From Clustering to Cluster Explanations via Neural Networks

Jacob Kauffmann, Malte Esders, Lukas Ruff, Grégoire Montavon, Wojciech Samek, *Member, IEEE*, and Klaus-Robert Müller, *Member, IEEE*

***Abstract*— A recent trend in machine learning has been to enrich learned models with the ability to explain their own predictions. The emerging field of explainable AI (XAI) has so far mainly focused on supervised learning, in particular, deep neural network classifiers. In many practical problems, however, the label information is not given and the goal is instead to discover the underlying structure of the data, for example, its clusters. While powerful methods exist for extracting the cluster structure in data, they typically do not answer the question *why* a certain data point has been assigned to a given cluster. We propose a new framework that can, for the first time, explain cluster assignments in terms of input features in an efficient and reliable manner. It is based on the novel insight that clustering models can be rewritten as neural networks—or "neuralized." Cluster predictions of the obtained networks can then be quickly and accurately attributed to the input features. Several showcases demonstrate the ability of our method to assess the quality of learned clusters and to extract novel insights from the analyzed data and representations.**

***Index Terms*— Explainable machine learning, *k*-means clustering, neural networks, "neuralization," unsupervised learning.**

Manuscript received 4 May 2020; revised 10 December 2021 and 22 April 2022; accepted 19 June 2022. This work was supported by the German Ministry for Education and Research under Grant 01IS14013A-E, Grant 01GQ1115, Grant 01GQ0850, Grant 01IS18025A, Grant 031L0207D, and Grant 01IS18037A; and in part by the German Research Foundation (DFG) in the DAEDALUS Graduate School and the Math+: Berlin Mathematics Research Center (EXC 2046/1) under Project 390685689. The work of Klaus-Robert Müller was supported in part by the Institute of Information and Communications Technology Planning and Evaluation (IITP) Grants, Korea Government (MSIT), under Grant 2019-0-00079; in part by the Artificial Intelligence Graduate School Program, Korea University, under Grant 2022-0-00984; and in part by the Development of Artificial Intelligence Technology for Personalized Plug-and-Play Explanation and Verification of Explanation. *(Corresponding authors: Grégoire Montavon; Klaus-Robert Müller.)*

Jacob Kauffmann and Malte Esders are with the Machine Learning Group, Berlin Institute of Technology, 10587 Berlin, Germany.

Lukas Ruff is with Aignostics, 10117 Berlin, Germany.

Grégoire Montavon is with the Machine Learning Group, Berlin Institute of Technology, 10587 Berlin, Germany, and also with the Berlin Institute for the Foundations of Learning and Data, 10587 Berlin, Germany (e-mail: gregoire.montavon@tu-berlin.de).

Wojciech Samek is with the Department of Electrical Engineering and Computer Science, Berlin Institute of Technology, 10587 Berlin, Germany, also with the Department of Artificial Intelligence, Fraunhofer Heinrich Hertz Institute, 10587 Berlin, Germany, and also with the Berlin Institute for the Foundations of Learning and Data, 10587 Berlin, Germany (e-mail: wojciech.samek@hhi.fraunhofer.de).

Klaus-Robert Müller is with the Machine Learning Group, Berlin Institute of Technology, 10587 Berlin, Germany, also with the Berlin Institute for the Foundations of Learning and Data, 10587 Berlin, Germany, also with the Department of Artificial Intelligence, Korea University, Seoul 136-713, South Korea; and also with the Max Planck Institut für Informatik, 66123 Saarbrücken, Germany (e-mail: klaus-robert.mueller@tu-berlin.de).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TNNLS.2022.3185901.

Digital Object Identifier 10.1109/TNNLS.2022.3185901

I. Introduction

CLUSTERING is an important class of unsupervised learning models that aims to reflect the intrinsic heterogeneities of common data generation processes [1]–[4]. Natural cluster structures are observed in a variety of contexts from gene expression [5] and ecosystems composition [6] to textual data [7]. Methods that can accurately identify the cluster structure have thus been the object of sustained research over the past decades [8]. Basic techniques, such as *k*-means [9], have been extended to operate in kernel feature spaces [10], [11], or on the representations built by a deep neural network [12]–[15].

Due to the ever-growing complexity of ML models and their use in increasingly sensitive applications, it has become crucial to endow these models with the capability to explain their own predictions in a way that is interpretable for a human. Explainable AI (XAI) has emerged as an important direction for machine learning, and excellent results have been reported in selected tasks, such as explaining the predictions of popular DNN classifiers [16]–[20].

In this article, we bring these newly developed explanation capabilities to clustering, a highly needed functionality, considering that in the first place one of the main motivations for performing a clustering is knowledge discovery. Especially in high-dimensional feature space, a clustering for knowledge discovery can only provide a few prototypical data points for each cluster. Such prototypes, however, do not reveal which features made them prototypical. Instead, we would like to let the clustering model explain itself in terms of the very features that have contributed to the cluster assignments. To the best of our knowledge, our work is the first-ever attempt to systematically and comprehensively obtain such explanations. Specifically, we are able to supply explanations of *why* each individual point is clustered in the way it is.

The method we propose, puts forward the novel insight that a broad range of clustering models can be rewritten, *without retraining*, as *functionally equivalent* neural networks, which then serve as a backbone to guide the explanation process. Technically, we suggest applying the following two steps: 1) the cluster model is "*neuralized*" by rewriting it as a functionally equivalent neural network with standard detection/pooling layers. 2) Cluster assignments formed at the output of the neural network are then *propagated* backwards using an LRP-type procedure (cf. [17], [21], [22]) until the input variables (e.g., pixels or words) are reached.

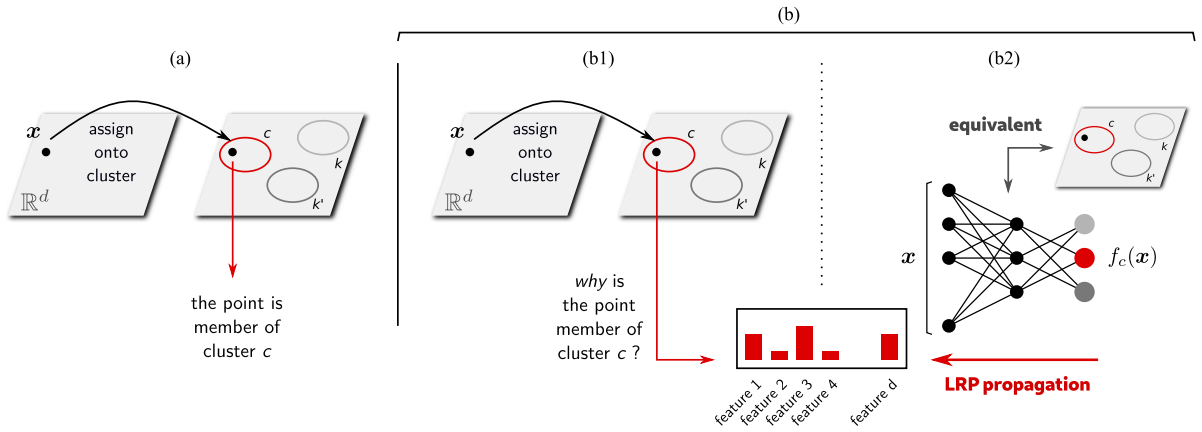

Fig. 1. From clustering to cluster explanations via neural networks. (a) Standard clustering scenario where data are assigned onto clusters according to the clustering model. (b) Overview of our contributions. (b1) We enrich the cluster assignment with an explanation highlighting what input features mostly contribute to the cluster decision. (b2) We achieve this technically by observing that the clustering decision can be rewritten as a neural network (neuralization), enabling fast and robust explanations via the LRP technique (propagation).

The proposed "neuralization-propagation" procedure (or short, NEON) is tested on a number of datasets and clustering models, including recent deep clustering models, such as SCAN [23]. Each time, NEON accurately explains cluster assignments and extracts useful insights. Experiments also demonstrate the practical value of our two-step approach compared with a potentially simpler one-step approach without neuralization. Our contributions can be summarized as follows.

1) Introduction of XAI to clustering, specifically, explanation of the assignment of individual data points onto clusters, in terms of input features.
2) Formulation of the clustering decisions for a broad range of clustering models as being functionally equivalent neural networks, thus enabling the application of state-of-the-art XAI techniques to these models.
3) Theoretical embedding of our neuralization-propagation approach to explaining clustering, specifically providing an interpretation of our approach, for special cases, in terms of Shapley values.
4) Demonstration of the benefit of bringing XAI to clustering showcased for two real-world examples and extensive quantitative validation of our proposed explanation method.

Fig. 1 shows a cartoon of our contributions in order to provide the general underlying intuition to the reader. We stress that our method applies to many popular clustering algorithms and is a generic blueprint as it does neither rely on predesigned interpretability structures nor algorithms, nor any retraining. This will prove useful in the future for shedding new light on existing cluster-based typologies used, e.g., in computational biology [24], [25] or consumer data [26], which researchers and practitioners have started to use increasingly to support their scientific reasoning and to take insightful decisions.

A. Related Work

So far, research on explanation methods has been overwhelmingly focused on the case of supervised learning. Methods based on the gradient [27]–[29], local perturbations [16], [30], or surrogate functions [18] do not make specific assumptions about the structure of the model and are, thus, applicable to a wide range of classifiers. Other methods require the classifier to have a neural network structure and apply a purposely designed backward propagation pass [17], [21], [22], [31]–[33] to produce accurate explanations at low computational cost. While recent work has extended the principle to other types of models, such as one-class support vector machines (SVMs) [34], LSTM networks [35], or graph neural networks [36], the method we propose here contributes by offering a solution to the so far unsolved problem of explaining cluster assignments.

Note that a few cluster interpretability techniques have so far been based on surrogate decision trees [37]–[41], where the decision tree is trained to approximate the k-means clustering as closely as possible, and where the cluster assignment is then interpreted using XAI techniques specific to decision trees. With such a surrogate approach, the user typically has to tradeoff faithfulness to the original model against explainability.

Related to the connections we establish in this article between clustering models and neural networks, some works explore ways of merging the two in order to produce better, more flexible ML models. For example deep clustering approaches typically build a clustering objective on top of deep representations [12], [14], [15], [42], [43]. Other approaches, such as TELL [44], design the neural network in a way that simulates a clustering model, so that the learned neural networks solution can be interpreted as a clustering solution. Note that in all these works, the purpose is more to enhance a basic clustering model by providing the flexibility of neural network representation and training, whereas our work focuses on making existing popular clustering algorithms explainable.

Another set of related works focuses on the problem of *learning* a good clustering model, by identifying a subset of relevant features that support the cluster structure. Some methods identify relevant features by running the same clustering algorithm multiple times on different feature subsets [45]. Other approaches simultaneously solve feature selection and

clustering by defining a joint objective function to be minimized [46]. While feature selection can identify the set of features required to represent the overall cluster structure, our work builds up by identifying among those features which ones are truly responsible for a given cluster or a given cluster assignment.

Further related works focus on quantitatively validating clustering solutions. Examples of validation metrics are compactness/separation of clusters [47], cluster stability under resampling/perturbations [48], [49], or purity, i.e., the absence of examples with different labels in the same cluster [50]. Our work enhances the validation of clustering models by producing human-interpretable feedback, a critical step to identifying whether cluster assignments are supported by meaningful features or by what the user would consider to be artifacts.

Finally, user interfaces have been developed to better navigate cluster structures, as they occur, e.g., in biology applications [51], [52]. Also, the use of prototypes has been proposed to visualize deep image clustering models [15] or explain kernel methods for property prediction of chemical compounds [53]. While these works produce useful and informative visualizations which may help to guide the process of clustering, our approach contributes by answering the precise question "*why* a given data point is assigned to a particular cluster."

