Journal Pre-proof

This looks more like that: Enhancing Self-Explaining Models by Prototypical Relevance Propagation

Srishti Gautam, Marina M.-C. Höhne, Stine Hansen, Robert Jenssen, Michael Kampffmeyer

PII: S0031-3203(22)00651-3
DOI: https://doi.org/10.1016/j.patcog.2022.109172
Reference: PR 109172

To appear in: *Pattern Recognition*

Received date: 21 December 2021
Revised date: 19 June 2022
Accepted date: 9 November 2022

Please cite this article as: Srishti Gautam, Marina M.-C. Höhne, Stine Hansen, Robert Jenssen, Michael Kampffmeyer, This looks more like that: Enhancing Self-Explaining Models by Prototypical Relevance Propagation, *Pattern Recognition* (2022), doi: https://doi.org/10.1016/j.patcog.2022.109172

Highlights

- Detailed analysis of the shortcomings of the current state-of-the-art self-explaining model ProtoPNet.
- A novel method improving the precision of prototype explanations: Prototypical Relevance Propagation.
- Extensive qualitative and quantitative evaluation of the explanations regarding artifact detection.
- A multi view clustering approach to utilize PRP to detect and remove artifactual data.

This looks *more* like *that*: Enhancing Self-Explaining Models by Prototypical Relevance Propagation

Srishti Gautam[a,*], Marina M.-C. Hhne[b,a], Stine Hansen[a], Robert Jenssen[a], Michael Kampffmeyer[a]

[a] *UiT The Arctic University of Norway, Troms, Norway*
[b] *Technische Universitt Berlin, Berlin, Germany*

Abstract

Current machine learning models have shown high efficiency in solving a wide variety of real-world problems. However, their black box character poses a major challenge for the comprehensibility and traceability of the underlying decision-making strategies. As a remedy, numerous post-hoc and self-explanation methods have been developed to interpret the models' behavior. Those methods, in addition, enable the identification of artifacts that, inherent in the training data, can be erroneously learned by the model as class-relevant features. In this work, we provide a detailed case study of a representative for the state-of-the-art self-explaining network, ProtoPNet, in the presence of a spectrum of artifacts. Accordingly, we identify the main drawbacks of ProtoPNet, especially its coarse and spatially imprecise explanations. We address these limitations by introducing Prototypical Relevance Propagation (PRP), a novel method for generating more precise model-aware explanations. Furthermore, in order to obtain a clean, artifact-free dataset, we propose to use multi-view clustering strategies for segregating the artifact images using the PRP explanations, thereby suppressing the potential artifact learning in the models. The code will be made available on github upon acceptance.

Keywords: Self-Explaining Models, Explainable AI, Deep Learning, Artifact

*Corresponding author
Email addresses: srishti.gautam@uit.no (Srishti Gautam), marina.hoehne@tu-berlin.de (Marina M.-C. Hhne), s.hansen@uit.no (Stine Hansen), robert.jenssen@uit.no (Robert Jenssen), michael.c.kampffmeyer@uit.no (Michael Kampffmeyer)

detection.

2010 MSC: 00-01, 99-00

1. Introduction

When applying AI models, especially in safety-critical areas, such as medical applications, autonomous driving, or criminal justice, we need to understand their underlying behavior to decide the model's trustworthiness. Here, the field of explainable AI (XAI) has established itself, where methods are being developed to illuminate the so-called black box models [1, 2]. XAI serves as an essential support in ethical, legal, and social issues and ultimately also contributes to an increased acceptance by the end user [3] by revealing the input features that led to a certain model prediction.

Using those XAI methods, recent work has shown that models can learn artifacts that are present in the training data [4]. Such artifacts can be based on a so-called selection bias in the training data, where, for example, objects of a class have a certain background, and as a result the background is learned instead of the object. Furthermore, the training data can be manipulated by inserting a special trigger called "backdoor" which, if present in a sample, always leads to the prediction of a specific target class - i.e. a "backdoor" to this target class [5] In addition, a phenomenon called "Clever Hans", refers to an artifact that is correlated with a certain class in the training data and hence, used for classification such that the model could make a right prediction, but for the wrong - the artifact - reason [4]. In order to guarantee a faithful use of AI systems, it is important to find and suppress those artifacts either from the model, i.e., from the learnt representations or from the data itself, thereby enabling the retraining of the model with a clean dataset.

Recently, so-called post-hoc XAI methods, such as Layerwise Relevance Propagation (LRP) [6] were able to uncover this undesirable behavior of AI models [4]. Post-hoc refers to the fact that the XAI method explains the prediction of the model after (post) the prediction is made. However, [7] suggested to use an influential alternative to post-hoc explainability, called self-explaining neural networks, which can intrinsically explain their decision making process. Towards this goal, [8] recently proposed a

network (ProtoPNet) that provides a transparent prediction by introducing a prototype layer between the final convolution layer and the output layer. This prototype layer consists of a fixed number of prototypes for each class, which can be thought of as representative instances for each class of the training data. During the classification process, for each image that is passed through the network, prototype-specific activation maps are computed based on the similarity between the image and the prototypes. The visualization is performed by upsampling the activation maps to the input size, thus highlighting the most relevant pixels contributing to the classification. Doing this procedure for both, the prototype (training) images and the test image, the regions of interest can be visualized, serving as a direct comparison for the user to capture the relation between the test image and the prototype images from the training set. This accordingly helps in comprehending the decision of the network by "this relevant feature of the test image looks like that relevant feature from the class-specific prototype image" (*This* looks like *that*).

Recalling the artifacts issue, the solution now appears to be clear when using self-explaining neural networks, such as ProtoPNet: If the model learned a feature corresponding to the artifact, then it must be reflected by at least one of the prototypes of the class consisting of such artifacts. Consequently, once the artifact prototypes have been identified, their influence on the prediction can be stopped by pruning.

Interestingly, in this work we demonstrate that this idea of removing the artifact prototypes is not feasible owing to the coarse and spatially imprecise explanations provided by ProtoPNet, which is, due to its model-agnostic upsampling. Therefore, building on the principles of the post-hoc explanation method LRP, we propose a novel method referred to as Prototypical Relevance Propagation (PRP) to attain more accurate model-aware explanations (example shown in Figure 1). We demonstrate that PRP efficiently captures the learned artifact, which might go unnoticed otherwise. Additionally, in this work, we go one step further and suppress the potential artifact learned by the models: using PRP, we illustrate that artifact information is entangled within the ProtoPNet, such that most prototypes capture artifact related features, making the above-mentioned pruning procedure not applicable. Therefore, we propose to clean the data instead of pruning the network. Knowing the ability of PRP of generating

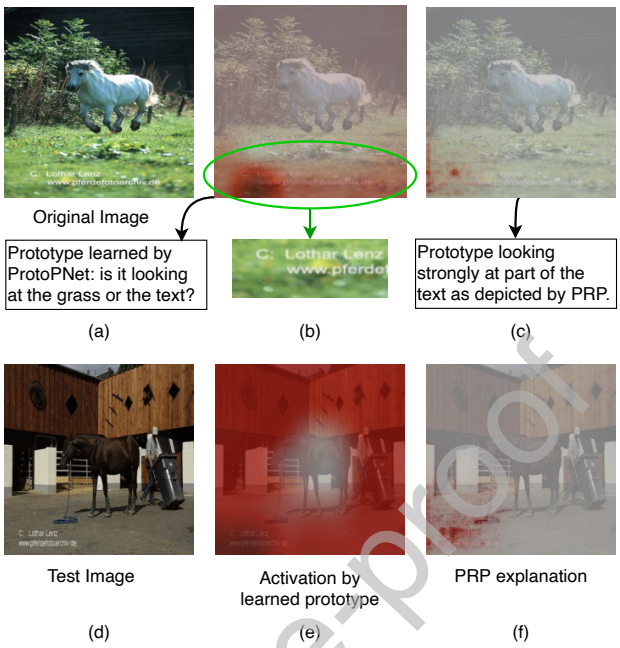


Figure 1: (a) Visualization of a horse image from the PASCAL VOC 2007 dataset [9] , (b) activation for a prototype of class *horse* learned by ProtoPNet, and (c) its PRP explanation. A Clever Hans artifact is present in the form of a watermark at the bottom of the image. Both, the ProtoPNet and the PRP explanation yield relevance to the bottom of the image, however, in the case of ProtoPNet, it remains unclear if the green grass, the text, or both together, were relevant for the prediction. Whereas the PRP explanation clearly shows that the text was used as relevant feature for the model's prediction. For a test image (d), the ProtoPNet's explanation and the PRP explanation for the learned prototype (b) are given in (e) and (f), respectively. The PRP explanation again corroborates the emphasis on the watermark text as opposed to ProtoPNet's explanation which is more widely spread across the image. The ProtoPNet explanation in (e) thus exhibits '*This* looks like *that*' behavior i.e explanation in (e) looks like prototype in (b). The PRP explanation in (f) exhibits '*This* looks *more* like *that*' behavior i.e, enhanced explanation in (f) looks *more* like that in (c).

multiple views of the input in terms of learned prototypical explanations, we filter out the data points containing the artifact using multi-view clustering approaches. Our presented approach preserves the strength yielded by ProtoPNet of obtaining "*This* looks like *that*" explanations, while at the same time suppressing potentially learned artifacts. Moreover, we show that utilising multiple views through multi-view clustering is more efficient than a single-view LRP-based clustering approach, SpRAy [4].

