests that the augmentation helps to alleviate the problem encountered in the JPEG experiment where the presence or absence of compression artefacts affected similarities in a way that humans find counter-intuitive. Yet, the problem is not completely resolved by the added augmentation step.

| Backbone | Altered test set | Original test set |
|---|---|---|
| ResNet-18 | 74.70 | 76.32 |
| ResNet-34 | 76.58 | 77.70 |
| VGG-19 | 74.59 | 76.30 |

*Table 4.* Test accuracies of the ProtoPNets used in the JPEG experiment with random JPEG augmentation during training. The altered dataset and the original dataset follow the same scheme as in Table 2.

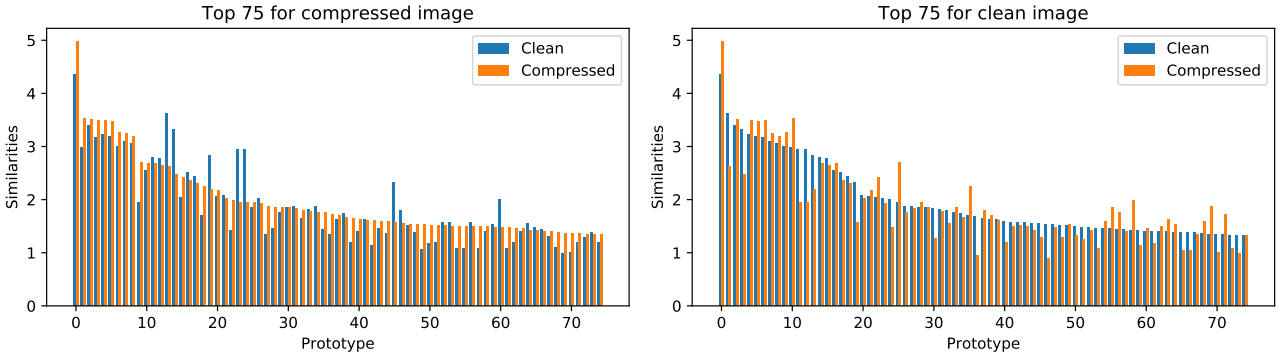



*Figure 14.* Highest similarities for the test image depicted in Figure 3 (a) on a ProtoPNet with the ResNet-18 backbone that was trained with JPEG compression as an augmentation step.

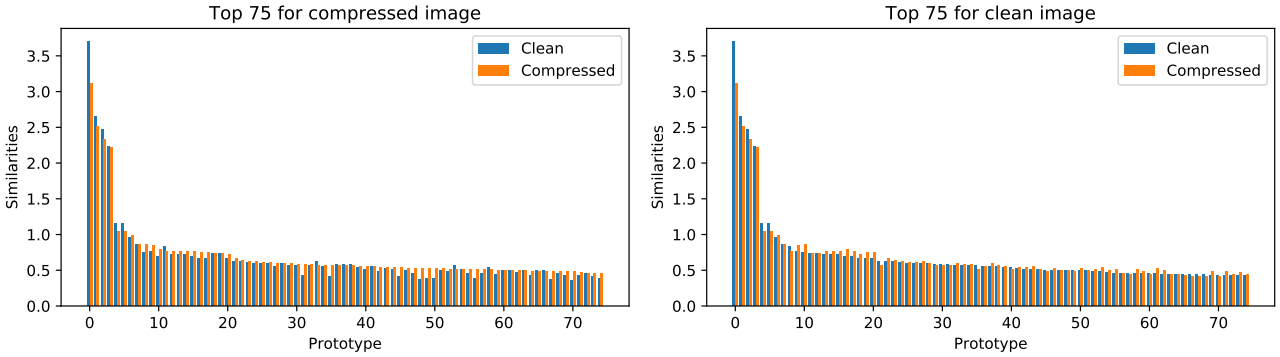



*Figure 15.* Highest similarities for the test image depicted in Figure 3 (b) on a ProtoPNet with the ResNet-34 backbone that was trained with JPEG compression as an augmentation step.

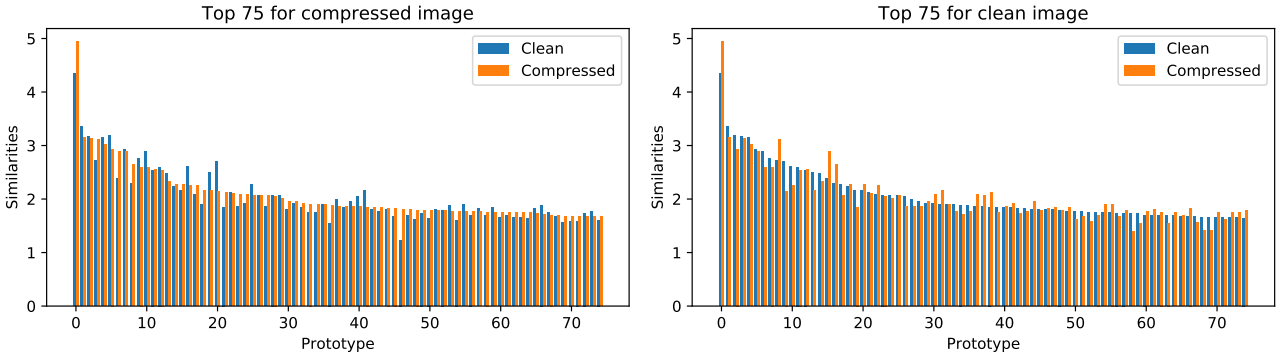



*Figure 16.* Highest similarities for the test image depicted in Figure 3 (c) on a ProtoPNet with the VGG-19 backbone that was trained with JPEG compression as an augmentation step.