II. Explaining K-Means Cluster Assignments

The k-means algorithm [9] is one of the best known approaches to clustering and is used in many scientific and industrial applications (e.g., [54]–[56]). This section presents our neuralization-propagation approach for explaining a k-means cluster assignment in terms of input features. Due to the simplicity of the k-means model, this section also has a tutorial purpose. More complex and powerful clustering models based on kernels [11], deep neural networks [12], [14], or more general clustering techniques, are discussed in Sections III–V.

The k-means algorithm finds a set of centroids that minimizes the total squared distance between each data point and their nearest centroid. The k-means model assigns points to clusters based on their distance to each centroid $\boldsymbol{\mu}_k \in \mathbb{R}^d$, specifically the model assigns a point $\boldsymbol{x} \in \mathbb{R}^d$ to cluster c if

$$\forall_{k \neq c} : \ \|\boldsymbol{x} - \boldsymbol{\mu}_c\|^2 < \|\boldsymbol{x} - \boldsymbol{\mu}_k\|^2. \tag{1}$$

In principle, it is conceivable to use XAI techniques, such as prediction difference analysis (PDA) [16], [57] or LIME [18], as they apply out-of-the-box to *any* model or decision function. However, these approaches require evaluating the function multiple times to test for the effect of each input dimension. This can become slow when the data is high dimensional, e.g., when clustering images or gene expression data [25]. Also, the local perturbation may not faithfully depict the overall contribution of a feature to the clustering decision, especially if multiple features need to be perturbed in order to affect the decision.

In the context of supervised learning, more efficient XAI techniques have been proposed, which rely on a model that induces the decision function, and from which meaningful gradient information and intermediate representations can be extracted. Such methods include, among others, integrated gradients (IGs) [29], or layerwise relevance propagation (LRP) [17], [22], [58]. The LRP method, in particular, leverages the neural network structure of the prediction to produce a robust explanation in the order of a single forward/backward pass. The LRP method was used in a wide range of applications (e.g., [22], [36], [59]–[64]), and can be embedded in the framework of deep Taylor decomposition [31].

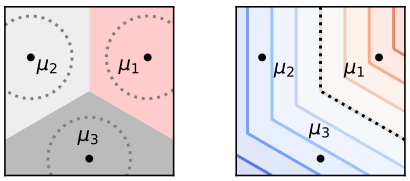

Fig. 2. Left: Decision function of a k-means clustering model with centroids $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\mu}_3$. Data points in the region highlighted in red are assigned to the cluster $c = 1$. Right: Contour plot of the function $f_c(\boldsymbol{x})$ for the cluster $c = 1$.

A. Neuralization of the Cluster Assignment

In order to bring these efficient XAI techniques to clustering, we propose to enrich the clustering decision function $g_c(\boldsymbol{x})$ with a neural network model. The latter is designed to exactly replicate the cluster assignments of the original clustering model and is more amenable to explainability. Furthermore, we also require that such a neural network model is obtained readily from the cluster solution (i.e., the centroids) without incurring any additional training step. We call the process of obtaining such a neural network "neuralization."

Proposition 1: The decision function of (1) can be reproduced by a two-layer neural network composed of a standard linear layer and a (min-)pooling layer

(Neuralized k-Means):

$$h_k = \boldsymbol{w}_k^\top \boldsymbol{x} + b_k \quad \text{(layer 1)}$$
$$f_c = \min_{k \neq c} \{h_k\} \quad \text{(layer 2)}$$

where $\boldsymbol{w}_k = 2(\boldsymbol{\mu}_c - \boldsymbol{\mu}_k)$ and $b_k = \|\boldsymbol{\mu}_k\|^2 - \|\boldsymbol{\mu}_c\|^2$, and assigning to cluster c if $f_c(\boldsymbol{x}) > 0$.

(See Appendix A of the Supplementary Material for the derivation). The first layer corresponds to a collection of linear functions aligned with the different cluster centroids. The min pooling selects which linear function is active at a given location. These two layers together build a piecewise linear function. A simple 2-D example with three clusters is shown in Fig. 2. We observe that the neural network output $f_c(\boldsymbol{x})$ (right) exactly reproduces the true cluster decision boundary, specifically, the Voronoi partition associated with the given k-means model (left).

This neural network can be also interpreted in neuroscientific terms as the alternation of "simple cells" and "complex cells" [65], or "executive organs" and "restoring organs" in automata theory [66]. We also note that earlier works have already linearized elements of the cluster model, such as the

square distance for the purpose of training [44]. Here, our contribution differs by extracting a piecewise linear view of the *whole* model, and additionally, identifying a neural network structure for this piecewise linear form. We provide similar neuralization results for the soft k-means case, as well as a probabilistic interpretation, in Appendix E of the Supplementary Material. We will also study more complex neuralization scenarios in Sections III and IV when considering kernel-based clustering and deep clustering.

B. Propagation of the Cluster Assignment

So far, we have rewritten the k-means decision function for each cluster as a neural network. This initial step gives access to a broader range of explanation techniques, such as IGs [29] or LRP [17], [67]. The LRP technique, in particular, leverages the neural network structure to produce a robust explanation in a single forward/backward pass. Unlike the standard gradient propagation pass which provides a highly localized view of the function, LRP applies propagation rules that redistribute the quantity to explain from layer to layer. These rules are purposely designed for the task of explanation. LRP ensures certain desirable properties of an explanation, such as conservation of predicted evidence and local continuity of explanations [17], [67].

Let us start with the output of the neural network f_c, which we wish to attribute as a first step to neurons in the intermediate layer $(h_k)_k$, by propagating through the min function. Similar to [34], we follow a min-take-most (MTM) strategy, where smallest inputs to that function receive the largest share of the quantity to redistribute; in particular, we apply the propagation rule

$$R_k = \frac{\exp(-\beta h_k)}{\sum_{k \neq c} \exp(-\beta h_k)} f_c \tag{2}$$

where R_k is the "relevance" of neuron h_k to the cluster assignment f_c, and where $\beta \in \mathbb{R}^+$ is a stiffness hyperparameter. The stiffness parameter interpolates between a uniform redistribution strategy ($\beta = 0$) and a min-take-all strategy ($\beta \to \infty$). Note that compared with these two extreme cases, our approach allows us to contextualize the explanation (i.e., not redistributing on clusters of competitors that are too far, and therefore, irrelevant), and at the same time, ensures continuity of the explanation as we transition from one nearest cluster competitor to another. We propose to set this parameter according to the simple heuristic

$$\beta = \mathbb{E}[f_c]^{-1} \tag{3}$$

where the expectation is computed over the whole dataset. In other words, considering f_c to be a "typical" score in the pool, we want the stiffness parameter to be inversely proportional to it.

We now consider how to further redistribute the intermediate relevance scores R_k to the input layer, where the dimensions correspond to observed quantities that are assumed to be interpretable by the user. To achieve this, we propose the LRP propagation rule

$$R_i = \sum_{k \neq c} \frac{(x_i - m_{ik}) \cdot w_{ik}}{\sum_i (x_i - m_{ik}) \cdot w_{ik}} R_k \tag{4}$$

where $\boldsymbol{m}_k = (\boldsymbol{\mu}_c + \boldsymbol{\mu}_k)/2$ is the midpoint between the centroids of the cluster of interest and the competitor. In other words, we attribute on dimensions where the input activation relative to the midpoint, $\boldsymbol{x} - \boldsymbol{m}_k$, matches the model response $\boldsymbol{w}_k$.

It can be noted that the proposed propagation rules ensure a certain number of desirable properties of an explanation, in particular, it satisfies the conservation property $\sum_i R_i = f_c(\boldsymbol{x})$, it preserves the continuity of $f_c(\boldsymbol{x})$, and it is invariant to any translation of the clustering in input space.

C. Theoretical Embedding

We provide further theoretical support for the rules in (2) and (4) by showing that their application produces, for special cases, explanations that coincide with the Shapley value. The Shapley value [30], [68], [69], originally proposed in the context of game theory, is an axiomatic solution to the problem of attributing the value of a coalition of players to individual players in the coalition. For our comparison, we interpret the set of players as the individual input features (or activations) and the withdrawal of a player from the coalition as replacing the corresponding feature value x_i by some reference value $\widetilde{x}_i$.

Proposition 2: Redistribution performed by (2) with parameter $\beta = 0$, corresponds to the Shapley value of the function $f_c(\boldsymbol{h})$ with the reference point $\widetilde{\boldsymbol{h}} = \boldsymbol{0}$.

(The proof is given in Appendix B of the Supplementary Material). The parameter $\beta = 0$ corresponds to a uniform redistribution of f_c to the cluster competitors. The corresponding reference point $\widetilde{\boldsymbol{h}} = \boldsymbol{0}$ can be interpreted as the image of a point $\widetilde{\boldsymbol{x}}$ in input space that is equidistant from all cluster centroids. (Note that this point may not exist in low-dimensional spaces).

Proposition 3: When the number of clusters is equal to 2, the model reduces to $f_c(\boldsymbol{x}) = \boldsymbol{w}_k^\top \boldsymbol{x} + b_k$, and redistribution by (2) and (4) corresponds to the Shapley value of the function $f_c(\boldsymbol{x})$ with the reference point $\widetilde{\boldsymbol{x}} = \boldsymbol{m}_k$.

(See Appendix B of the Supplementary Material for the proof). In other words, the explanation coincides with Shapley values with the reference point $\widetilde{\boldsymbol{x}}$ chosen at the midpoint between the clusters centroids $\boldsymbol{\mu}_k$ and $\boldsymbol{\mu}_c$. Such a reference point is a natural choice for explaining why a point is a member of cluster c and not of cluster k.

III. Extension to Kernel K-Means

The standard k-means clustering algorithm has strong limitations in terms of representation power, as it only allows to represent clusters that are pairwise linearly separable. The kernel k-means model [11] is a straightforward extension of k-means where the data is first mapped to a feature space via some map $\boldsymbol{x} \mapsto \Phi(\boldsymbol{x})$ induced by some kernel function $\mathbb{K}(\boldsymbol{x}, \boldsymbol{u})$. The decision function implemented by kernel k-means is given by

$$\forall_{k \neq c} : \ \|\Phi(\boldsymbol{x}) - \boldsymbol{\mu}_c\|^2 < \|\Phi(\boldsymbol{x}) - \boldsymbol{\mu}_k\|^2 \tag{5}$$

where the centroids are also defined in feature space.

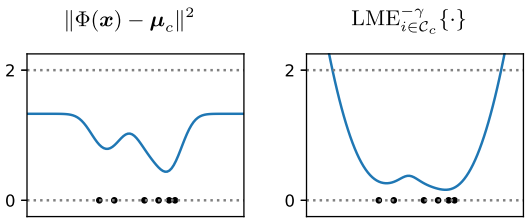

Fig. 3. Distance between some data point $\boldsymbol{x}$ and a cluster c depicted as a collection of black dots. The distance is either computed in feature space, or using the soft min pooling of (6).