Our main contributions are as follows:

- We identify and address key issues with inaccurate explanations provided by the self-explaining model, ProtoPNet.
- We propose a novel PRP method for enhancing ProtoPNet's explanations by generating more precise model-aware explanations.
- We compare PRP with ProtoPNet's explanation heatmaps, both qualitatively and quantitatively and show that eradicating learned artifact features, such as the Clever Hans and Backdoor artifacts, from ProtoPNet is unfeasible.
- We show the ability of PRP in utilizing multiple explanations from different prototypes, which can be utilized to suppress artifacts from the data by using multi-view clustering.

2. Related work

2.1. Explainability methods

Recently, there has been increased interest in both post-hoc explanation methods and self-explaining neural networks. Post-hoc explainability methods can be separated into two overarching categories: model-agnostic and model-aware approaches. Model-agnostic approaches [10], such as LIME [11] and SHAP [12], consider the models as black-boxes and are thus applicable to arbitrary model architectures and can be used to compare models based on the explanations that they produce. In contrast, model-aware approaches [13] take the internal structure of the model into account, yielding more precise model based explanations. Here LRP [6] has been widely used to explain the decisions of various deep neural networks, such as convolutional neural networks, recurrent neural networks and graph neural networks [14]. LRP assigns relevances to the input features by backpropagating the prediction score, i.e., the output relevance, successively layer by layer until it is distributed over the input features. Hence, the distribution of relevance is based on how much a particular node contributed to the output.

Another new and promising category of explanation methods are self-explaining networks, which inherently explain the decisions they make, thereby making the models transparent by design. These include networks that align the latent space to known visual concepts in order to increase transparency in the decisions [8, 15]. These also include models that utilize attention mechanisms [16] and thus also provide some form of self-explainability. Other works consider self-explainability in terms of concept learning [17, 18]. Further, recently, some research has been originated to develop frameworks with a joint architecture consisting of an explainer and a classifier which learn in conjunction [19, 20]. ProtoPNet [8] proposes to learn a specific number of class based prototypes as a part of the architecture. These are then used for visualizing lower spatial dimensional concepts from the training images, thus providing explanations during the decision process itself. SENN [21] is a type of general self-explaining model that is fully transparent and designed by progressively generalizing linear classifiers to complex models. Although the self-explainable concepts in SENN are using prototypes similar to ProtoPNet, the former only shows which training images are important for a decision. ProtoPNet, on the other hand, additionally shows what part of the test image looks like which part of the training images, thus providing more comprehensible information. The Classification-By-Components (CBC) network [22] is designed based on Biederman's theory in psychology, which assigns positive, negative, and indefinite reasoning to different components used for classification. Unlike CBC, ProtoPNet is more flexible in terms of i) learning components (prototypes) of varying sizes in the input domain, and ii) having the capability of being incorporated into any network architecture.

Inspired by ProtoPNet, XProtoNet [23] was recently introduced for automated diagnosis in chest radiography. It addresses the issue that ProtoPNet looks at fixed patch sizes in the feature map while computing its similarity with the prototypes. As a remedy, [23] adds an occurrence module in the network for learning features of dynamic size for the prototypes. However, the issues that we address in this work do remain in XProtoNet, making it prone to misleading explanations due to the model-agnostic upsampling used for prototype visualizations.

2.2. Artifacts

Real-world data used for training deep neural networks are prone to containing spurious, incomplete, or wrongly labeled samples thus leading to unwanted artifactual data. In this work, we acknowledge this inherent problem and focus on two common artifacts, Clever Hans and Backdoor, whose suppression is the focus of this work. Clever Hans artifacts refer to the unintentional spurious correlations present in the training data, which a model might use to base or strengthen their decisions on and is thus likely to fail in a real-world scenario, where the artifact is absent. This undesirable setting has also been explored recently by [4], in which they propose a semi-automated method, SpRAy, based on spectral cluster analysis on LRP maps, to discover prediction strategies based on an artifact. In other scenarios, the network might be forced to learn undesirable features based on the malicious addition of hidden associations in the data with the goal to produce incorrect inference results, referred as backdoor attacks. These kinds of attacks — where, in contrast to the Clever Hans scenario, both the data and labels are intentionally modified — are addressed in detail in [5, 24].

3. An Evaluation of ProtoPNet

While the effectiveness of post-hoc explainability methods has been investigated extensively [25, 26] and their benefit has been questioned [7], there is a significant gap in the research for the analyses of the effectiveness of self-explainable approaches regarding quantitative analysis of the provided explanations [27]. Therefore, in this section, we provide a detailed analysis of ProtoPNet and its inherent explanations using a case study of Clever Hans artifact detection. As a representative for the self-explaining model, we focus on ProtoPNet as it claims to provide easily comprehensible case-based reasoning and is applicable to arbitrary CNN architectures by inserting a single prototype layer [8]. Additionally, it not only provides information about the features that the model's decision is based on, but also links this information to similar features in the training data, captured by the prototypes, thus imitating human decision making.

3.1. ProtoPNet

ProtoPNet introduces self-explanation in a deep learning network by incorporating a prototype layer between the last convolutional layer and the output layer. Thereby, each class is associated with a fixed number of prototypes. The output of the prototype layer is connected linearly to the output layer to generate class logits. The network is optimized by iterating the following three steps: 1) The whole network, except the last layer, is trained using stochastic gradient descent. For each prototype, the squared L_2 similarity between the patches of the convolutional output from the backbone and the prototype is calculated, thus generating an activation map. Global max pooling is applied to the activation map to generate a single similarity score corresponding to a single prototype. The loss function is a combination of the cross entropy loss, a cluster loss and a separation loss. The cluster loss encourages the training images to have a patch close to at least one of their own class prototypes. The separation loss, on the other hand, encourages the training image patches to be far from the prototypes of other classes [8]. For completeness, the losses are provided in the Appendix. 2) All prototypes are projected onto the patch of the training image from the same class as the prototype with the highest similarity score, thus maintaining inherent interpretability. These can be visualised in the input space by upsampling the activation map of the prototype image to the input size. 3) Finally, a convex optimization of the last layer is performed to further improve accuracy, while keeping the learned prototypes fixed. The prototype activations are visualized by upsampling the similarity between the prototypes and the embedded input image to the input image size. This highlights the parts of the image which strongly activate the respective prototype, thus creating a concept of "*this* looks like *that*" while making the decisions.

3.2. Evaluation of ProtoPNet's explanations

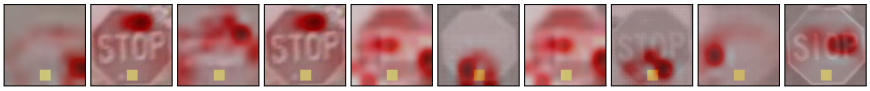
Although self-explaining models as ProtoPNet appear promising, as more transparent alternatives to the typical black-box neural networks, we demonstrate that, atleast for ProtoPNet, the explanation capability still lacks precision. In the case of ProtoPNet, the relevant areas on which the model decision is based on do not concisely depict the relevant features of a prototype as shown in Figure 1. The original image (a) in Figure

1 shows a horse image containing a watermark in the lower left corner. One of the 10 prototypes for class Horse was learned by ProtoPNet from image 1(a). The ProtoPNet's explanation for this prototype is shown in Figure 1(b). From 1(b), we can observe that the lower left corner was important for the model to predict the image as a horse. However, the exact pixels, that significantly contributed to the predictions remain unknown. Now, using the model-aware PRP method, we backpropagate the prototype information from the prototype layer through the network to the input image, which allows us to reveal and visualize the model-aware, faithfully distributed relevance scores on the input image as shown in Figure 1(c). From the PRP explanation, we observe that high relevance (dark red pixels) was allocated to parts of the text. Thus, the PRP explanation leads to an increased understanding of the underlying behavior of the model. For a randomly chosen test image, shown in Figure 1(d), the activation for the learned prototype 1(b) as visualized by ProtoPNet and PRP are given in Figure 1(e) and 1(f), respectively. The PRP explanation identifies the watermark (Clever Hans) as a relevant feature for predicting the horse class, in contrast to the ProtoPNet explanation, which is too crude to identify important features and is therefore widely spread across the entire image.

Accordingly, we detect and address the following drawbacks of ProtoPNet:

- The activation maps used for the prototype visualizations in ProtoPNet have a low resolution due to downsampling and feature aggregation functions in the network. From this significantly low resolution activation map, ProtoPNet performs model-agnostic upsampling using bilinear interpolation to the size of the input image, thus leading to very **coarse explanations**.
- The effective receptive field of a position in the activation map tends to cover large parts of the image, which is not captured by the naive upsampling. Consequently, there is no truthful spatial localization of the relevance to the correct input area, leading to **spatially imprecise explanations**.

In the next subsection, we discuss in detail these drawbacks of ProtoPNet's explanations using the Clever Hans artifact as an example.



1 2 3 4 5 6 7 8 9 10

Figure 2: CH-100: Visualization of the prototypes learned for the stop sign class for the scenario where Clever Hans artifacts were inserted in 100% of the stop sign class images for the modified LISA dataset. As observed, while prototype 6 and 8 can be considered as artifact prototypes, none of the prototypes clearly highlight the artifact.

3.3. Case Study: Clever Hans artifact detection with ProtoPNet

Ideally, ProtoPNet should capture any artifact in the data as an "artifact prototype" if it is using the artifact for prediction. However, due to its coarse and spatially imprecise explanations, the heatmaps of ProtoPNet hinder the detection of artifact prototypes. In the following, we investigate the behavior of ProtoPNet in the presence of Clever Hans artifacts in the data.

We aim to detect the aforementioned artifact prototypes using ProtoPNet's explanations combined with the difference in classification results in the presence and absence of artifacts in the test data. Following this, we prune the detected artifact prototypes, thus hypothetically suppressing the artifacts learnt by the model. However, due to its misleading explanations, we demonstrate experimentally that ProtoPNet's heatmaps are deficient in capturing and identifying the learned artifact by the model, thus proving the task of pruning artifact prototypes futile for making the model artifact-free.

For considering a controlled environment, we use the 5-class version of the LISA traffic sign dataset [28] and place a Clever Hans artifact, a yellow square (see Figure 5 (Input)), in 100% of the training data of the stop sign class (dataset details are provided in Section 5.1), which we refer to as CH-100. We train the ProtoPNet (for implementation details see 5.2), with 10 prototypes per class as in [8] for ease of comparison.

To evaluate the impact of an artifact on the model, we evaluate the performance on two test data sets: an **Artifact Test** data set, where the Clever Hans, i.e., the yellow square, is inserted into 100% of the images of the stop sign class ; and a **Clean Test** data set, which contains no yellow square. The accuracy results for both test data sets are shown in Table 1. We can observe that the model, trained on the CH-100 dataset,

Table 1: Comparison of the model accuracies for the stop sign class between the artifact test (artifacts in 100% test images) and clean test (artifacts in 0% test images) dataset for : 1) CH-100, 2) CH-50 datasets, along with the accuracies for pruning artifact prototypes as well as retraining the last layer after pruning.

	CH-100	CH-100 Remove prototype 6 & 8	CH-100 Retraining last layer	CH-50	CH-50 Remove prototype 4 & 9	CH-50 Retraining last layer
Artifact Test	100%	21.6%	88.8%	100%	100%	100%
Clean Test	6.5%	38.2%	38.2%	94.6%	93.0%	94.5%

has 100% classification accuracy on the artifact test data and only 6.5% on the clean test data. This large drop in the accuracy indicates that the model has learned the inserted artifact. In order to detect the prototypes that are responsible for this behavior, we visualize the 10 prototypes learned by the network for the stop sign class in Figure 2, where the upsampled activation heatmap is overlayed, such that the relevant areas of each prototype can be identified visually. Although no prototype is clearly focusing on the artifact, it appears that prototypes 6 and 8 might be learning a part of the artifact. By removing individual prototypes as well as combinations of prototypes for the stop sign class, we can confirm that prototypes 6 and 8 are the most responsible ones for detecting the artifact (Figure 3) — the accuracy for artifact test data only drops when prototypes 6 or 8 are removed, with the biggest drop of 78.39% when both of these are removed together. Also note that no retraining is done yet after pruning the prototypes.

Now, trusting the explanations provided, we remove the artifact prototypes 6 and 8 and assume that this leads to the elimination of the artifact effect. As can be seen in Table 1, the accuracy for the artifact stop sign class drops considerably after removing prototypes 6 and 8. However, this is not the case as seen after retraining the last layer i.e, reweighing the connection of the prototypes to the final classification layer. The accuracy for the artifact stop sign class increases again to 88.8% once the last layer weights are retrained. Moreover, for clean test data, the accuracy remains the same, i.e, 38.2% before and after retraining the last layer, thus refuting the potential learning of meaningful features for the stop sign class by the model after retraining. Hence, the results indicate that the remaining prototypes include artifact information as well, highlighting the lack of accurate explanations by ProtoPNet.

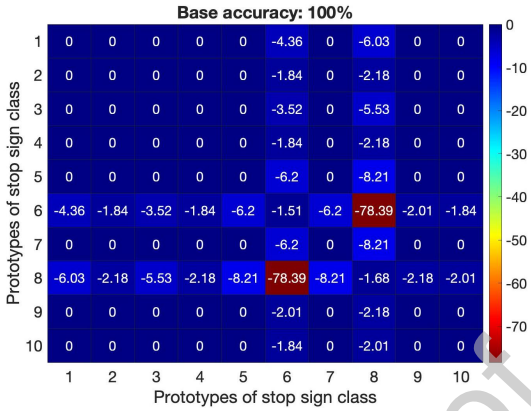


Figure 3: CH-100: Detection of artifact prototypes by removing individual stop-sign class prototypes (1 to 10) (diagonal) and their combinations (non-diagonal) for artifact test data. The accuracies are represented as a drop from the base accuracy of 100% when no prototypes are removed. The highest drop of 78.39% is observed when prototypes 6 and 8 are removed together thus highlighting them as artifact prototypes.

Thus, as shown in the above experiment, the explanations provided by the upsampling strategy of ProtoPNet are insufficient in order to reveal the model's behavior and detect the artifacts faithfully.

4. Prototypical Relevance Propagation and Enhanced Suppression of Artifacts

In the following we will address the two main drawbacks of ProtoPNet's visualizations, i.e., low resolution activation maps and spatially imprecise prototype explanations (as investigated in the section above), by our proposed method called Prototypical Relevance Propagation (PRP). Our aim is to maintain the advantage of self-explanatory architecture through prototypes and simultaneously improve the quality of prototypical explanations by adding, inspired by LRP, a model-aware explanation strategy.

4.1. Prototypical Relevance Propagation (PRP)

The original prototype visualization step in ProtoPNet is achieved through upsampling and is therefore decoupled from the other steps in its end-to-end training. Instead of upsampling, inspired by LRP, we suggest as a novel solution to use the knowledge of the inner workings of the network when backpropagating the similarity values of a

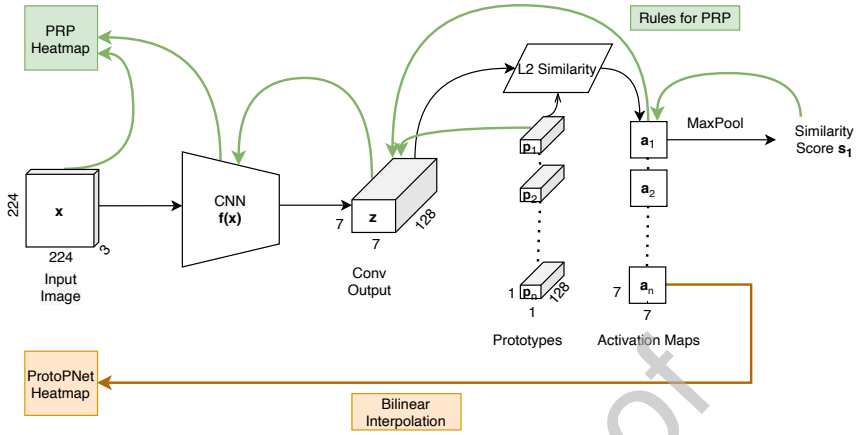


Figure 4: ProtoPNet: Forward propagation and backward propagation for PRP maps (green) and ProtoPNet Heatmaps (orange). The input image $\mathbf{x}$ is first passed through a CNN f, which computes $f(\mathbf{x})$ to give output $\mathbf{z}$. The squared L_2 similarity is then computed between $\mathbf{z}$ and individual prototypes $\mathbf{p}_m$ to get activation maps $\mathbf{a}_m$. These are then upsampled to get ProtoPNet heatmaps. On the other hand, similarity scores $\mathbf{s}_m$ are used to compute model-aware PRP heatmaps. All the parameters in the figure are depicted according to the experiment settings used in this work.

prototype to the input, such that we obtain model-aware prototypical explanations. We refer to our method as PRP and the generated explanation maps as PRP maps.

For the following considerations, let the input images be represented as $\mathbf{x}$ and convolutional output from the backbone CNN as $\mathbf{z} \in \mathcal{R}^{H \times W \times D}$. Let $\mathbf{P} = \{\mathbf{p}_m\}_{m=1}^{n}$ be the n prototypes learned by the network, each with a shape of $H_1 \times W_1 \times D$. Following [8], we set $H_1 = W_1 = 1$ and $D = 128$. Moreover, let $\mathbf{S} = \{\mathbf{s}_m\}_{m=1}^{n}$ be the similarity scores and $\mathbf{A} = \{\mathbf{a}_m\}_{m=1}^{n}$ the activation maps for each prototype. The forward computations in ProtoPNet, illustrated in Figure 4, are defined as follows:

1. The computation from the input to the convolutional output is given by $\mathbf{z} = f(\mathbf{x})$, where the function f represents the trained backbone CNN.
2. The activation maps are computed as squared L_2 similarities between the last convolutional output layer and the prototypes in the prototype layer:

$$\mathbf{a}_m = \log\left((||\widetilde{\mathbf{z}} - \mathbf{p}_m||_2^2 + 1)/(||\widetilde{\mathbf{z}} - \mathbf{p}_m||_2^2 + \epsilon)\right) \quad (1)$$

where $\widetilde{\mathbf{z}}$ are patches of $\mathbf{z}$ of the same size as the prototypes $\mathbf{p}_m$ and $\epsilon = 10^{-4}$ is

a small constant introduced for numerical stability.

3. The similarity score based on the activation maps is calculated as $\mathbf{s}_m = \max(\mathbf{a}_m)$

The similarity scores of the test image with prototypes are the inputs to the final fully connected layer, which produces the logits for all output classes. Hence, the final classification is based on a linear combination of the similarity scores of different prototypes.