If we were to apply the same explanation framework as in Section II, we would obtain an explanation in terms of dimensions of the feature space, and we would then need to further backpropagate through the feature map (FM) Φ. While this is technically possible [e.g., for a Gaussian kernel $\mathbb{K}(\boldsymbol{x},\boldsymbol{u}) = \exp(-\gamma\,\|\boldsymbol{x}-\boldsymbol{u}\|^2)$, one can use random approximations of the FM], we consider instead a more intuitive formulation, specific to the Gaussian kernel case, where the distance to a particular cluster is modeled by a soft minimum over distance to the cluster members. Specifically, we consider in place of (5) the decision function

$$\forall_{k\neq c}:\ \underset{i\in\mathcal{C}_c}{\mathrm{LME}^{-\gamma}}\ \left\{\|\boldsymbol{x}-\boldsymbol{u}_i\|^2\right\} < \underset{j\in\mathcal{C}_k}{\mathrm{LME}^{-\gamma}}\ \left\{\|\boldsymbol{x}-\boldsymbol{u}_j\|^2\right\} \quad (6)$$

where $(\boldsymbol{u}_i)_i$ and $(\boldsymbol{u}_j)_j$ are sets of data points (or support vectors) representing the two clusters, $\mathcal{C}_c, \mathcal{C}_k \subset \mathbb{N}$ are the nonoverlapping sets of indices of support vectors that represent these clusters, where $\mathrm{LME}^{-\gamma}$ denotes a generalized F-mean with $F(t) = e^{-\gamma t}$, that is,

$$\underset{i\in\mathcal{C}}{\mathrm{LME}^{-\gamma}}\ \{s_i\} = -\frac{1}{\gamma}\log\left(\frac{1}{|\mathcal{C}|}\sum_{i\in\mathcal{C}}\exp(-\gamma\, s_i)\right). \quad (7)$$

The latter can be interpreted as a soft min pooling and it converges to a hard min pooling when $\gamma \to \infty$.

The two distance measures on which the decision functions of (5) and (6) are based are illustrated for some toy 1-D cluster c composed of six data points in Fig. 3.

While the two functions clearly differ, one can also observe that they build comparable level sets. In fact, we show in Proposition 4 that these two measures of distance are essentially the same up to some monotonous nonlinear transformation, thereby leading to the same decision function.

Proposition 4: Let $\boldsymbol{\mu}_c = (1/Z_c)\sum_{i\in\mathcal{C}_c}\Phi(\boldsymbol{u}_i)$, where Φ is some FM associated with the Gaussian kernel $\mathbb{K}(\boldsymbol{x},\boldsymbol{u}) = \exp(-\gamma\,\|\boldsymbol{x}-\boldsymbol{u}\|^2)$ and Z_c is a normalization factor. The two distance functions appearing in (5) and (6) can be related as

$$\underset{i\in\mathcal{C}_c}{\mathrm{LME}^{-\gamma}}\ \left\{\|\boldsymbol{x}-\boldsymbol{u}_i\|^2\right\} = g_c(\|\Phi(\boldsymbol{x})-\boldsymbol{\mu}_c\|^2) \quad (8)$$

where g_c is a monotonically increasing function defined as

$$g_c(\xi) = \gamma^{-1}\mathrm{Li}_1(\xi/2+\Delta_c) + \gamma^{-1}H_c \quad (9)$$

with Li_1 is the polylogarithm of order 1, $\Delta_c = (1-\|\boldsymbol{\mu}_c\|^2)/2$, and $H_c = \log(|\mathcal{C}_c|/Z_c)$.

A proof is given in Appendix C of the Supplementary Material. Formally, equivalence between the two decision functions [(5) and (6)] is ensured when the function g_c does not depend on the choice of cluster c. When choosing the normalization factor $Z_c = |\mathcal{C}_c|$ (standard kernel k-means), the term H_c vanishes but the term Δ_c remains, and the converse happens if setting $\|\boldsymbol{\mu}_c\| = 1$, i.e., $Z_c = \|\sum_{i\in\mathcal{C}_c}\Phi(\boldsymbol{u}_i)\|$ (spherical kernel k-means). In practice, both terms remain near zero if we observe that each cluster is equally heterogeneous and has consequently the same norm in feature space. In that case, the two decision boundaries become equivalent. An advantage of the latter decision function is that it can be exactly reproduced by a neural network.

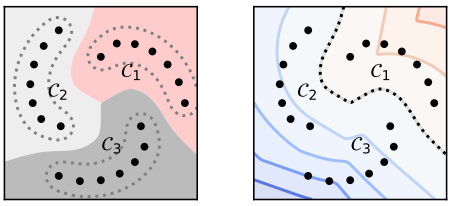

Fig. 4. Left: Partition implemented by a kernel k-means clustering with three clusters supported by seven support vectors each. Right: Neural network output $f_c(\boldsymbol{x})$ associated with the first cluster.

Proposition 5: The decision function in (6) can be reproduced by a four-layer neural network composed of a linear layer followed by three pooling layers

(Neuralized Kernel k-Means):

$$\begin{aligned} h_{ijk} &= \boldsymbol{w}_{ij}^\top\boldsymbol{x} + b_{ij} && \text{(layer 1)}\\ h_{jk} &= \underset{i\in\mathcal{C}_c}{\mathrm{LME}^{\gamma}}\ \{h_{ijk}\} && \text{(layer 2)}\\ h_{k} &= \underset{j\in\mathcal{C}_k}{\mathrm{LME}^{-\gamma}}\ \{h_{jk}\} && \text{(layer 3)}\\ f_c &= \min_{k\neq c}\ \{h_k\} && \text{(layer 4)} \end{aligned}$$

where $\boldsymbol{w}_{ij} = 2\cdot(\boldsymbol{u}_i - \boldsymbol{u}_j)$ and $b_{ij} = \|\boldsymbol{u}_j\|^2 - \|\boldsymbol{u}_i\|^2$, where LME^{γ} and $\mathrm{LME}^{-\gamma}$ can be interpreted as soft max pooling and soft min pooling, respectively, and assigning to cluster c if $f_c(\boldsymbol{x}) > 0$.

The proof is given in Appendix D of the Supplementary Material. An example showing the equivalence between the neural network output and (6) is given in Fig. 4.

This neural network we have proposed is depicted in Fig. 5 (middle) and can now be used to support the process of explanation. Because the network is again composed of linear and pooling layers, propagation rules proposed for the k-means case remain applicable. In particular, redistribution in pooling layers can be achieved using (2) (and switching the sign for the soft max-pooling case).[1] The directional redistribution in the first layer can be achieved using (4). However, we must handle the case where some relevance lands on a deactivated (or weakly activated) neuron h_{ijk}, as the latter does not provide directionality in input space. Such a special case can be handled by only propagating part of the relevance (and dissipating the rest), specifically, by performing the reassignment

$$R_{ijk} \leftarrow R_{ijk}\cdot(h_{ijk}/h_k). \quad (10)$$

[1]The relevance attributed to neuron h_{ijk} is, thus, given as

$$R_{ijk} = \frac{\exp(\gamma\, h_{ijk})}{\sum_{i\in\mathcal{C}_c}\exp(\gamma\, h_{ijk})}\cdot\frac{\exp(-\gamma\, h_{jk})}{\sum_{j\in\mathcal{C}_k}\exp(-\gamma\, h_{jk})}\cdot R_k.$$

The latter ensures that the relevance continuously converges to zero as the neuron h_{ijk} becomes deactivated.

In terms of computational cost, we note that the number of neurons in our neuralized k-means model grows quadratically with the number of support vectors per cluster, whereas the complexity of a simple evaluation of the decision function is linear with the number of support vectors. For NEON to remain computationally favorable, the number of support vectors must be kept small, typically, in the order of ten support vectors per cluster. Practical approaches to produce a limited number of support vectors include, e.g., reduced sets [70]–[72], vector quantization [73], or representing each cluster as a mixture model with finitely many mixture elements (we use this approach in Section VI-A). Alternatively, when for modeling purposes it is necessary to maintain a large number of support vectors per cluster, one can adopt a pruning strategy, where we only evaluate in the forward and backward pass the most relevant part of the network, i.e., neurons to which the min- and max-pooling functions in the network are effectively sensitive.

IV. Extension to Deep Clustering

Unlike kernel k-means, deep k-means makes use of an FM given explicitly as a sequence of layer-wise mappings $\Psi(\boldsymbol{x}) = \Psi_L \circ \cdots \circ \Psi_1(\boldsymbol{x})$, and the FM is typically learned via backpropagation to produce the desired cluster structure.

Various formulations of deep k-means have been proposed in the literature [12], [14], [15], [23], [74], [75], often achieving vastly superior performance compared with shallow clustering approaches. Clustering solutions produced by [14], [15] optimize a hard k-means objective based on distances in feature space. Using the same assignment model as for k-means, but this time in feature space, we decide for cluster c if

$$\forall_{k\neq c}: \ \|\Psi(\boldsymbol{x}) - \boldsymbol{\mu}_c\|^2 < \|\Psi(\boldsymbol{x}) - \boldsymbol{\mu}_k\|^2. \tag{11}$$

This lets us rewrite the full model as a the stacking of the L layers of the neural network Ψ with the neuralized k-means model defined in Proposition 1:

*(Neuralized Deep **k**-Means):*

$$\begin{aligned} \boldsymbol{a} &= \Psi_L \circ \cdots \circ \Psi_1(\boldsymbol{x}) && \text{(layers } 1,\ldots,L) \\ h_k &= \boldsymbol{w}_k^\top \boldsymbol{a} + b_k && \text{(layer } L+1) \\ f_c &= \min_{k\neq c} \{h_k\} && \text{(layer } L+2) \end{aligned}$$

where $\boldsymbol{w}_k = 2\cdot(\boldsymbol{\mu}_c - \boldsymbol{\mu}_k)$ and $b_k = \|\boldsymbol{\mu}_k\|^2 - \|\boldsymbol{\mu}_c\|^2$. The neural network is depicted in Fig. 5 (right). Note that beyond a simple application of standard k-means on top of a given layer, there have been many proposals for deep clustering.

Other quite popular formulations make use of a soft cluster assignment model, specifically, a softargmax model [23], [42], or a t-Student similarity model [12], [43]. These soft clustering approaches bring a probabilistic interpretation of cluster assignments and enable entropy-based optimization criteria. In the soft k-means models of [23] and [42], the data are first projected on some direction $\boldsymbol{\mu}_c$ associated with the cluster and mapped to a probability score using a softmax. Here, we first consider the explanation of the clustering outcome, in other words, we place the decision boundary at the location where there is as much evidence for the given cluster assignment as for the assignment onto the nearest competitor

$$\forall_{k\neq c}: \ p_c(\boldsymbol{x}) > p_k(\boldsymbol{x}) \quad \text{with} \quad p_c(\boldsymbol{x}) = \frac{\exp(\boldsymbol{\mu}_c^\top \boldsymbol{a})}{\sum_k \exp(\boldsymbol{\mu}_k^\top \boldsymbol{a})} \quad \text{and} \quad \boldsymbol{a} = \Psi(\boldsymbol{x}). \tag{12}$$

Proposition 6: The decision function of (12) can be expressed by the neural network

Neuralized Deep Soft Clustering (Relative):

$$\begin{aligned} \boldsymbol{a} &= \Psi_L \circ \cdots \circ \Psi_1(\boldsymbol{x}) && \text{(layers } 1,\ldots,L) \\ h_k &= \boldsymbol{w}_k^\top \boldsymbol{a} && \text{(layer } L+1) \\ f_c &= \min_{k\neq c} \{h_k\} && \text{(layer } L+2) \end{aligned}$$

where $\boldsymbol{w}_k = \boldsymbol{\mu}_c - \boldsymbol{\mu}_k$ and testing for $f_c \geq 0$. Furthermore, f_c has a probabilistic interpretation as the log-likelihood ratio $\log(p_c(\boldsymbol{x})/\max_{k\neq c}\{p_k(\boldsymbol{x})\})$.

A proof is given in Appendix E of the Supplementary Material. The solutions in [12] and [43] are also based on a soft-assignment model, where the exponential terms are replaced by t-Student distributions. The latter does not allow for a similar neural network reformulation as earlier; however, they still converge to hard k-means when the clusters become increasingly distant.

Alternatively, one can be interested in why an assignment onto a cluster exceeds a particular probability threshold. Specifically, we would like to explain the decision function

$$p_c(\boldsymbol{x}) > \theta \tag{13}$$

where the probability scores are defined in the same way as in (12), and where θ is some value between 0 and 1.