Now, to improve the precision of the prototype visualizations, we calculate a certain prototype m by propagating the relevance of this prototype back to the input features. Note that the relevance of a specific prototype is exactly its similarity score. Therefore, the first backpropagation step considers the redistribution of the similarity scores towards the activation map with respect to the max pooling layer:

1. An activation map is computed by backpropagating the respective similarity score with the LRP rule in the Max pooling layer:

$$\mathbf{R}_{mij}^{(AM,S)} = \begin{cases} \mathbf{R}_m^{(S)} & \text{if argmax}_{ij}(\mathbf{a}_m), \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

where S refers to the similarity score layer, AM to the activation map layer and i, j specify the spatial location in the respective layers. We define the relevance at layer S as $\mathbf{R}_m^{(S)} = \mathrm{s}_m$.

2. To distribute the relevance from the activation map back to the convolutional output, we need to incorporate the information from the forward pass. The forward computation as given in Eq. 1 computes the similarity between each prototype and each output patch of the convolutional layer ($CONV$), with both having D channels, thus compressing the channel dimension to 1 in the activation map. In this step, we redistribute the relevance from the one channel activation map back to the D channels of the convolutional output, weighted by the corresponding channel-wise L_2 similarities computed during the forward pass. We define the channel-wise similarities between each CNN patch $\widetilde{\mathbf{z}}$ and the prototype $\mathbf{p}_m$ as:

$$\gamma_{mc} = \frac{1}{d_{mijc} + \epsilon} \tag{3}$$

where, with $d_{mijc} = ||\widetilde{\mathbf{z}}_\mathbf{c} - \mathbf{p}_{mc}||_2^2$ for each channel c. Afterwards, we use the LRP$_\epsilon$ [6] rule to distribute relevances to convolutional output according to γ_{mc}:

$$\mathbf{R}_{mijc}^{(CONV,AM)} = \frac{\gamma_{mc}}{\sum_{k=1}^{D} \gamma_{mk} + \epsilon} \mathbf{R}_{mij}^{(AM)} \tag{4}$$

3. Finally, the PRP maps are computed by distributing the relevance from the convolutional output to the input features with the LRP CoMPosite (**LRP**$_{CMP}$) rule [29]: First, the LRP$_{\alpha\beta}$ rule is applied to the convolutional layers

$$\mathbf{R}_{i\leftarrow j}^{(l,l+1)} = \left(\alpha \frac{z_{ij}^+}{z_j^+} + \beta \frac{z_{ij}^-}{z_j^-}\right) \mathbf{R}_j^{(l+1)}, \tag{5}$$

where $z_{ij} = x_i w_{ij}$ is the mapping of the input x from neuron $i \rightarrow j$ with weight w_{ij}, $z_j = \sum_i z_{ij}$, $\alpha + \beta = 1$ and $\alpha \geq 1$. Note that positive and negative activations are treated separately and we use $\alpha = 1$ and $\beta = 0$. [1]

Second, the Deep Taylor Decomposition based rule DTD$_{z^B}$ [30] is applied to the input features

$$\mathbf{R}_{i\leftarrow j}^{(l,l+1)} = \left(\frac{z_{ij} - l_i w_{ij}^+ - h_i w_{ij}^-}{\sum_i z_{ij} - l_i w_{ij}^+ - h_i w_{ij}^-}\right) \mathbf{R}_j^{(l+1)}, \tag{6}$$

where l_i and h_i are the smallest and largest pixel values.

The algorithm for generating PRP maps is summarized in Algorithm 1.

4.2. Multi-view Clustering

In order to analyse the class-wise prediction strategies and reveal potential strategies that are based on artifacts, [4] introduced SpRAy, a method that utilizes spectral cluster analysis to cluster LRP explanations into their key prediction strategies. Similar to SpRAy, we want to make use of the PRP maps to identify class specific global discriminative features. However, we do have multiple explanations for each image, i.e., the prototype explanations, which can be thought of as multiple views of an image

[1] Note, for notation simplicity, we follow previous works [6, 29] and consider the convolutional layers as fully-connected layers with shared weights.

Algorithm 1: PRP

Input: Model f, image $\mathbf{x}$, prototype number m

$\mathbf{z} = f(\mathbf{x})$ /* Forward computation */

Compute $\mathbf{a}_m$ // Eq.1

$\mathbf{R}_m^{(S)} = \mathbf{s}_m = \max(\mathbf{a}_m)$

Compute $\mathbf{R}_{mij}^{(AM,S)}$ // Eq.2 /* Backward computation */

Compute $\mathbf{R}_{mijc}^{(CONV,AM)}$ // Eq.4

for $l \in CONV - 1, ..., 1$ **do**

  $\mathbf{R}_{i \leftarrow j}^{(l,l+1)}$ using $\mathbf{LRP}_{CMP}$ rules

end

Output: $\mathbf{R}^{(1)}$

explanation. Thus, unlike SpRAy, which uses one LRP explanation for one image, our proposed method exploits multiple views of an image explanation.

In ProtoPNet, each class is associated with a fixed number of class prototypes. These can be regarded as capturing, and thus searching for, different features in each input image. Consequently, if there are artifacts present in a class during training, the PRP explanation maps for this class prototypes will be able to reflect the contrast between artifact and non-artifact features learnt by the model. Therefore, interpreting the different prototype activations as various views of the same image, allows us to compare/cluster the prototype activations with multi-view clustering algorithms in order to detect global class-discriminative features in the data. Traditional multi-view clustering methods include learning a common representation from multiple views of data followed by clustering [31] or learning adaptive representations based on clustering [32]. Further, several multi-view clustering algorithms have been proposed that build on spectral clustering and consider a consensus Laplacian matrix among all the views [33, 34]. In contrast, deep-learning based multi-view clustering methodologies learn a common encoding with the help of deep neural networks, which then can be leveraged by the clustering module [35]. Since a variation in clustering results can be observed using different multi-view clustering methodologies, in this work, we demon-

strate the performance with a recent deep learning based clustering method [35] and a representative spectral multi-view clustering algorithm [33].

The deep multi-view clustering in [35] first transforms each input into its representation using view-specific encoders . The fused representation for all views is then computed using the fusion weights, which are also learned during the end-to-end training. This representation is then passed through a fully connected network to obtain the final cluster assignments. Deep divergence based clustering (DDC) [36] losses are incorporated to optimize the model. This approach is termed as Simple Multi-View Clustering (SiMVC). [35] then introduces an auxiliary method which incorporates selective contrastive alignment of representations called Contrastive Multi-View Clustering (CoMVC) by adding a contrastive loss to the SiMVC framework. We provide the results with CoMVC in this work considering its additional advantage of aligning the representations at the sample level.

The spectral multi-view clustering methods work on the general principle of computing a consensus Laplacian matrix among all views. Co-regularized Multi-view Spectral Clustering (Co-Reg [33]) works by co-regularizing the clustering hypotheses. They obtain the combined Laplacian matrix by regularizing eigenvectors of the Laplacians through two schemes: 1) pairwise co-regularization, where they encourage the pairwise similarities across all views to be high and 2) centroid-based co-regularization, where they encourage each view to be closer to a common centroid.

5. Experiments & Results

In this section, we first discuss the dataset and implementation details followed by detailed analysis of ProtoPNet and PRP heatmaps. Finally, we discuss in detail artifact suppression using multi-view clustering.

5.1. Dataset

In this work, we conduct experiments for both the Clever Hans and the Backdoor artifact using the LISA traffic sign dataset [28]. This dataset consists of video frames captured from a driving car. We follow the strategy of [5], where we extract the frames

and resize them to 224x224 to be compatible with the original ProtoPNet architecture. The 47 classes in the dataset are partitioned into 5 high-level classes, as proposed by [5], consisting of restriction, speed limits, stop, warning, and yield signs (details provided in the appendix). In addition, we use the PASCAL VOC 2007 dataset [9] for evaluation as it naturally contains a Clever Hans artifact. [2]

5.1.1. Clever Hans

As artifact, we place a yellow post-it note, as shown in the input image in Figure 5, in 100%, 50% and 20% of the stop sign images in the training data of the LISA traffic sign dataset to create the CH-100, CH-50 and CH-20 Clever Hans training datasets, respectively. We do not add Clever Hans artifacts to the PASCAL VOC 2007 dataset since it inherently includes a watermark tag of the photographer in about 15-20% of the images in the horse class [4].

5.1.2. Backdoor

According to the data manipulation scheme for backdoor attacks from [5] we insert the artifact, i.e., the yellow post-it, as shown in Figure 5 (Input), in 15% of the stop sign images and assign them to the speed limit class. We refer to this corrupted training dataset as BD-15.

In order to create both, an artifact and a non-artifact i.e., a clean test dataset of the LISA traffic sign dataset, we insert the artifact in either 100% or 0% of the stop sign images, referred as Artifact Test and Clean Test data, respectively. Those test datasets are used for evaluating our experiments on the Clever Hans (CH-100, CH-50 and CH-20) as well as the Backdoor (BD-15) scenarios.

5.2. Implementation

We train ProtoPNet with ResNet34 as backbone architecture, fixing the number of prototypes to 10 for each class. Note that all training parameters have been set

[2] Since in PASCAL VOC 2007, one image can belong to several classes, we deliberately remove the person class from this dataset to decrease ambiguity. The person images overlap to a large extent with the images of the other classes, leading to a lot of duplicate images in multiple classes.