Proposition 7: The decision function of (13) can be expressed by the neural network

Neuralized Deep Soft Clustering (Absolute):

$$\begin{aligned} \boldsymbol{a} &= \Psi_L \circ \cdots \circ \Psi_1(\boldsymbol{x}) && \text{(layers } 1,\ldots,L) \\ h_k &= \boldsymbol{w}_k^\top \boldsymbol{a} + b_k && \text{(layer } L+1) \\ f_c &= \underset{k\neq c}{\text{LME}^{-1}}\{h_k\} && \text{(layer } L+2) \end{aligned}$$

where $\boldsymbol{w}_k = \boldsymbol{\mu}_c - \boldsymbol{\mu}_k$ and $b_k = -\log(N-1) + \log((1-\theta)/\theta)$, and testing for $f_c \geq 0$. Furthermore $f_c = \log(p_c(\boldsymbol{x})/(1 - p_c(\boldsymbol{x}))) + \log((1-\theta)/\theta)$, i.e., a log-likelihood ratio plus an offset.

A proof is given in Appendix E of the Supplementary Material. As for the k-means and kernel k-means cases, the MTM propagation rule can be applied to the top layer. For the last neuralized variant featuring the LME computation, one also needs to handle the case, where nonzero relevance scores R_k land on deactivated neurons ($h_k = 0$). To avoid this, we perform the reassignment $R_k \leftarrow R_k \cdot (h_k/f_c)$. For

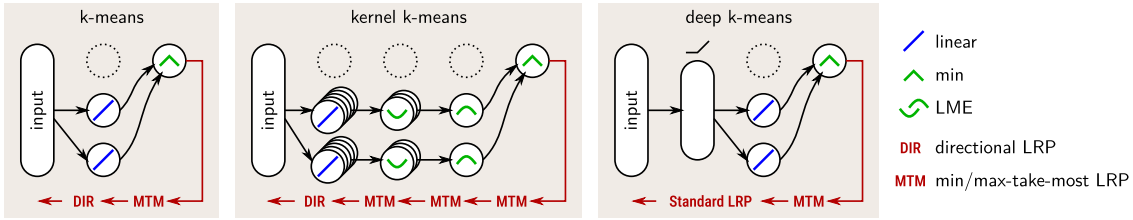

Fig. 5. Examples of clustering models whose cluster assignments can be explained with our NEON approach. The neuralized models, each of which can be expressed as combinations of detection layers and pooling layers, are depicted along with the propagation rules applied at each layer.

further propagation of relevance scores into the neural network, we notice that all layers up to layer $L+1$ form a standard neural network. Hence, propagation rules designed in the context of neural networks are applicable. For propagation rules specific to deep neural networks, we refer to the papers [58], [76], which cover in particular convolutional layers and LSTM blocks.

V. Extension to Any Clustering

Not all clusterings can be readily obtained by standard/kernel/deep k-means or combinations of them. Algorithms, such as DBSCAN [77], hierarchical agglomerative clustering [78], or spectral clustering [79], [80], are based on a different principle, and typically lead to different cluster solutions. For these clusterings, we observe, however, that the decision function they implement is typically based on evaluating distances between individual data points. Hence, the kernel k-means model we have proposed provides a natural surrogate for modeling the cluster assignment of these models. In particular, the identified four-layer architecture can be kept fixed, and the parameters (e.g., data point weightings) can be fine-tuned to fit the decision boundary. Once the model boundaries coincide, the model can be used in a second step to extract explanations. The same fine-tuning strategy can be used to handle cluster solutions that are not the sole result of a clustering algorithm but that have instead been curated by humans to match their expert knowledge.

Compared with a standard surrogate approach that would use a generic classifier to fit the cluster assignments, using a standard/kernel/deep k-means surrogate ensures that the needed adjustment is minimal, thereby preventing the decision strategy of the two models to become substantially different. In particular, one minimizes the risk of introducing a Clever Hans effect into the surrogate model (cf. [81]), or removing such Clever Hans effect. The risk would indeed be that the surrogate model yields a false interpretation (too optimistic or too pessimistic) of the original model's decision strategy.

VI. Applications

We have proposed to extend XAI to clustering and have contributed the neuralization-propagation technique (NEON) to efficiently extract these explanations. In the following, we demonstrate on three showcase examples how one benefits in multiple ways from enriching cluster assignments with explanations.

A. Better Validation of a Clustering Model

The following showcase demonstrates how an explanation of cluster assignments can serve to produce a rich and nuanced assessment of cluster quality that goes beyond conventional metrics, such as cluster purity.

We consider for this experiment the 20 newsgroups dataset [82] that contains messages from 20 public mailing lists, recorded around the year 1996. Headers, footers, and quotes are removed from the messages. Each document $\mathcal{D}$ is represented as a collection of words defined as any sequence of letters of length at least three. Stop words are removed. Document vectors are then produced by mapping each word t it contains to its tok2vec[2] representation $\varphi(t)$ (similar to word2vec [83]), and computing the average $\boldsymbol{x} = (1/|\mathcal{D}|)\sum_{t\in\mathcal{D}}\varphi(t)$. We cluster the data using a kernel k-means model with ten support vectors per cluster. Initializing the kernel clustering with ground truth labels and training the kernel k-means model with an EM-style procedure (see Appendix F of the Supplementary Material for details), the cluster assignment converges to a local optimum with the final assignment visualized in Fig. 6 (middle).

We now focus on assessing the quality of the learned clusters. The adjusted rand index (ARI) metric gives a score of 32%, whereas the same model trained with fixed assignments to the true labels reaches 45%. From this score, one could conclude that the algorithm has learned "bad" clusters. Instead, cluster explanations, which expose to the user *what* in a given document is relevant for its membership to a certain cluster, will give a quite different picture. We first note that a direct application of the NEON method we have proposed to obtain such an explanation would result in an explanation in terms of the dimensions of the input vector $\boldsymbol{x}$, which is not interpretable by a human as word and document embeddings are usually abstract. A more interpretable word-level explanation can be achieved, by observing that the mapping from words to document (an averaging of word vectors) and the first layer of the neuralized kernel k-means, are both linear. Thus, they can be combined into a single "big" linear layer that takes as input each word distinctly. These scores can then be pooled over word dimensions [84], leading to a single-relevance score R_t for each individual word t. These explanations can be rendered as highlighted text.

We select a few messages that we show in Fig. 6. The two messages on the left are assigned to the same cluster but were posted to different newsgroups (i.e., have different labels, and thus, hurt the ARI). Here, NEON highlights in both documents the term "version." Closely related terms like "DOS," "windows," and "Ghostscript" are highlighted as well. The fact that "version" was found in both messages and that other related words were present constitutes an explanation

[2]We use spaCy `md` word embeddings: https://spacy.io

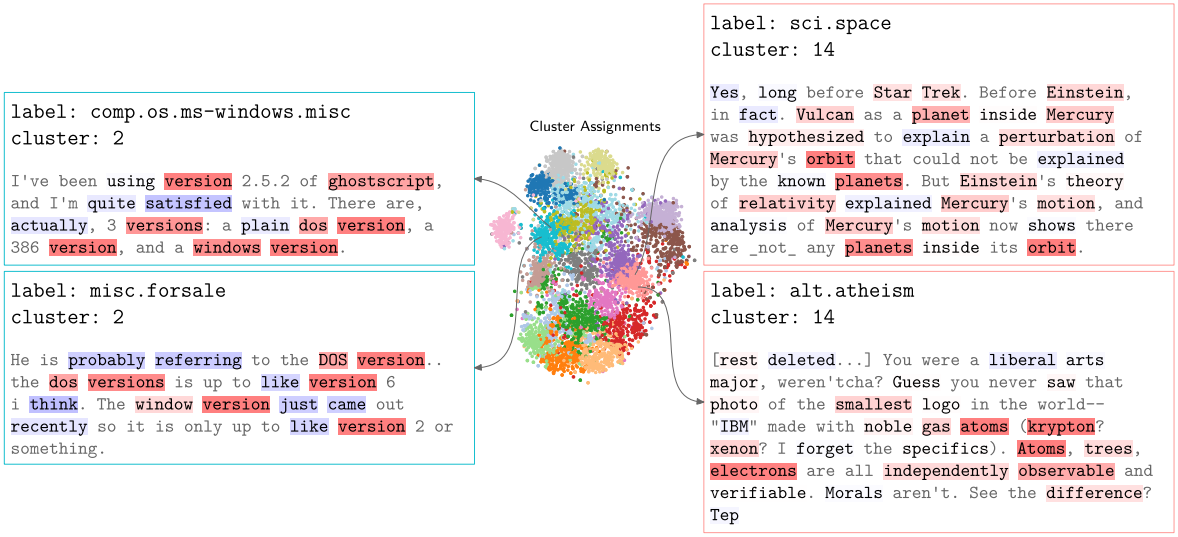

Fig. 6. Application of NEON to the clustering of newsgroup data. Newsgroup texts, where words relevant for cluster membership, are highlighted. Gray words are out of vocabulary.

and justification for these two messages being assigned to the same cluster.

As a second example, consider messages on the right in Fig. 6, posted on two different groups, but that are assigned to the same cluster. The top message is discussing the specifics of Mercury's motion, while the bottom message draws an analogy between physical objects and morals. The most relevant terms are related to physics, such as "Einstein" or "atoms." Also, more broadly used terms (that may appear in other clusters too) like "motion" or "smallest" provide evidence for cluster membership. Here again, the words that have been selected hint at the meaningful similarity between these two messages, thus justifying the assignment of these messages to the same cluster.

Overall, in this showcase experiment, minimizing the clustering objective has led to a rather low ARI. According to common validation procedures, this would constitute a reason for rejection. Instead, the cluster membership explanations produced by NEON could pinpoint to the user meaningful cluster membership decisions that speak in favor of the learned cluster structure.

B. Getting Insights Into Neural Network Representations

Our second showcase example demonstrates how cluster explanations can be applied beyond clusters assessment, in particular, how it can be used as a way of getting insights into some given data representation Ψ, e.g., some layer of a neural network. A direct inspection of the multiple neurons composing the neural network layer is generally unfeasible as there are many such neurons, and their relation to the input is highly nonlinear. The problem of understanding deep representations has received significant attention in recent years [81], [85], [86].

We consider the data representations built by the well-known VGG-16 convolutional network [87]. The VGG-16 network consists of a classifier built on a feature extractor. The feature extractor is composed of five blocks alternating multiple convolutions and ReLU activations. Each block terminates with a 2×2 spatial pooling, thereby creating increasingly more abstract and spatially invariant representations.

To analyze representations produced by VGG-16, we feed some images of interest into the network, leading to spatial activation maps at the output of each block. Collecting the activations at the output of a given block, we build a dataset, where each spatial location in the block corresponds to one data point. After this, we apply k-means with $K = 8$ on these data points (rescaled to the unit norm) and neuralize the model. For each cluster, we consider the model outputs f_c, and propagate these outputs backward through the network using LRP in the neuralized model and further down into the VGG-16 layers to form a collection of pixelwise heatmaps associated with each cluster. When computing the explanations, we set β according to our heuristic in (3), and in convolution layers, we use LRP-γ [58] with $\gamma = 0.25$ in blocks 1–3 and $\gamma = 0.1$ in blocks 4 and 5.

Cluster explanations are shown in Fig. 7 for an artificial spiral image, and one of the well-known "dogs playing poker" images, titled "Poker Game" by Cassius Marcellus Coolidge, 1894. Images were fed to the network at resolution 448×448. In the *artificial spiral image*, clusters at the output of block 3 map to edges with certain angle orientations as well as colors (black and white) or edge types (black-to-white, or white-to-black). Interestingly, strictly vertical and strictly horizontal edges fall in clusters with very high angle specificity, whereas edges with other angles fall into broader clusters. When building clusters at block 4, color and edge information become less prominent. Clusters are now very selective for the angle of the curvature, something needed to represent higher-level concepts. Hence, this analysis reveals to the user a specific property of the VGG-16 neural network which is the progressive building of curvature in deep representations. In the *Poker Game* image, we observe at block 3 a cluster that spans the green texture in the background, one that spans

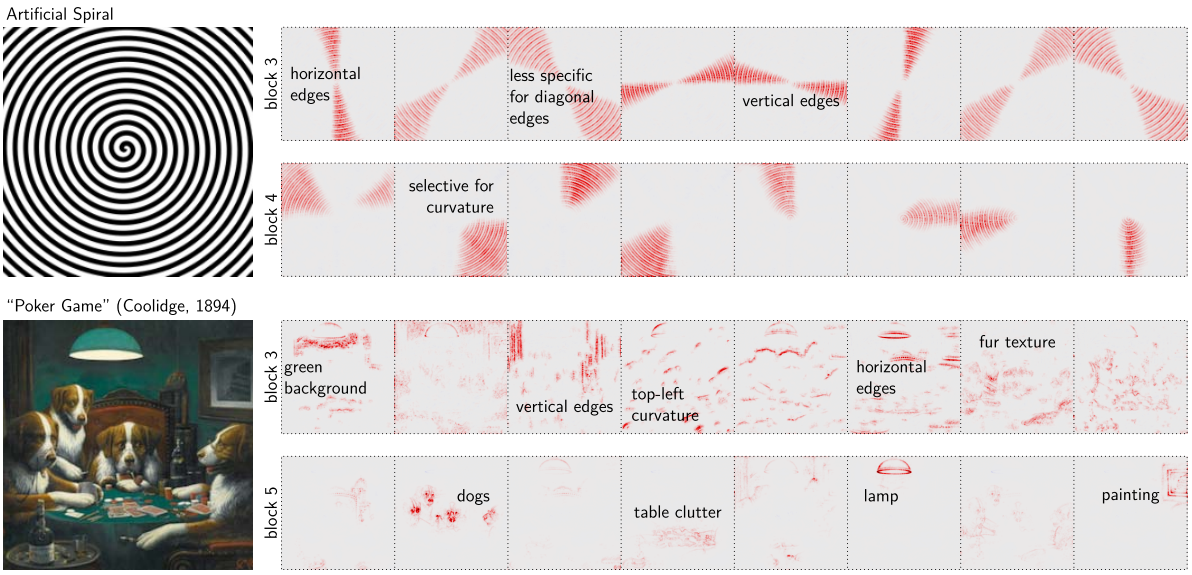

Fig. 7. NEON analysis of images represented at different layers of a deep neural network (pretrained VGG-16). K-means clustering with $K = 8$ is performed at the output of these two blocks. Each column shows the pixelwise contributions for one of these clusters.

the fur texture associated with the dogs, and further clusters that react to edges of various orientations. After block 5, the clusters once again form higher-level concepts. There is a cluster for the big lamp at the top of the image, a cluster for the painting in the upper right, and a cluster that represents the dogs. Note that it only represents the most discriminative part of the dog, and builds invariance w.r.t. other parts of the dog, in particular, the fur texture. This reveals to the user how VGG-16 progressively builds high-level abstractions and becomes invariant to certain visual features.

To summarize, our cluster explanations could extract useful insight into the way VGG-16 represents its input from a small selection of images. In particular, our analysis does away with the high dimensionality of neural network representation by providing an explanation that fits in only eight heatmaps; hence, easily interpretable by the user.

C. Getting Insights Into the Data

While XAI techniques have shown helpful to shed light on the decision strategy associated with specific models and data representations, it also provides a useful tool to extract insight into the data distribution itself (exploratory data analysis). This is often desirable in scientific applications [25], [61], where the model serves to discover interesting correlations in the data, rather than being of interest on its own. Our last showcase demonstrates that NEON, in conjunction with a well-functioning clustering model, can extract such insight into the data. In particular, we find that clusters of the data can be linked to contiguous patterns in pixel space, often corresponding to the image segments provided by the user.

To demonstrate this property of the data, we consider the PASCAL VOC 2007 dataset [88], which comes with segmentation masks separating the different objects. We consider a similar setting to Section VI-B, where we build a collection of K-cluster models based on activation vectors at different spatial locations and at a given layer of the pretrained VGG-16

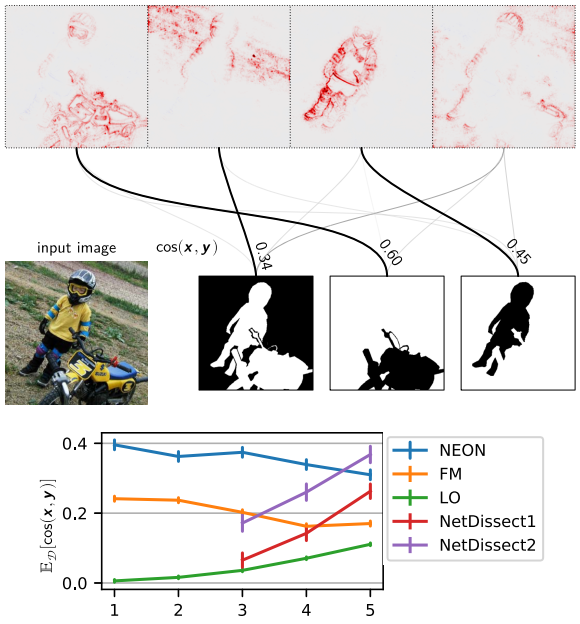

Fig. 8. Quantitative evaluation of NEON's ability to extract meaningful summaries. *Top:* The cluster explanation is matched with ground truth object segmentation masks by means of cosine similarity. *Bottom:* Comparison of NEON to other methods. For each method, we show the average cosine score over the whole dataset. Results are shown for different blocks on the x-axis.

network. The assignment of these activation vectors onto the learned clusters is then attributed to the input pixels using our NEON explanation framework to form a collection of K heat maps. Fig. 8 (top) shows an example of heatmaps we get for an image of a kid with a small motorbike. We observe that the attribution of cluster membership onto pixels highlights that each cluster represents distinct objects in the image, here, the kid, the motorbike, and the background. We perform an experiment, where we measure to what degree explained clusters match the different segmentation masks. The similarity between heatmaps and segmentation masks is measured by

a maximum weight matching (Hungarian algorithm) between masks and clusters, where the weight is given by their cosine similarity. The procedure is depicted in Fig. 8 (middle). The matching is reduced to a single score $S \in [-1, +1]$ by averaging the cosine similarity of all matchings. A perfect score of $S = 1$ can only be achieved if the clusters are strictly equivalent to the matched segmentation masks.

For comparison, we construct two simple baselines that do not make use of clustering: The first baseline takes the top-k most activated (in the L_∞ sense) FMs. The second baseline takes the top-k most activated locations (LO). In addition, we consider a recently proposed method, NetDissect [86], which identifies meaningful segments of an image by thresholding spatial activation maps. Thresholds applied by NetDissect are learned in a supervised manner to match a rich set of concepts (e.g., *wood*, *red*, or *carpet*) from the Broden dataset. The NetDissect1 baseline takes the top-K segmentation maps. NetDissect2 takes K centroids from all segmentation maps. For every method in our benchmark, we fix $K = 4$ (the average number of objects in the dataset) and apply the same LRP propagation rules for NEON, FM, and LO. Examples of heatmaps produced by each method are given in Appendix J in the Supplementary Material.

Average cosine similarities for each method applied at the output of each block[3] are given in Fig. 8 (bottom). The NEON approach clearly and consistently delivers the best results except for block 5, where NetDissect2 shows a better performance. Interestingly, the highest correlation is found in lower layers, confirming that low-level features such as color or textures are good descriptors of the spatial occupancy of an object, whereas higher-level features may build too much invariance to comprehensively highlight segments (see also Section VI-B). The higher performance of NetDissect in a higher layer can be attributed to the smoother way it renders explanation in pixel-space (cf. Appendix J in the Supplementary Material), thereby "undoing" some of the invariances the neural network might have built.

Overall, our NEON approach allows us to shed light on the statistics of complex data distributions, for example, by finding that clusters in image data, especially those coding for low-level information content, such as texture or color, substantially correlate with image segments.

VII. Evaluation

While Section VI has demonstrated the multiple practical benefits one can get from bringing XAI to clustering, we would like to study here more specifically the technical ability of NEON as an explanation method for clustering. We consider a broad spectrum of desiderata of an explanation method and evaluate NEON against a number of simply contributed baselines. We stress that the baselines we use were originally proposed for explaining classification, however, with some adaptations that we propose, they can be extended to the clustering case, and therefore, serve as baselines in our evaluation.

[3]NetDissect only has results for blocks 3–5 due to its high computational cost in the lower layers.

Algorithm 1 AUC Computation for a Data Point $\boldsymbol{z} \in \mathbb{R}^d$ and the Explanation $(R_i)_i \in \mathbb{R}^d$ of Its Prediction

$\mathcal{I} = \varnothing$
`curve = [ ]`
for $\imath \in$ `argsort(`$(-R_i)_i$`)` **do**
  $\mathcal{I} = \mathcal{I} \cup \{\imath\}$
  $\boldsymbol{x} \sim p_{\text{KDE}}(\boldsymbol{x} \mid \boldsymbol{z}_\mathcal{I})$
  `curve.append(`$g_c(\boldsymbol{x})$`)`
end for
return `area_under(curve)` $\cdot 100 \,/\, d$

In particular, we consider IGs [29], where the explanation scores are computed by integrating the model output between the origin and the data point $\boldsymbol{x}$ following some linear path. We then apply PDA [57], [89], where we score the different dimensions based on the effect on the decision function of removing the corresponding feature. The missing feature is either set to zero (PDA_0) or imputed using a KDE conditional sampler (PDA_{cs}), which we describe in Appendix G of the Supplementary Material. Finally, we include four simple baselines: random attribution, squared features $\boldsymbol{x}^2$, sensitivity analysis $(\nabla f)^2$, which computes the square of the derivative along each input dimension, and a method specific to standard k-means, "nearest centroid analysis" (NCA) that computes $(\boldsymbol{x}-\boldsymbol{\mu}_k)^2-(\boldsymbol{x}-\boldsymbol{\mu}_c)^2$, where $\boldsymbol{\mu}_c$ and $\boldsymbol{\mu}_k$ are the centroids of the assigned cluster and nearest competing cluster, respectively, and where the squaring operation applies elementwise.

A. Desiderata and Evaluation Metrics

In the context of explaining image classifiers, [90] proposed the "pixel-flipping" technique for evaluating explanations. The technique consists of constructing a plot that keeps track of the decision function (in our case, this will be the cluster indicator function $g_c(\boldsymbol{x}) = 1_{\{\boldsymbol{x} \to \text{cluster } c\}}$) as we add or remove features by order of relevance according to the explanation, and measuring the area under the curve (AUC). We start from this algorithm and adapt it to our setting. In particular, instead of flipping pixels, we consider general features, and similar to [36] start from an "empty" data point, and add the features from most to least relevant. Missing features are inpainted using a conditional sampler built on the simple kernel density estimation (KDE) model, the details of which we provide in Appendix G in the Supplementary Material, or replaced by zero when the input features are activations of a deep neural network. The procedure for computing the AUC is detailed in Algorithm 1, where the AUC output is a number between 0 and 100. The higher the AUC, the better the explanation. The analysis can be extended to a whole dataset by averaging the AUC obtained for each individual data point and repeating the whole procedure multiple times to reduce the variance produced by the KDE sampling.