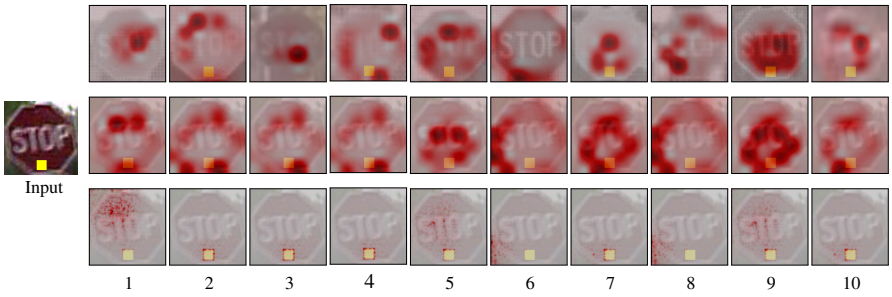


Figure 5: CH-50: Top row depicts the learned prototypes 1 to 10 for the stop sign class with Clever Hans in 50% of the training data, the middle row depicts the ProtoPNet's heatmaps corresponding to the respective prototypes for the test image shown on the left, while the bottom row shows the corresponding PRP maps, which, we can observe, capture more precise information.

according to [8]. The network is trained for 1000 epochs, where a projection (push) of the prototypes is done every 10 epochs. After each push, the last layer is trained for 20 epochs. The learning rate is reduced by a factor of 0.1 every 5 epochs and the training is stopped when the training accuracy converges and the cluster loss becomes smaller than the separation loss on the training set [8]. While ProtoPNet uses bilinear interpolation for visualization, which takes 0.001 seconds on average, computed for 1000 images, PRP has an additional overhead of 0.71 seconds for one backward pass to generate the heatmaps. Note, given that heatmaps are produced only after training the model, this overhead can be considered negligible. The code is implemented using PyTorch and the experiments were run on 2 GeForce RTX 2080 Ti GPUs [3].

5.3. PRP maps vs ProtoPNet heatmaps

In the following, we conduct an experiment, where we add a Clever Hans feature to the training dataset to investigate the difference between the heatmaps of ProtoPNet and the ones that PRP generates. Therefore, we add the Clever Hans artifact to 50% of the stop sign images in the training data (CH-50). The 10 prototypes for the stop sign class, learned by the ProtoPNet trained on the manipulated dataset, are shown in the first row of Figure 5. Given a test image, shown at the very left of Figure 5, the heatmaps of

[3] Code will be made open source available at github upon acceptance.

ProtoPNet and the PRP heatmaps for the image are shown in the middle and bottom row of Figure 5. Corroborating our earlier observations, we again note here that the ProtoPNet heatmaps are coarse, highlighting wider areas in the test image, and that neighboring regions of the artifact are focused upon, rather than the precise location of the artifact. In contrast, from the PRP maps, we can clearly observe that all prototypes are focusing precisely on the Clever Hans feature, some more (prototypes 2, 3, 4, 5, 7, 9, 10) and some less (prototypes 1, 6, 8). It is shown later that prototypes 6 and 8 are in fact not learning any significant features and even react strongly to random noise. With the new insight into the model behavior gained through the PRP maps, we can shed new light on the hypothesis from Section 3.3. The idea was to remove the prototypes that had learned the Clever Hans, retrain the last layer and thus eliminate the Clever Hans effect. Given the original prototype explanation, this made sense, as only 2 of the 10 prototypes had learned the Clever Hans feature. With the PRP maps, however, we gain new knowledge and can see that all prototypes (some more, some less) take into account the Clever Hans feature.

We also note here that ProtoPNet heatmaps are highlighting all pixels in the image activated by different prototypes (before Max Pooling). If they were highlighting only the maximally activated region (after Max Pooling), they would only be able to depict connected regions in the image space, considering the naive upsampling heavily based on spatial location correspondence between the activation map and the input image. On the other hand, PRP maps represent the maximally activated pixels and are still able to highlight disjointed areas in the image, as can be seen in the PRP map for Prototype 5 in Figure 5, where both the artifact and "ST" in the stop sign are indicated as relevant. Figure 6 illustrates the difference between PRP maps and ProtoPNet heatmaps for a stop sign image with no artifact. PRP maps, as shown in the bottom row, are of higher resolution and, as noticed in this case, tend to show more accurate information than the normal upsampled heatmaps from ProtoPNet. PRP maps also contain higher variability, as shown by explanations for Prototype 2 and 4 in Figure 6, which therefore yields more information from the original prototypes to explain the test pattern.

In the following, we quantitatively evaluate the faithfulness of the PRP maps and ProtoPNet heatmaps regarding their ability to capture the most discriminative class-

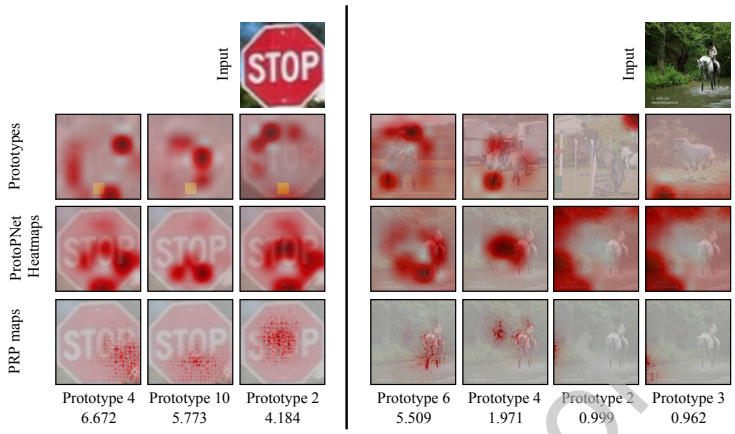
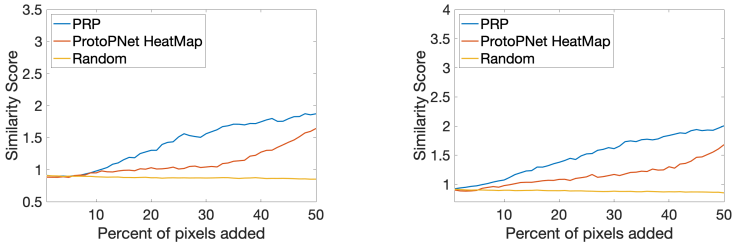


Figure 6: PRP Maps vs Activation Map Upsampling for CH-50 (left) and PASCAL VOC 2007 (right). The top 3 activated prototypes for the stop sign class and the top 4 activated prototypes for the horse class for the respective input images are shown in the second row in descending order of similarity scores (last row). The third row shows the heatmaps generated by ProtoPNet and the last row shows the corresponding PRP maps.

wise information. For this, we follow the strategy presented in [37], referred to as the Relevance ordering test, where we start from a random image and monitor both the similarity scores as we gradually add the most relevant pixels to the image.

Primarily, we are interested in the trustworthiness of the ProtoPNet heatmaps and PRP maps with regard to their calculated pixel relevance for activating the prototypes. Therefore, first, for an input image, the PRP maps and the ProtoPNet heatmaps are computed, followed by sorting the pixels in descending order of their assigned relevance by PRP and ProtoPNet explanations, respectively. We then compute the similarity scores for different prototypes of the stop sign images while gradually adding the pixel with the next highest relevance to a random image. We compute this for 50 randomly chosen clean images from the stop sign class and compute the average across all images followed by an average over all prototypes. The same experiment is repeated with the same images, this time adding the Clever Hans artifact. The average results for all prototypes of the stop sign class are shown in Figure 7. The x-axis represents the percentage of pixels that are replaced by the relevant pixels of the test image and the y-axis represents the corresponding similarity scores. As a baseline, we start from



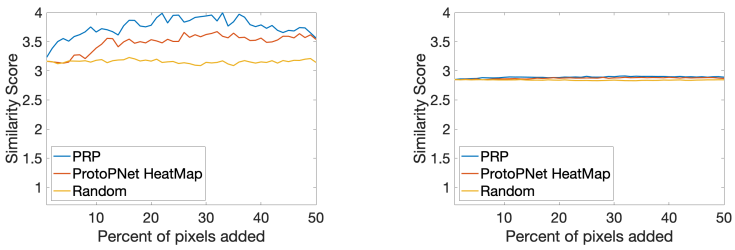
Images without artifact Images with artifact

Figure 7: CH-50: Quantitative evaluation of PRP Maps vs ProtoPNet Heatmaps via relevance ordering test. The results are shown as an average over all the prototypes and averaged over the same images without (left) and with artifact (right).

a random image and gradually replace a percentage of randomly chosen pixels by their test image pixel values and refer to this as the Random approach. From Figure 7 we can observe that for both test case scenarios, i.e. the stop sign images with and without the artifact, adding the most relevant pixels, based on the PRP explanations, results in a significantly steeper slope (blue) than using the ProtoPNet heatmaps (orange). Therefore, conclusively, we can state that the relevance of the important discriminate features distributed by PRP is more accurate than by ProtoPNet explanations. These quantitative results also uncover ineffective prototypes which are not learning anything specific from the training images and are reacting very highly even to random noise, as shown in Figure 8. This behavior is observed in both test scenarios of clean and artifact data, with the results depicted for artifact test images in Figure 8 for prototypes 6 and 8.

5.4. Assessing the network behavior with PRP maps

So far, we have established the drawbacks of ProtoPNet, which are the lack of higher resolution and spatially precise explanations, which hinder the user in identifying the most relevant discriminative features. Accordingly, we proposed a method — PRP — to overcome this lack of precise explanations. Our proposed PRP maps provide a higher level of fine grained explanations while keeping the benefit of "this-looks-like-that" behavior of the ProtoPNet, as shown in Figure 9 for both LISA (CH-50) and PASCAL VOC 2007 datasets. Therefore, we still have inherent interpretability, where each class is being represented by a fixed number of prototypes. This exponentially



Prototype 6 Prototype 8

Figure 8: CH-50: Relevance ordering test results shown for prototypes 6 and 8 of the stop sign class for the artifact test images. Both of these are not learning anything specific, therefore having high similarity with even random data.