Consider now the five desiderata of an explanation listed in [91], namely, *fidelity*, *understandability*, *sufficiency*, *low construction overhead*, and *runtime efficiency*. We argue that Algorithm 1 captures to a reasonable extent the first three of them: *fidelity* (**D1**): Algorithm 1 keeps track of the

model output as we add features. This favors techniques that explain the model output, rather than some other function. *Understandability* (**D2**): It is desirable that the explanation is understandable by its user, e.g., expressible in terms of input features, and simple enough (e.g., a few relevant features). Algorithm 1 implements such desiderata by verifying whether a few most relevant features returned by the explanation produce a substantial increase in the model output. *Sufficiency* (**D3**): The explanation should be sufficient for its user, i.e., provide sufficient information about the model's decision strategy. Algorithm 1 requests a score for each individual feature (or at least a full ranking of those features). This favors explanations with this level of resolution compared with more coarse-grained explanations.

To assess the fulfillment of the last two desiderata, we proceed as follows: *low construction overhead* (**D4**): The explanation technique should not be too complex or costly to implement. Our evaluation will rank explanation methods depending on whether they only need access to the decision function, access to some differentiable function reproducing the decision function, or access to the neural network internals of that function. *Runtime efficiency* (**D5**): The explanation should be computable quickly. In our evaluation, we will provide the algorithmic complexity of each explanation method and perform additional runtime comparisons.

B. AUC Evaluation Results

To test desiderata **D1**–**D3**, we first perform the AUC evaluation presented in Algorithm 1 on a set of models trained on different datasets of various dimensionality and complexity. We consider first a set of standard k-means models trained on a number of datasets from the UCI repository (details and links to the datasets are provided in Appendix H of the Supplementary Material), and where the number of clusters K is determined using the elbow method [92]. Then, we consider more complex kernel k-means models which we train on further datasets from the UCI repository. We also consider the kernel k-means model trained on the 20 newsgroup dataset [82] (`news` in Table I) which we have showcased in Section VI-A. The training algorithm we have used for kernel k-means is detailed in Appendix F in the Supplementary Material. Finally, we consider deep k-means models built on the popular STL-10 [93] image recognition dataset. We consider either a standard k-means model built on the features at the output of block 5 of the VGG-16 deep neural network pretrained on ImageNet (VGG-s), or the same VGG-16 network without supervised pretraining (VGG-u) and coupled with the recently proposed SCAN [23] clustering model[4] for deep clustering. For each dataset and model, we set the NEON hyperparameter according to the heuristic in (3). For deep models, we choose β in the same way and furthermore choose the LRP rule LRP-γ [58], with the parameter γ set heuristically to 0.1. For these two deep clustering models, we consider as a unit of interpretability the 256 FMs at the output of block 3 of the VGG-16 network, and thus, produce explanations in $\mathbb{R}^{256}$. Results are shown in Table I.

[4]We train exactly the same model as in [23], but replace the resnet-18 feature extractor by a VGG-16 feature extractor, which comes with extensively tested LRP rules [58], [63]. Our trained model reaches a clustering accuracy of 72.6 (compared with 76.7 for the original model [23], but well above earlier deep clustering proposals).

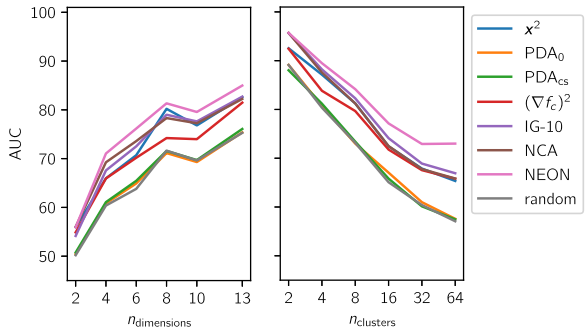

Fig. 9. Effect of the number of retained dimensions d and the number of clusters K on the AUC performance of each explanation method on the `winer` dataset.

We observe that the proposed NEON explanation method is superior to all baselines for the vast majority of considered clustering models and datasets. We note the relatively poor performance of PDA, where the removal of individual features seems insufficient to capture the more global structure of the cluster assignment. To get further insights into the performance of NEON, we perform an experiment where we take an existing dataset, the `winer` dataset, and generate scenarios of varying complexity by training the clustering model between $K = 2$ and $K = 64$, and also removing input features to generate dataset dimensions from $d = 2$ to $d = 13$. The results are shown in Fig. 9.

We observe that in every regime, NEON has equal or superior performance to all baselines. Anecdotally, NEON performs equivalently to NCA for $K = 2$, but it starts to outperform it as soon as the number of clusters grows.

C. Sensitivity of NEON to Hyperparameters

Unlike other baseline methods used in our benchmark, NEON comes with a "stiffness" hyperparameter β, which we have proposed to choose heuristically following (3). For deep clustering, one also needs to choose the parameter γ associated with the propagation in convolution layers. We would like to test the sensitivity of NEON to these parameters, first to verify the soundness of our heuristic, but also to check whether other choices of parameters lead to further improvements or conversely a degradation of NEON performance. Results are given in Fig. 10, where we superpose on the same plot the performance at the heuristically set value for the hyperparameter (orange dot), the performance for other values of the hyperparameter (solid gray line), and the performance of best performing baseline (dotted blue line).

We observe that the simple heuristic proposed in (3) nicely correlates with the peak of AUC performance, thereby providing empirical justification for the proposed heuristic. We note that even if the hyperparameter β is chosen inadequately, AUC performance degrades in most cases only to a minor extent. Conversely, an optimization of the NEON hyperparameters brings slight additional gains to the AUC score. Notably, the seemingly limited performance of NEON on deep clustering

TABLE I

AUC SCORE COMPUTED WITH ALGORITHM 1 AND SERVING AS A PROXY FOR THE FULFILLMENT OF DESIDERATA **D1**–**D3**. THE HIGHER THE AUC SCORE THE BETTER THE EXPLANATIONS. WE FIND THAT THE PROPOSED NEON METHOD SCORES THE HIGHEST FOR THE VAST MAJORITY OF CLUSTERING MODELS. ENTRIES WHERE METHODS ARE INAPPLICABLE OR COMPUTATIONALLY PROHIBITIVE ARE DENOTED BY "—"

dataset			model		methods							
name	N	D	K	type	random	$\boldsymbol{x}^2$	PDA_0	PDA_{cs}	$(\nabla f_c)^2$	IG-10	NCA	NEON
buddy	249	6	7	kmeans	71.77	75.00	70.06	70.81	73.32	74.76	76.41	**78.42**
c2000	2000	68	7	kmeans	90.71	95.12	90.67	90.87	92.01	93.82	92.15	**95.21**
hepac	615	11	8	kmeans	61.16	74.04	59.67	60.03	76.50	77.56	76.20	**80.19**
seeds	210	7	6	kmeans	75.73	78.64	76.07	76.09	81.62	79.54	81.16	**82.81**
winer	178	13	6	kmeans	78.36	85.04	77.01	78.15	82.99	85.87	85.10	**87.23**
news	250	300	20	kernel	40.07	42.83	51.55	—	40.40	40.40	—	**54.50**
trpad	980	10	9	kernel	58.68	71.34	58.87	58.32	68.76	71.44	—	**74.92**
sales	811	52	6	kernel	82.30	**87.28**	82.66	82.33	86.72	86.51	—	87.21
water	527	38	5	kernel	79.30	87.30	79.13	78.89	85.01	86.82	—	**87.66**
whlsl	440	6	8	kernel	54.98	63.91	54.88	55.11	64.53	64.93	—	**67.37**
STL-10	5000	256	10	deep (VGG-s)	50.52	66.66	75.30	—	56.11	66.99	—	**77.93**
STL-10	5000	256	100	deep (VGG-s)	32.42	53.34	48.78	—	39.47	41.85	—	**65.09**
STL-10	5000	256	1000	deep (VGG-s)	27.32	50.36	46.63	—	34.99	38.85	—	**52.38**
STL-10	5000	256	10	deep (VGG-u / SCAN)	58.66	68.54	75.69	—	59.40	70.84	—	**85.76**
STL-10	5000	256	100	deep (VGG-u / SCAN)	38.72	49.02	31.77	—	41.73	25.52	—	**55.38**
STL-10	5000	256	1000	deep (VGG-u / SCAN)	22.98	**32.45**	9.28	—	27.63	6.83	—	23.40

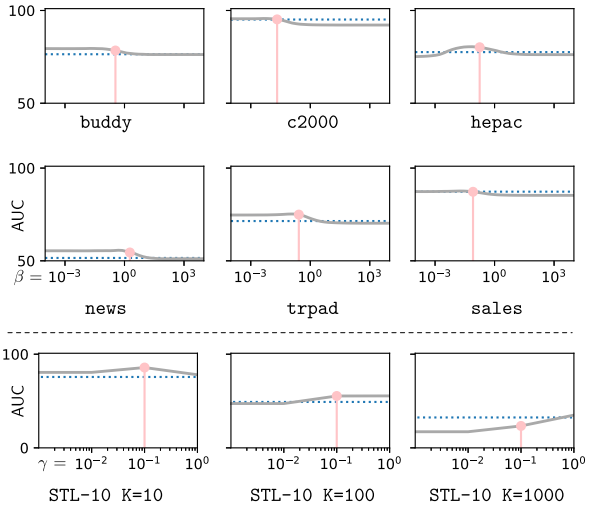

Fig. 10. Evaluating of NEON hyperparameters on a selection of clustering models. First row: k-means models, second row: kernel k-means models, third row: deep models (VGG-u/SCAN). The y-axis shows the pixel flipping AUC. The first two rows show the effect of the MTM parameter β, with the orange marker indicating the proposed heuristic $\beta = \mathbb{E}[f_c]^{-1}$, the dotted line is the best performing baseline (cf. Table I). The last row shows the effect of the LRP convolution parameter γ, with the orange marker indicating our heuristic $\gamma = 0.1$, where we set $\beta = \mathbb{E}[f_c]^{-1}$.

with $K = 1000$ can be overcome by choosing a larger value for the parameter γ, in turn, making NEON again the best performing method. In addition to maximizing the AUC score, the hyperparameters of NEON and the possibility to optimize them can be especially useful when bringing explainability to new tasks with specific performance metrics.

D. Construction Overhead and Runtime

Finally, we would like to study the fulfillment by NEON of desiderata **D4** (low construction overhead) and **D5** (runtime efficiency), compared with other methods in our benchmark. We resort to qualitative analysis for **D4**, where we categorize

TABLE II

FULFILLMENT OF LOW CONSTRUCTION OVERHEAD AND RUNTIME EFFICIENCY DESIDERATA FOR THE METHODS IN OUR BENCHMARK

Method	Overhead (**D4**)	Runtime (**D5**)	
		standard	kernel
$(\boldsymbol{x} - \widetilde{\boldsymbol{x}})^2$	–	$\mathcal{O}(d)$	$\mathcal{O}(d)$
PDA	g_c	$\mathcal{O}(Kd^2)$	$\mathcal{O}(Kd^2p)$
$(\nabla f_c)^2$	∇f_c	$\mathcal{O}(Kd)$	$\mathcal{O}(Kdp)$
IG-10	∇f_c	$\mathcal{O}(10Kd)$	$\mathcal{O}(10Kdp)$
NCA	$(\boldsymbol{\mu}_c)_c$	$\mathcal{O}(Kd)$	—
NEON	NN	$\mathcal{O}(Kd)$	$\mathcal{O}(Kdp^2)$

methods according to what needs to be constructed in additionally to the clustering decision function. Results are shown in Table II (second column). The symbol "–" indicates that we do not even need the decision function, "g_c" indicates that we need the decision function only, "∇f_c" indicates that we need a differentiable surrogate function f_c and its gradient, "$(\boldsymbol{\mu}_c)_c$" indicates that we need the cluster centroids, and finally, "NN" indicates that we need the neural-network equivalent of the surrogate function f_c. The proposed NEON method has the highest overhead in our benchmark as it requires a neural network equivalent. However, since we have already derived these neural network equivalents in the technical sections, there is no significant obstacle to applying NEON to the studied models (k-means, kernel k-means, deep clustering, and related).