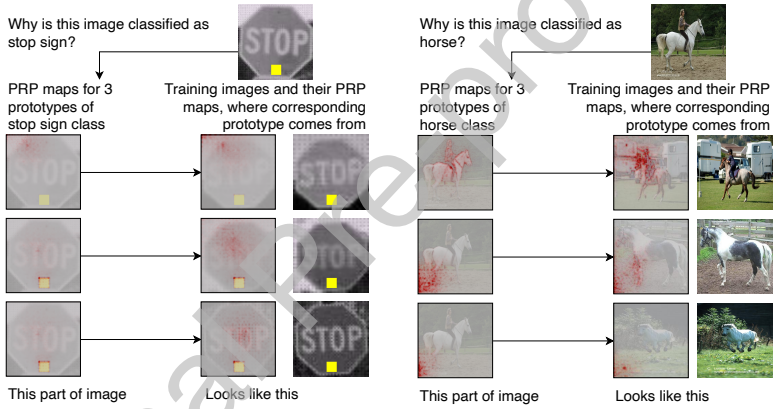


Figure 9: *This* looks *more* like *that*: Enhanced ProtoPNet self-explainability with PRP for a LISA stop sign image from the CH-50 dataset (left) and a PASCAL VOC horse image (right).

reduces the need for the manual laborious task of analysing individual ad-hoc explainability heatmaps for assessing deep neural networks. Additionally, this also reduces the need to use semi-automated methodologies like SpRAy [4] to find patterns in a model's explanations with a huge number of explanation maps.

We can now directly visually identify the strategies learned by the network by only looking at a few representative prototypes for each class. For instance, we manually cluster the PRP maps of the stop sign class for the LISA dataset, as shown in Figure 10. We can observe, that aside from learning the artifact, the network is also relying on the textual part of the stop signs as well as on the corner features. Note, that we have

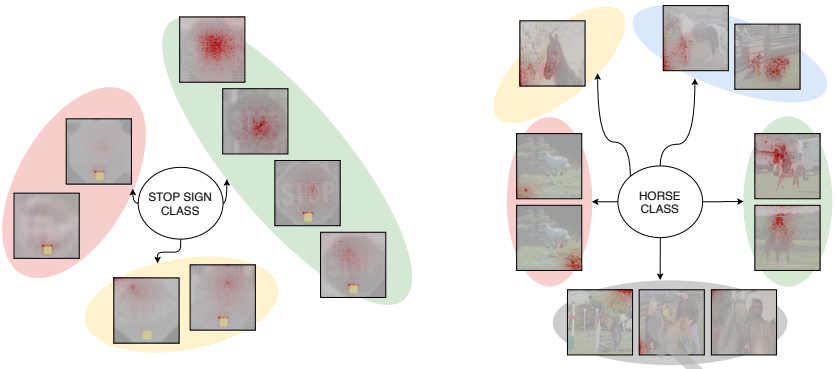


Figure 10: Representing cluster of prototypes for the stop sign class (left) and Horse class (right). For the stop sign class, the red cluster predominantly highlights the artifact, the green cluster indicates the text, while the yellow cluster captures the corner features. For the Horse class, red cluster looks at the "Clever-Hans" i.e, the watermark in the images, the yellow cluster highlights the features of the horse's mouth, the blue cluster indicates the presence of horse-type legs, the green cluster looks if there is a rider present, and the gray cluster captures the background features and is thus insignificant.

excluded prototypes 6 and 8 from the assessment since they did not capture any useful information (see Figure 8).

Following this, we investigate the performance of PRP and ProtoPNet explanations on the PASCAL VOC 2007 dataset in order to uncover relevant features learned by the networks for predicting the class horse. First, we show a few prototypes (top 4 activated) that were learned by the model for the horse class along with their ProtoPNet heatmaps and PRP Maps, shown in Figure 6 (right). Here, we can observe that PRP explanations capture the relevant features in a more fine grained manner and are able to identify a Clever Hans strategy used by the model where it tends to focus on the text in the watermark in prototype 3, rather than on the horse. In contrast, the information in ProtoPNet's heatmaps in the second row of Figure 6 is ambiguous since prototype 3 is allocating relevance to a broader background area. The strategies learnt by the network for recognizing a horse are grouped manually and visualized in Figure 10. The four effective groups, disregarding the insignificant gray cluster, which focuses on the background features, represent the horse class in terms of a horse's face, legs, presence of a rider, and finally the Clever Hans watermark.

Table 2: Accuracy (ACC) and F1-scores (F1) for different data scenarios with several multi-view clustering methodologies on PRP maps along with comparison with SpRAy on both PRP and LRP maps. Best and insignificantly different results, computed using t-test, are marked in bold.

	SpRAy-LRP[4]		**SpRAy-PRP**[4]		**CoMVC**[35]		**Co-Reg**[33]	
	ACC (%)	F1	ACC (%)	F1	ACC (%)	F1	ACC (%)	F1
CH-50	54.06±1.62	0.68±0.01	53.52±0.75	0.68±0.04	**99.99±0.00**	**0.99±0.00**	99.57 ± 0.00	0.99±0.00
CH-20	75.92±1.11	0.08±0.03	81.98±1.55	0.28±0.03	82.27±20.52	0.75±0.24	**94.54±0.00**	**0.86±0.00**
BD-15	83.18±5.76	0.21±0.24	85.72±3.87	0.30±0.15	66.85±6.91	0.76±0.06	**99.42±0.00**	**0.98±0.00**

5.5. Multi-View Clustering for suppressing artifacts

Artifacts in the data can be learned by the model, which subsequently might lead to the model exhibiting undesirable behavior, as shown in [4] and demonstrated above in case of the self-explaining network ProtoPNet. Consequently, it is essential to either remove the artifacts from the data, or to ensure that the model is not using those spurious attributes present in the data for prediction. We tried the latter in the introductory experiments on ProtoPNet — identifying and removing the artifact prototypes. However, as we observed, this is not possible since the artifact is not always perceivable by the ProtoPNet heatmaps even if the artifact was learned by a particular prototype. Using our suggested method, we are now able to find the prototypes that are activated by the artifact. It was further discovered using PRP in the previous sections, that almost all the prototypes incorporate the artifact features, thus suggesting the entanglement of the artifact information within the whole network. Therefore, instead of pruning the artifact prototypes, we propose to detect the samples in the training dataset that activate the artifact prototypes, which can be subsequently removed from the training data set before retraining the ProtoPNet on the cleansed dataset.

Using PRP, we obtain k PRP maps corresponding to the artifact-containing class for each image, where k corresponds to the number of learned prototypes for that class. We can consider these PRP maps as k different views of the same image and can thus build on existing multi-view clustering methodologies to automatically cluster the training images and thereby discover clusters corresponding to artifact-containing images. In this work, we cluster the images into 2 clusters, an artifact and a clean data cluster.

To demonstrate the efficiency of PRP in detecting artifacts in the data, we test differ-

ent multi-view clustering methodologies on the LISA dataset with 50% and 20% Clever Hans features added to the stop sign images. We further use the same methodologies for backdoor detection thereby demonstrating PRP's efficiency in multiple artifact scenarios. We also compare our clustering approach with SpRAy, which performs spectral clustering analysis on single view LRP maps, and demonstrate that our approach is able to capture better information in PRP maps, especially in the setting with multiple views.

5.5.1. Clever Hans type artifacts in 50% training data

The accuracy for CH-50 for the artifacts in the stop sign class in 100% (artifact test) and 0% (clean test) data is shown in Table 1. As we can observe, the accuracy for the stop sign class drops from 100% to 94.6% when there is no artifact in the test data. From Figure 5, prototypes 4 and 9 can be considered as "artifact" prototypes according to ProtoPNet heatmaps. But as can be seen in Table 1, there is no effect on the artifact test accuracy when removing those two prototypes. The same holds when we remove the prototypes followed by a retraining of the model. On the other hand, a decrease in the accuracy for the clean test data is observed. This additionally supports our assertion of imprecise and even misleading information provided by ProtoPNet's heatmaps.

In order to obtain a clean data set, we aim to identify the samples that contain an artifact in the first place in order to remove them from the training set. Assuming that the information on whether an artifact is present in a data point is recognizable in the PRP maps, we cluster the PRP maps in two clusters. For comparison, we use a set of representative algorithms to cluster the data, including SpRAy [4], CoMVC [35] and Co-Reg [33]. We downsample the heatmaps to a size of 80x80, as this had negligible impact on the results and led to a reduced computation time.

The results for accuracy and F1-scores for the artifact cluster for different clustering methods are given in Table 2. We follow the experiments in [35] and train CoMVC for 100 epochs for 20 runs and report the results from the run resulting in the lowest unsupervised cost-function value. We repeat this 5 times and report mean and standard deviation.

As observed from Table 2, CoMVC is working very efficiently to separate the ar-

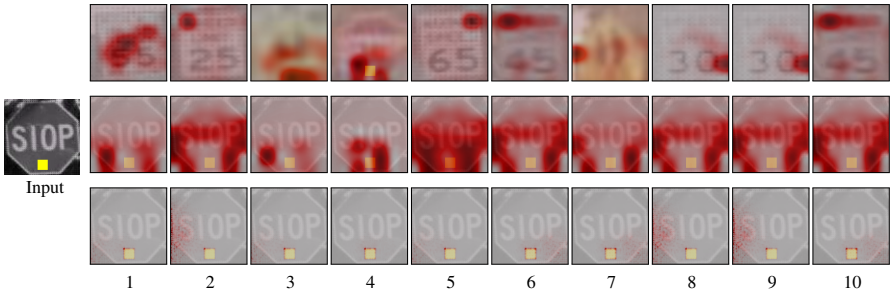


Figure 11: BD-15: Top row depicts the learned prototypes 1 to 10 for the speed limit class with the Backdoor in 15% of the stop sign images (labeled as speed limit), the middle row depicts the ProtoPNet's heatmaps corresponding to the respective prototypes for the test image shown on the left and the bottom row shows the corresponding PRP maps for the prototypes, which capture more precise information.

tifact images from the clean images. We also report the results for multi-view spectral clustering algorithm Co-Reg in Table 2. Although being more computationally expensive, Co-Reg is able to cluster the data effectively. Co-Reg always obtains an accuracy of above 94% in separating the artifact data, and thus prove to be highly successful in detecting the artifacts. CoMVC on the other hand performs with almost 100% accuracy when the artifact and non-artifact classes are balanced, i.e, in the current setting of CH-50.

To compare against the multi-view clustering approaches, we apply SpRAy [4], on the LRP maps for the true class (SpRAy-LRP) as well as PRP maps for the prototypes of the true class (SpRAy-PRP). For SpRAy-LRP, we compute LRP maps using the rules in Section 4.1, followed by LRP_ϵ for the last layer and a combination of relevance for all prototypes. More details are provided in the Appendix. Accordingly, we obtain one LRP map for each image, which is scaled down to 80x80 and flattened before applying SpRAy. For SpRAy-PRP, we combine the PRP map images by summing them across the channels and concatenating all 10 PRP maps for each image to get a 10x80x80 map. We then flatten it and apply SpRAy.

The results for both are shown in Table 2. As observed, SpRAy fails in clustering the artifacts in CH-50 data using both LRP and the concatenation of PRP maps. This behavior is expected since both SpRAy-LRP and SpRAy-PRP do not capture de-

pendencies among multiple views of the same objects as opposed to other multi-view clustering methodologies.

5.5.2. Clever Hans type artifact in 20% training data

In the following, we want to capture the scenarios when less Clever Hans artifacts are included in the training data. Therefore, we evaluate the efficiency of multi-view clustering methodologies on the unbalanced dataset CH-20. The stop sign class accuracy for artifact and clean test data is 99.7% and 95.8%, respectively. This depicts that the stop sign class is still affected by the Clever Hans effect.

Applying the multi-view clustering methodologies to this scenario, we report the accuracy and F1-score in Table 2. Results show that SiMVC is performing best with 97.99% accuracy, with comparable performance by almost all the other multi-view clustering methods. SpRAy fails again with a very low F1-scores of 0.04 and 0.08 on LRP and PRP maps, clustering almost all images into one cluster.

5.5.3. Backdoor type artifact in 15% training data

Similar to the experiments above, we examine the backdoor setting, using the generated BD-15 dataset. The prototypes and their corresponding heatmaps for the speed limit class are shown in Figure 11. The test accuracy for the case that the artifact is present in 100% of the stop sign test images is given in Table 3. Most of the stop sign images are now classified as speed limits and only 1% of the stop sign images are classified correctly.

The prototypes of the speed limit class, as learned by ProtoPNet, show that only one prototype has learned the backdoor artifact, while all the remaining 9 prototypes correspond to the speed limit class, as shown in Figure 11. As per ProtoPNet's explanations, removing prototype 4 of the speed limit class should solve the problem of backdoor attacks. We remove the prototype and retrain the last layer and report the accuracies in Table 3.

We can observe that removing the backdoor prototype has only a minor effect on the accuracy of the stop sign class, which increased from 1.0% to 6.5%. However, after retraining the last layer it again drops to only 2.5%. This behaviour of the network

Table 3: Accuracy on the artifact test (backdoor in 100% of the images in the stop sign class test data) and clean test data for BD-15, along with corresponding accuracies after removing the artifact prototype and retraining the last layer.

	BD-15	Remove prototype 4	Retraining last layer
Artifact Test	1.0%	6.5%	2.5%
Clean Test	96.0%	96.0%	95.6%

thus emphasizes the inherent learning of the backdoor artifact by the network, which is not limited to only learning a specific backdoor prototype, as incorrectly suggested by ProtoPNet visualizations. Here, the PRP explanations decode the behavior of the model as well - they indicate that almost all prototypes are activated by the artifact, even if those prototypes refer to the speed limit signs.

We therefore use multi-view clustering to clean the data of the backdoor feature and report the results in Table 2. SiMVC and CoMVC are still performing better than SpRAy-PRP with F1-scores of 0.60 and 0.57 respectively, as opposed to 0.02 F1-score of SpRAy-PRP. Although, SpRAy-LRP is performing well in this setting with a F1-score of 0.91, this is due to the fact that LRP maps consist of negative relevances from the stop sign class in addition to the positive relevances from the speed limit class. This helps in accentuating the difference between speed limit and backdoor stop sign images. Furthermore, all the multi-view spectral clustering-based algorithms are able to separate these clusters efficiently, with the best being Co-Reg with an accuracy of 99.42% and a F1-score of 0.98.

6. Conclusion

Considering the success of machine learning algorithms in diverse safety-critical applications, it is instrumental to verify the behavior of these models. In this work, we assess the faithfulness of the explanations provided by a well known self-explainable network, ProtoPNet, which has subsequently been utilized as a baseline for a variety of works [38, 23]. We provide an in-depth assessment of ProtoPNet's behavior in the presence of a range of artifacts. Our results indicate that, despite the attractiveness of ProtoPNet owing to its self-explaining characteristic, it is still very far from achiev-

ing the required quality of explanations. Considering this, we propose a model-aware method, PRP, to generate more precise and higher resolution prototypical explanations. These enhanced explanations help in uncovering more credible decision strategies, while keeping the self-explainability intact. We further show that these explanations are able to uncover the spurious artifact features learned by the model, which are then efficiently identified and removed via our proposed multi-view clustering strategy.

While PRP has been analysed extensively in this work, it needs to be explored further for variations of datasets as well as artifacts. So far, a limitation is the requirement of the manual analysis of clusters to distinguish the model and data heuristics despite the effective clustering performed by the proposed methodology. The behavior of the clustering further needs to be analysed in the future work in the presence of multiple artifacts per class. The design of explainable approaches with the inherent capability to leverage artifactual data in addition to clean data without capturing the artifact features would be ideal instead of removing the data and is therefore a main focus of future work. Finally, the benefit of using PRP in combination with other prototypical self-explainable models will be explored further in the future work.

The insights obtained in this work highlight the importance of evaluating the quality of self-explaining machine learning approaches and will pave the way towards the development of more robust and precise models, thereby increasing their trustworthiness.

Acknowledgment

This work was financially supported by the Research Council of Norway (RCN), through its Centre for Research-based Innovation funding scheme (Visual Intelligence, grant no. 309439), and Consortium Partners. The work was further partially funded by RCN FRIPRO grant no. 315029 and RCN IKTPLUSS grant no. 303514. Moreover, the work was partly supported by the German Ministry for Education and Research through the third-party funding project Explaining 4.0 (ref. 01IS20055).

Appendix

ProtoPNet: Cost function

The overall cost function for ProtoPNet is:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CE}} + \lambda_{\text{clst}}\mathcal{L}_{\text{clst}} + \lambda_{\text{sep}}\mathcal{L}_{\text{sep}} \tag{7}$$

$\mathcal{L}_{\text{CE}}$ is the cross entropy (CrsEnt) loss, $\mathcal{L}_{\text{clst}}$ is the cluster loss and $\mathcal{L}_{\text{sep}}$ is the separation loss, defined as:

$$\mathcal{L}_{\text{CE}} = \min_{W} \frac{1}{N} \sum_{i=1}^{N} \text{CrsEnt}(\hat{\mathbf{y}}_i, \mathbf{y}_i) \tag{8}$$

$$\mathcal{L}_{\text{clst}} = \frac{1}{N} \sum_{i=1}^{N} \min_{m:\mathbf{p}_m \in \mathbf{P}_{\mathbf{y}_i}} \min_{\widetilde{\mathbf{z}}} ||\widetilde{\mathbf{z}} - \mathbf{p}_m||_2^2 \tag{9}$$

$$\mathcal{L}_{\text{sep}} = -\frac{1}{N} \sum_{i=1}^{N} \min_{m:\mathbf{p}_m \notin \mathbf{P}_{\mathbf{y}_i}} \min_{\widetilde{\mathbf{z}}} ||\widetilde{\mathbf{z}} - \mathbf{p}_m||_2^2 \tag{10}$$

where N are the total number of training images, $\mathbf{y}_i$ is the true label for image i, $\hat{\mathbf{y}}_i$ is the predicted label, W represents the learnable parameters of the whole network, $\mathbf{P}_{\mathbf{y}_i}$ are all the prototypes belonging to class $\mathbf{y}_i$ and $\widetilde{\mathbf{z}}$ are the patches of the convolutional output which are of the same size as the prototypes.