Regarding the runtime efficiency (**D5**), we perform a complexity analysis of the different explanation methods, where d is the number of input dimensions, K is the number of clusters, and p is the number of support vectors per cluster in the kernel k-means case. Results are shown in Table II (last column). We observe that for k-means, the NEON computational cost is lower or equal to most explanation methods, by only requiring a single forward and backward pass, whereas several explanation methods need to evaluate the model multiple

times. (An empirical runtime comparison to all baselines for various k-means models can be found in Appendix I of the Supplementary Material). For kernel k-means, results are more balanced, with NEON being slower than simple sensitivity analysis, but running faster than the more advanced PDA and IG competitors if the number of support vectors is smaller than the number of input dimensions or the number of integration steps, respectively. Hence, while for standard k-means, we can generally claim that NEON has high efficiency, for kernel k-means, one needs to additionally ensure that the number of support vectors remains small, typically less than 10.

Overall, we have demonstrated in our evaluation that NEON fares on average the highest, comparing favorably to all competitors when considering the multiple aspects that enter into the assessment of an explanation method. Therefore, NEON constitutes so far the most appropriate and powerful method for tackling the problem of explaining cluster assignments.

VIII. Conclusion

We have contributed by for the first time bringing XAI to clustering and have proposed a general framework, called neuralization-propagation, for explaining cluster assignments of a broad range of clustering models. The proposed method converts, *without retraining*, the clustering model into a *functionally equivalent* neural network composed of detection and pooling layers. This conversion step which we have called "neuralization" enables cluster assignments to be efficiently attributed to input variables by means of a reverse propagation procedure.

The quantitative evaluation shows that our explanation method is capable of identifying cluster-relevant input features in a precise and systematic manner, from the simplest k-means model to some of the most recent proposals, such as the SCAN deep clustering model [23]. The performance remains high across all considered data types, in particular, abstract vector data, text, natural images, or neuron activations.

The method we have proposed complements standard cluster validation techniques by providing rich interpretable feedback on the nature of the clusters that are built. Furthermore, when paired with a well-functioning clustering algorithm, it provides a useful tool for exploratory data analysis and knowledge discovery where complex data distributions are first summarized into finitely many clusters that are then exposed to the human in an interpretable manner.

References

[1] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Comput. Surv.*, vol. 31, no. 3, pp. 264–323, Sep. 1999.

[2] R. Xu and D. Wunsch, II, "Survey of clustering algorithms," *IEEE Trans. Neural Netw.*, vol. 16, no. 3, pp. 645–678, May 2005.

[3] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Upper Saddle River, NJ, USA: Prentice-Hall, 1988.

[4] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. New York, NY, USA: Springer, 2009.

[5] D. Jiang, C. Tang, and A. Zhang, "Cluster analysis for gene expression data: A survey," *IEEE Trans. Knowl. Data Eng.*, vol. 16, no. 11, pp. 1370–1386, Nov. 2004.

[6] H. Celiker and J. Gore, "Clustering in community structure across replicate ecosystems following a long-term bacterial evolution experiment," *Nature Commun.*, vol. 5, no. 1, Aug. 2014.

[7] D. Mekala, V. Gupta, B. Paranjape, and H. Karnick, "SCDV: Sparse composite document vectors using soft clustering over distributional representations," in *Proc. Conf. Empirical Methods Natural Lang. Process.*, 2017, pp. 659–669.

[8] A. K. Jain, "Data clustering: 50 years beyond K-means," *Pattern Recognit. Lett.*, vol. 31, no. 8, pp. 651–666, 2010.

[9] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, Berkeley, CA, USA, vol. 1, 1967, pp. 281–297.

[10] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.

[11] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel K-means: Spectral clustering and normalized cuts," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2004, pp. 551–556.

[12] J. Xie, R. B. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proc. 33rd Int. Conf. Mach. Learn.*, 2016, pp. 478–487.

[13] J. R. Hershey, Z. Chen, J. L. Roux, and S. Watanabe, "Deep clustering: Discriminative embeddings for segmentation and separation," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, Mar. 2016, pp. 31–35.

[14] B. Yang, X. Fu, N. D. Sidiropoulos, and M. Hong, "Towards K-means-friendly spaces: Simultaneous deep learning and clustering," in *Proc. ICML*, vol. 70, 2017, pp. 3861–3870.

[15] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *Proc. 15th Eur. Conf. Comput. Vis.*, 2018, pp. 139–156.

[16] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision* (Lecture Notes in Computer Science), vol. 8689. Cham, Switzerland: Springer, 2014, pp. 818–833.

[17] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PLoS ONE*, vol. 10, no. 7, Jul. 2015, Art. no. e0130140.

[18] M. T. Ribeiro, S. Singh, and C. Guestrin, "'Why should I trust you?': Explaining the predictions of any classifier," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2016, pp. 1135–1144.

[19] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual explanations from deep networks via gradient-based localization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 618–626.

[20] W. Samek, G. Montavon, A. Vedaldi, L. K. Hansen, and K.-R. Müller, *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning* (Lecture Notes in Computer Science), vol. 11700. Cham, Switzerland: Springer, 2019.

[21] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digit. Signal Process.*, vol. 73, pp. 1–15, Feb. 2017.

[22] W. Samek, G. Montavon, S. Lapuschkin, C. J. Anders, and K.-R. Müller, "Explaining deep neural networks and beyond: A review of methods and applications," *Proc. IEEE*, vol. 109, no. 3, pp. 247–278, Mar. 2021.

[23] W. V. Gansbeke, S. Vandenhende, S. Georgoulis, M. Proesmans, and L. V. Gool, "SCAN: Learning to classify images without labels," in *Computer Vision* (Lecture Notes in Computer Science), vol. 12355. Glasgow, U.K.: Springer, 2020, pp. 268–285.

[24] S. Tavazoie, J. D. Hughes, M. J. Campbell, R. J. Cho, and G. M. Church, "Systematic determination of genetic network architecture," *Nature Genet.*, vol. 22, no. 3, pp. 281–285, Jul. 1999.

[25] G. Ciriello, M. L. Miller, B. A. Aksoy, Y. Senbabaoglu, N. Schultz, and C. Sander, "Emerging landscape of oncogenic signatures across human cancers," *Nature Genet.*, vol. 45, no. 10, pp. 1127–1133, Sep. 2013.

[26] A. K. Kau, Y. E. Tang, and S. Ghose, "Typology of online shoppers," *J. Consum. Marketing*, vol. 20, no. 2, pp. 139–156, Apr. 2003.

[27] J. M. Zurada, A. Malinowski, and I. Cloete, "Sensitivity analysis for minimization of input data dimension for feedforward neural network," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 1994, pp. 447–450.

[28] D. Baehrens, T. Schroeter, S. Harmeling, M. Kawanabe, K. Hansen, and K.-R. Müller, "How to explain individual classification decisions," *J. Mach. Learn. Res.*, vol. 11, no. 6, pp. 1803–1831, 2010.

[29] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 3319–3328.

[30] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4768–4777.

[31] G. Montavon, S. Lapuschkin, A. Binder, W. Samek, and K.-R. Müller, "Explaining nonlinear classification decisions with deep Taylor decomposition," *Pattern Recognit.*, vol. 65, pp. 211–222, May 2017.

[32] W. Landecker, M. D. Thomure, L. M. A. Bettencourt, M. Mitchell, G. T. Kenyon, and S. P. Brumby, "Interpreting individual classifications of hierarchical networks," in *Proc. IEEE Symp. Comput. Intell. Data Mining (CIDM)*, Apr. 2013, pp. 32–38.

[33] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 3145–3153.

[34] J. Kauffmann, K.-R. Müller, and G. Montavon, "Towards explaining anomalies: A deep Taylor decomposition of one-class models," *Pattern Recognit.*, vol. 101, May 2020, Art. no. 107198.

[35] L. Arras, G. Montavon, K.-R. Müller, and W. Samek, "Explaining recurrent neural network predictions in sentiment analysis," in *Proc. 8th Workshop Comput. Approaches Subjectivity, Sentiment Social Media Anal.*, 2017, pp. 159–168.

[36] T. Schnake *et al.*, "Higher-order explanations of graph neural networks via relevant walks," *IEEE Trans. Pattern Anal. Mach. Intell.*, early access, Jun. 5, 2021, doi: 10.1109/TPAMI.2021.3115452.

[37] C. Schäfer and J. Laub, "Annealed κ-means clustering and decision trees," in *Classification—The Ubiquitous Challenge*. Berlin, Germany: Springer, 2005, pp. 682–689.

[38] D. Bertsimas, A. Orfanoudaki, and H. M. Wiberg, "Interpretable clustering via optimal trees," 2018, *arXiv: 1812.00539*.

[39] R. Fraiman, B. Ghattas, and M. Svarc, "Interpretable clustering using unsupervised binary trees," *Adv. Data Anal. Classification*, vol. 7, no. 2, pp. 125–145, Jun. 2013.

[40] M. Moshkovitz, S. Dasgupta, C. Rashtchian, and N. Frost, "Explainable K-means and K-medians clustering," in *Proc. 37th Int. Conf. Mach. Learn. (ICML)*, vol. 119, Jul. 2020, pp. 7055–7065.

[41] P. Geurts, N. Touleimat, M. Dutreix, and F. d'Alché-Buc, "Inferring biological networks with output kernel trees," *BMC Bioinf.*, vol. 8, no. S2, pp. 1–12, May 2007.

[42] K. G. Dizaji, A. Herandi, C. Deng, W. Cai, and H. Huang, "Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2017, pp. 5747–5756.

[43] X. Guo, X. Liu, E. Zhu, and J. Yin, "Deep clustering with convolutional autoencoders," in *Neural Information Processing* (Lecture Notes in Computer Science), vol. 10635. Cham, Switzerland: Springer, 2017, pp. 373–382.

[44] X. Peng, Y. Li, I. W. Tsang, H. Zhu, J. Lv, and J. T. Zhou, "XAI beyond classification: Interpretable neural clustering," *J. Mach. Learn. Res.*, vol. 23, pp. 6:1–6:28, Jan. 2022.

[45] J. G. Dy and C. E. Brodley, "Feature selection for unsupervised learning," *J. Mach. Learn. Res.*, vol. 5, pp. 845–889, Jan. 2004.

[46] M. H. C. Law, M. A. T. Figueiredo, and A. K. Jain, "Simultaneous feature selection and clustering using mixture models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 1154–1166, Sep. 2004.

[47] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "On clustering validation techniques," *J. Intell. Inf. Syst.*, vol. 17, no. 2, pp. 107–145, Dec. 2001.

[48] T. Lange, V. Roth, M. L. Braun, and J. M. Buhmann, "Stability-based validation of clustering solutions," *Neural Comput.*, vol. 16, no. 6, pp. 1299–1323, 2004.

[49] M. Meila, "How to tell when a clustering is (approximately) correct using convex relaxations," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 7418–7429.

[50] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge, U.K.: Cambridge Univ. Press, 2008.

[51] T. Metsalu and J. Vilo, "ClustVis: A web tool for visualizing clustering of multivariate data using principal component analysis and heatmap," *Nucleic Acids Res.*, vol. 43, no. W1, pp. W566–W570, May 2015.

[52] M. Kern, A. Lex, N. Gehlenborg, and C. R. Johnson, "Interactive visual exploration and refinement of cluster assignments," *BMC Bioinf.*, vol. 18, no. 1, pp. 1–13, Sep. 2017.

[53] K. Hansen, D. Baehrens, T. Schroeter, M. Rupp, and K.-R. Müller, "Visual interpretation of kernel-based prediction models," *Mol. Informat.*, vol. 30, no. 9, pp. 817–826, Sep. 2011.

[54] P. D'haeseleer, "How does gene expression clustering work?" *Nature Biotechnol.*, vol. 23, no. 12, pp. 1499–1501, Dec. 2005.

[55] D. Sculley, "Web-scale K-means clustering," in *Proc. 19th Int. Conf. World Wide Web (WWW)*, 2010, pp. 1177–1178.

[56] J. O. Hanson *et al.*, "Global conservation of species' niches," *Nature*, vol. 580, no. 7802, pp. 232–234, Mar. 2020.

[57] L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling, "Visualizing deep neural network decisions: Prediction difference analysis," in *Proc. ICLR*, 2017, pp. 1–12.

[58] G. Montavon, A. Binder, S. Lapuschkin, W. Samek, and K.-R. Müller, "Layer-wise relevance propagation: An overview," in *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning* (Lecture Notes in Computer Science), vol. 11700. Cham, Switzerland: Springer, 2019, pp. 193–209.

[59] S. Lapuschkin, A. Binder, K.-R. Müller, and W. Samek, "Understanding and comparing deep neural networks for age and gender classification," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCVW)*, Oct. 2017, pp. 1629–1638.

[60] Y. Ding, Y. Liu, H. Luan, and M. Sun, "Visualizing and understanding neural machine translation," in *Proc. 55th Annu. Meeting Assoc. Comput. Linguistics*, 2017, pp. 1150–1159.

[61] F. Horst, S. Lapuschkin, W. Samek, K.-R. Müller, and W. I. Schöllhorn, "Explaining the unique nature of individual gait patterns with deep learning," *Sci. Rep.*, vol. 9, no. 1, p. 2391, Feb. 2019.

[62] L. Perotin, R. Serizel, E. Vincent, and A. Guerin, "CRNN-based multiple DoA estimation using acoustic intensity features for ambisonics recordings," *IEEE J. Sel. Topics Signal Process.*, vol. 13, no. 1, pp. 22–33, Mar. 2019.

[63] O. Eberle, J. Buttner, F. Krautli, K.-R. Müller, M. Valleriani, and G. Montavon, "Building and interpreting deep similarity models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 3, pp. 1149–1161, Mar. 2022.

[64] C. J. Anders, L. Weber, D. Neumann, W. Samek, K.-R. Müller, and S. Lapuschkin, "Finding and removing Clever Hans: Using explanation methods to debug and improve deep models," *Inf. Fusion*, vol. 77, pp. 261–295, Jan. 2022.

[65] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *J. Physiol.*, vol. 160, no. 1, pp. 106–154, Jan. 1962.

[66] J. Von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," *Automata Stud.*, vol. 34, pp. 43–98, Dec. 1956.

[67] G. Montavon, "Gradient-based vs. propagation-based explanations: An axiomatic comparison," in *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning* (Lecture Notes in Computer Science), vol. 11700. Cham, Switzerland: Springer, 2019, pp. 253–265.

[68] L. S. Shapley, "A value for n-person games," in *Contributions to Theory Games*, vol. 2. Princeton, NJ, USA: Princeton Univ. Press, 1953.

[69] E. Strumbelj and I. Kononenko, "An efficient explanation of individual classifications using game theory," *J. Mach. Learn. Res.*, vol. 11, pp. 1–18, Jan. 2010.

[70] B. Schölkopf *et al.*, "Input space versus feature space in kernel-based methods," *IEEE Trans. Neural Netw.*, vol. 10, no. 5, pp. 1000–1017, Sep. 1999.

[71] K.-R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, "An introduction to kernel-based learning algorithms," *IEEE Trans. Neural Netw.*, vol. 12, no. 2, pp. 181–201, Jan. 2001.

[72] B. Schölkopf and A. J. Smola, *Learning With Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2002.

[73] R. Zhang and A. I. Rudnicky, "A large scale clustering scheme for kernel K-means," in *Proc. 16th Int. Conf. Pattern Recognit.*, 2002, pp. 289–292.

[74] Y. Li, P. Hu, J. Z. Liu, D. Peng, J. T. Zhou, and X. Peng, "Contrastive clustering," in *Proc. AAAI Conf. Artif. Intell.*, 2021, pp. 8547–8555.

[75] X. Peng, J. Feng, S. Xiao, W.-Y. Yau, J. T. Zhou, and S. Yang, "Structured autoencoders for subspace clustering," *IEEE Trans. Image Process.*, vol. 27, no. 10, pp. 5076–5086, Oct. 2018.

[76] L. Arras *et al.*, "Explaining and interpreting LSTMs," in *Explainable AI: Interpreting, Explaining and Visualizing Deep Learning* (Lecture Notes in Computer Science), vol. 11700. Cham, Switzerland: Springer, 2019, pp. 211–238.

[77] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. 2nd Int. Conf. Knowl. Discovery Data Mining*, 1996, pp. 226–231.

[78] J. C. Gower and G. J. S. Ross, "Minimum spanning trees and single linkage cluster analysis," *Appl. Statist.*, vol. 18, no. 1, p. 54, 1969.

[79] M. Meila and J. Shi, "Learning segmentation by random walks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2000, pp. 873–879.

[80] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Proc. Adv. Neural Inf. Process. Syst.*, 2001, pp. 849–856.

[81] S. Lapuschkin, S. Wäldchen, A. Binder, G. Montavon, W. Samek, and K.-R. Müller, "Unmasking Clever Hans predictors and assessing what machines really learn," *Nature Commun.*, vol. 10, no. 1, p. 1096, 2019.
[82] T. Joachims, "A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization," in *Proc. 14th Int. Conf. Mach. Learn.*, 1997, pp. 143–151.
[83] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proc. 1st Int. Conf. Learn. Represent.*, 2013, pp. 1–12.
[84] L. Arras, F. Horn, G. Montavon, K.-R. Müller, and W. Samek, "'What is relevant in a text document?': An interpretable machine learning approach," *PLOS ONE*, vol. 12, no. 8, Aug. 2017, Art. no. e0181142.
[85] A. Nguyen, A. Dosovitskiy, J. Yosinski, T. Brox, and J. Clune, "Synthesizing the preferred inputs for neurons in neural networks via deep generator networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3387–3395.
[86] B. Zhou, D. Bau, A. Oliva, and A. Torralba, "Interpreting deep visual representations via network dissection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 9, pp. 2131–2145, Sep. 2018.
[87] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. 3rd Int. Conf. Learn. Represent.*, 2015, pp. 1–14.
[88] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. (Jun. 28, 2022). *The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results*. [Online]. Available: http://host.robots.ox.ac.uk/pascal/VOC/voc2007/
[89] I. Covert, S. Lundberg, and S. Lee, "Explaining by removing: A unified framework for model explanation," 2020, *arXiv: 2011.14878*.
[90] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller, "Evaluating the visualization of what a deep neural network has learned," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 11, pp. 2660–2673, Nov. 2017.
[91] W. R. Swartout and J. D. Moore, *Explanation in Second Generation Expert Systems*. Berlin, Germany: Springer-Verlag, 1993, pp. 543–585.
[92] V. Satopaa, J. Albrecht, D. Irwin, and B. Raghavan, "Finding a 'kneedle' in a haystack: Detecting knee points in system behavior," in *Proc. 31st Int. Conf. Distrib. Comput. Syst. Workshops*, Jun. 2011, pp. 166–171.
[93] A. Coates, A. Y. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *Proc. AISTATS*, vol. 15, 2011, pp. 215–223.

Jacob Kauffmann received the bachelor's and master's degrees in computer science from Berlin Institute of Technology, Berlin, in 2014 and 2017, respectively, where he is currently pursuing the Ph.D. degree with the Machine Learning Group.

Malte Esders received the bachelor's degree in cognitive and neurobiological psychology from Utrecht University, Utrecht, The Netherlands, in 2014, and the master's degree in computational neuroscience from the Berlin Institute of Technology, Berlin, Germany, in 2017, where he is currently pursuing the Ph.D. degree with the Machine Learning Group.

Lukas Ruff received the bachelor's degree in mathematical finance from the University of Konstanz, Konstanz, Germany, in 2015, and the joint master's degree in statistics from the Humboldt University of Berlin, Berlin, Germany, the Berlin Institute of Technology (TU Berlin), Berlin, and Freie Universität Berlin, Berlin, in 2017, and the Ph.D. degree with a focus on anomaly detection from the Machine Learning Group, TU Berlin, in 2021.

He is currently with AIGNOSTICS, Berlin, a startup applying machine learning methods for digital pathology.

Grégoire Montavon received the master's degree in communication systems from the École Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, in 2009, and the Ph.D. degree in machine learning from the Technische Universität Berlin, Berlin, Germany, in 2013.

He is currently a Senior Researcher with the Machine Learning Group, Technische Universität Berlin, and the Berlin Institute for the Foundations of Learning and Data. He is also a member of the European Laboratory for Learning and Intelligent Systems (ELLIS) Unit Berlin, Berlin. His research interests include explainable machine learning, deep neural networks, and unsupervised learning.

Dr. Montavon is an Editorial Board Member of *Pattern Recognition*. He was a recipient of the 2020 Pattern Recognition Best Paper Award.

Wojciech Samek (Member, IEEE) received the master's degree in computer science from the Humboldt University of Berlin, Berlin, Germany, in 2010, and the Ph.D. degree (Hons.) in machine learning from the Technical University of Berlin, Berlin, in 2014.

He is currently a Professor of computer science with the Technical University of Berlin and jointly heading the Department of Artificial Intelligence, Fraunhofer Heinrich Hertz Institute, Berlin. He is also an Associate Faculty Member with the Berlin Institute for the Foundation of Learning and Data, Berlin, the European Laboratory for Learning and Intelligent Systems (ELLIS) Unit Berlin, Berlin, and the Deutsche Forschungsgemeinschaft (DFG) Graduate School, BIOQIC, Berlin. He received scholarships from the German Academic Scholarship Foundation and the DFG Research Training Group Graduiertenkollegs (GRK) 1589/1. His research interests include deep learning, explainable AI, neural network compression, and federated learning.

Dr. Samek is a Senior Editor of the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, an Editorial Board Member of *Pattern Recognition*, and an Elected Member of the IEEE Machine Learning for Signal Processing (MLSP) Technical Committee. He has been serving as an AC for NAACL'21. He was a recipient of multiple Best Paper Awards, including the 2020 Pattern Recognition Best Paper Award.

Klaus-Robert Müller (Member, IEEE) received the Diploma degree in physics and the Ph.D. degree in computer science from the Technische Universität Karlsruhe, Karlsruhe, Germany, in 1989 and 1992, respectively.

He held a post-doctoral position with GMD FIRST (later Fraunhofer FIRST), Berlin. He was a Research Fellow with The University of Tokyo, Tokyo, Japan, from 1994 to 1995. In 1995, he founded the Intelligent Data Analysis Group, GMD FIRST, where he was the Director until 2008. From 1999 to 2006, he was a Professor with the University of Potsdam, Potsdam, Germany. He has been a Professor of computer science with the Technische Universität Berlin, Berlin, Germany, since 2006, where he is also the Co-Director of the Berlin Big Data Center. His research interests include intelligent data analysis and machine learning with applications in neuroscience, with an emphasis on brain–computer interfaces, medicine, with an emphasis on digital pathology, physics, chemistry, and humanities.

Dr. Müller was a recipient of the Olympus Prize for Pattern Recognition in 1999, the SEL Alcatel Communication Award in 2006, the Science Prize of Berlin by the Governing Mayor of Berlin in 2014, and the Vodafone Innovations Award in 2017. In 2012, he was an Elected Member of the German National Academy of Sciences-Leopoldina in 2017, the Berlin Brandenburg Academy of Sciences in 2021, the German National Academy of Engineering-Acatech in 2021, and an External Scientific Member of the Max Planck Society in 2017. Since 2019, he has been an ISI Highly Cited Researcher.