SpRAy-LRP

For SpRAy based on LRP maps, we first backpropagate the output relevances i.e, class scores to the similarity score layer. We follow the $\mathbf{LRP}_{CMP}$ rule and use the LRP_{ϵ} rule [29]:

$$\mathbf{R}_{i \leftarrow j}^{(l,l+1)} = \frac{z_{ij}}{z_j + \epsilon \,.\, \text{sign}(z_j)} \mathbf{R}_j^{(l+1)} \tag{11}$$

For the rest of the network, the rules for PRP are used. Considering that we are now computing relevance corresponding to all the prototypes, we combine them to get the relevance at $CONV$ layer as:

$$\mathbf{R}_{ijc}^{(CONV,AM)} = \sum_{m=1}^{n} \mathbf{R}_{mijc}^{(CONV,AM)} \tag{12}$$

LISA 5 class dataset

Table 4: Combination of classes from LISA dataset for 5-class CH-100, CH-50, CH-20 and BD-15 datasets.

Restriction signs	noRightTurn, keepRight, thruMergeLeft, thruMergeRight, thruTrafficMergeLeft, doNotPass, noLeftTurn, doNotEnter, rightLaneMustTurn
Speed limits	speedLimit40, speedLimit25, speedLimit35, speedLimit50, speedLimit45, truckSpeedLimit55, speedLimit65, speedLimit55, speedLimit30, speedLimit15, schoolSpeedLimit25
Stop signs	stopAhead, stop
Warning signs	turnLeft, signalAhead, zoneAhead25, school, curveLeft, pedestrianCrossing, curveRight, rampSpeedAdvisory50, rampSpeedAdvisoryUrdbl, dip, rampSpeedAdvisory40, merge, turnRight, slow, roundabout, speedLimitUrdbl, zoneAhead45, intersection, laneEnds, rampSpeedAdvisory45, rampSpeedAdvisory20, rampSpeedAdvisory35, addedLane
Yield signs	yield, yieldAhead

References

[1] C. Barata, M. E. Celebi, J. S. Marques, Explainable skin lesion diagnosis using taxonomies, Pattern Recognition 110 (2021) 107413.

[2] A. I. Aviles-Rivero, P. Sellars, C.-B. Schnlieb, N. Papadakis, Graphxcovid: Explainable deep graph diffusion pseudo-labelling for identifying covid-19 on chest x-rays, Pattern Recognition 122 (2022) 108274.

[3] E. Tjoa, C. Guan, A survey on explainable artificial intelligence (xai): Toward medical xai, IEEE TNNLS 32 (11) (2021) 4793–4813.

[4] S. Lapuschkin, S. Wäldchen, A. Binder, G. Montavon, W. Samek, K.-R. Müller, Unmasking clever hans predictors and assessing what machines really learn, Nature Communications 10 (1).

[5] B. Chen, W. Carvalho, N. Baracaldo, H. Ludwig, B. Edwards, T. Lee, I. Molloy, B. Srivastava, Detecting backdoor attacks on deep neural networks by activation clustering, in: SafeAI@AAAI, 2019.

[6] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Mller, W. Samek, On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation, PLOS ONE 10 (7) (2015) 1–46.

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

[8] C. Chen, O. Li, A. Barnett, J. Su, C. Rudin, This looks like that: Deep learning for interpretable image recognition, in: NeurIPS, 2019.

[9] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, A. Zisserman, The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results, http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[10] M. Ivanovs, R. Kadikis, K. Ozols, Perturbation-based methods for explaining deep neural networks: A survey, Pattern Recognition Letters 150 (2021) 228–234.

[11] M. T. Ribeiro, S. Singh, C. Guestrin, "why should i trust you?": Explaining the predictions of any classifier, KDD '16, 2016, p. 11351144.

[12] S. M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, 2017, p. 47684777.

[13] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, D. Batra, Grad-cam: Visual explanations from deep networks via gradient-based localization, in: ICCV 2017, pp. 618–626.

[14] J. Sun, S. Lapuschkin, W. Samek, Y. Zhao, N. Cheung, A. Binder, Explanation-guided training for cross-domain few-shot classification, in: ICPR, 2021.

[15] X. Li, X. Song, T. Wu, Aognets: Compositional grammatical architectures for deep learning, CVPR 2019.

[16] V. Mnih, N. Heess, A. Graves, K. Kavukcuoglu, Recurrent models of visual attention, in: NeurIPS, 2014, p. 22042212.

[17] Q. Zhang, Y. N. Wu, S. Zhu, Interpretable convolutional neural networks, in: CVPR, 2018, pp. 8827–8836.

[18] Z. Chen, Y. Bei, C. Rudin, Concept whitening for interpretable image recognition, Nature Machine Intelligence 2 (12) (2020) 772782.

[19] J. Parekh, P. Mozharovskyi, F. dAlché-Buc, A framework to learn with interpretation, in: Advances in Neural Information Processing Systems, Vol. 34, 2021, pp. 24273–24285.

[20] I. Rio-Torto, K. Fernandes, L. F. Teixeira, Understanding the decisions of cnns: An in-model approach, Pattern Recognition Letters 133 (2020) 373–380.

[21] D. Alvarez Melis, T. Jaakkola, Towards robust interpretability with self-explaining neural networks, in: S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, R. Garnett (Eds.), NeurIPS, 2018.

[22] S. Saralajew, L. Holdijk, M. Rees, E. Asan, T. Villmann, Classification-by-components: Probabilistic modeling of reasoning over a set of components, in: NeurIPS, 2019.

[23] E. Kim, S. Kim, M. Seo, S. Yoon, Xprotonet: Diagnosis in chest radiography with global and local explanations, in: CVPR, 2021, pp. 15714–15723.

[24] K. Eykholt, I. Evtimov, E. Fernandes, B. Li, A. Rahmati, C. Xiao, A. Prakash, T. Kohno, D. Song, Robust physical-world attacks on deep learning visual classification, in: CVPR, 2018, pp. 1625–1634.

[25] L. Sixt, M. Granz, T. Landgraf, When explanations lie: Why many modified bp attributions fail, in: ICML, 2020.

[26] J. Luo, J. Zhao, B. Wen, Y. Zhang, Explaining the semantics capturing capability of scene graph generation models, Pattern Recognition 110 (2021) 107427.

[27] H. Zheng, E. Fernandes, A. Prakash, Analyzing the interpretability robustness of self-explaining models, ArXiv abs/1905.12429.

[28] A. Mogelmose, M. M. Trivedi, T. B. Moeslund, Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey, IEEE Transactions on Intelligent Transportation Systems (2012) 1484–1497.

[29] M. Kohlbrenner, A. Bauer, S. Nakajima, A. Binder, W. Samek, S. Lapuschkin, Towards best practice in explaining neural network decisions with lrp, in: IJCNN, 2020, pp. 1–7.

[30] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, K.-R. Mller, Explaining nonlinear classification decisions with deep taylor decomposition, Pattern Recognition 65 (2017) 211–222.

[31] K. Chaudhuri, S. M. Kakade, K. Livescu, K. Sridharan, Multi-view clustering via canonical correlation analysis, in: ICML, 2009, p. 129136.

[32] M. Cheng, L. Jing, M. K. Ng, Tensor-based low-dimensional representation learning for multi-view clustering, IEEE Transactions on Image Processing 28 (5) (2019) 2399–2414.

[33] A. Kumar, P. Rai, H. Daume, Co-regularized multi-view spectral clustering, in: J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, K. Q. Weinberger (Eds.), NeurIPS, 2011.

[34] L. Zong, X. Zhang, X. Liu, H. Yu, Weighted multi-view spectral clustering based on spectral perturbation, in: AAAI, 2018.

[35] D. J. Trosten, S. Lokse, R. Jenssen, M. Kampffmeyer, Reconsidering representation alignment for multi-view clustering, in: CVPR, 2021, pp. 1255–1265.

[36] M. Kampffmeyer, S. Lkse, F. M. Bianchi, L. Livi, A.-B. Salberg, R. Jenssen, Deep divergence-based approach to clustering, Neural Networks (2019) 91–101.

[37] S. Kolek, D. A. Nguyen, R. Levie, J. Bruna, G. Kutyniok, A rate-distortion framework for explaining black-box model decisions, in: xxAI - Beyond Explainable AI (2020), pp. 91–115.

[38] P. Hase, C. Chen, O. Li, C. Rudin, Interpretable image recognition with hierarchical prototypes, HCOMP (2019) 32–40.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Srishti Gautam is currently a Ph.D. student in the Machine Learning group at UiT The Arctic University of Norway. She previously received her MS by research degree from IIT Mandi, India in 2017. Her research interests include development of algorithms focusing on explainable AI and their application to medical and other real world data.

Marina M.-C. Höhne (née Vidovic) received her PhD from TU Berlin in 2017. Since 2020 she is leading the research group UMI lab dealing with explainable artificial intelligence at the Technical University of Berlin. Since 2021 she has a secondary employment as Associate Professor in the Machine Learning group at UiT The Arctic University of Norway.

Stine Hansen received her M.Sc. degree from UiT The Arctic University of Norway in 2018. She is currently a Ph.D. student in the Machine Learning group at UiT. Her research interests include medical image analysis and computer vision.

Robert Jenssen directs SFI Visual Intelligence (visual-intelligence.no). He is a Professor in the Machine Learning Group (machine-learning.uit.no) at UiT The Arctic University of Norway and an Adjunct Professor at the University of Copenhagen and at the Norwegian Computing Center.

Michael Kampffmeyer is an Associate Professor in the Machine Learning Group at UiT The Arctic University of Norway and a Senior Researcher at the Norwegian Computing Center. Research interests include the development of deep learning algorithms that learn from limited labeled data and their interpretability.