From Clustering to Cluster Explanations via Neural Networks

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

***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, 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*—unsupervised learning, k-means clustering, neural networks, 'neuralization', explainable machine learning**

I. Introduction

Clustering is a successful unsupervised learning model that reflects the intrinsic heterogeneities of common data generation processes [1], [2], [3], [4]. Natural cluster structures are observed in a variety of contexts from e.g. 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], [13], [14], [15].

J. Kauffmann is with the Berlin Institute of Technology (TU Berlin), 10587 Berlin, Germany.

M. Esders is with the Berlin Institute of Technology (TU Berlin), 10587 Berlin, Germany.

L. Ruff is with Aignostics, 10117 Berlin, Germany.

G. Montavon is with the Berlin Institute of Technology (TU Berlin), 10587 Berlin, Germany; and BIFOLD – Berlin Institute for the Foundations of Learning and Data, 10587 Berlin, Germany. E-mail: gregoire.montavon@tu-berlin.de.

W. Samek is with Fraunhofer Heinrich Hertz Institute, 10587 Berlin, Germany; and BIFOLD – Berlin Institute for the Foundations of Learning and Data, 10587 Berlin, Germany. E-mail: wojciech.samek@hhi.fraunhofer.de

K.-R. Müller is with the Berlin Institute of Technology (TU Berlin), 10587 Berlin, Germany; BIFOLD – Berlin Institute for the Foundations of Learning and Data, 10587 Berlin, Germany; the Department of Artificial Intelligence, Korea University, Seoul 136-713, Korea; and the Max Planck Institut für Informatik, 66123 Saarbrücken, Germany. E-mail: klaus-robert.mueller@tu-berlin.de.

(Corresponding Authors marked with asterisk: Grégoire Montavon, Klaus-Robert Müller)

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], [17], [18], [19], [20].

In this paper, 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 as neural networks, which then serve as a backbone to guide the explanation process. Technically, we suggest to apply 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.

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 to a potentially simpler one-step approach without neuralization. Our contributions can be summarized as follows:

- Introduction of XAI to clustering, specifically, explanation of the assignment of individual data points onto clusters, in terms of input features.
- 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.

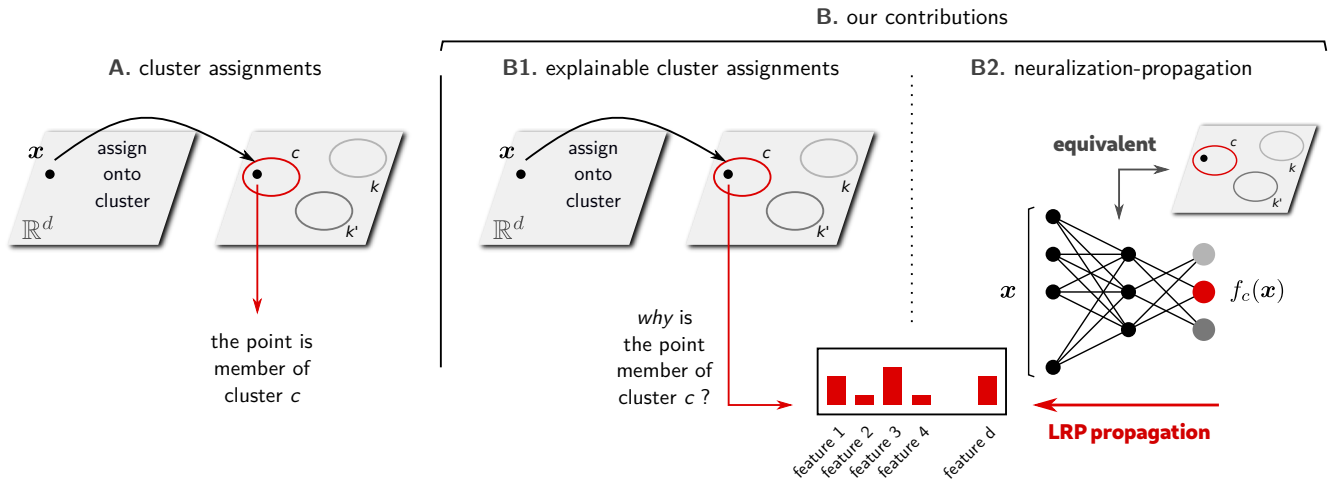

Fig. 1. From clustering to cluster explanations via neural networks. **A.** Standard clustering scenario where data is assigned onto clusters according to the clustering model. **B.** Overview of our contributions. **B1.** We enrich the cluster assignment with an explanation highlighting what features 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).

- 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.
- 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 into 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], [28], [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], [32], [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 SVMs [34] or LSTM networks [35], the method we propose here contributes by offering a solution to the so far unsolved problem of explaining cluster assignments.

Note that the few cluster interpretability techniques have so far been based on surrogate decision trees [36], [37], [38], [39], [40], 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 Explainable AI techniques specific to decision trees. With such a surrogate approach, the user typically has to trade off faithfulness to the original model against explainability.

Related to the connections we establish in this paper 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], [41], [42]. Other models, in particular, the k-meansNet [43] 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 algorithm explainable.

Another set of related works focus 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 [44]. Other approaches simultaneously solve feature selection and clustering by defining a joint objective function to be minimized [45]. 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 [46], cluster stability under resampling / perturbations [47], [48], or purity, i.e. the absence of examples with different labels in the same cluster [49].

Our work enhances the validation of clustering models by producing human-interpretable feedback, a critical step to identify whether cluster assignments are supported by meaningful features or by what the user would consider to be artifacts.

Lastly, user interfaces have been developed to better navigate cluster structures, as they occur, e.g. in biology applications [50], [51]. 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 [52]. 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. [53], [54], [55]). This section presents our neuralization approach for explaining a k-means cluster assignment on the 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. At every step of k-means, points are assigned 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 Explainable AI techniques such as prediction difference analysis [16], [56] or LIME [18], as they apply out-of-the-box to *any* model or decision function. However, these approaches require to evaluate 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, local perturbation may not faithfully depict the overall contribution of a feature to the clustering decision, especially if multiple features needs to be perturbed in order to affect the decision.

In the context of supervised learning, more efficient Explainable AI 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 [29], or Layer-wise Relevance Propagation (LRP) [17], [22], [57]. 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], [58], [59], [60], [61], [62], [63], [64]), and can be embedded in the framework of deep Taylor decomposition [31].

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 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 Eq.* (1) *can be reproduced by a two-layer neural network composed of a standard linear layer and a (min-)pooling layer:*

Neuralized k-means

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

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* $f_c(\boldsymbol{x}) > 0$.

(cf. Appendix A of the Supplement for a 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 two-dimensional 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 to the given k-means model (left).

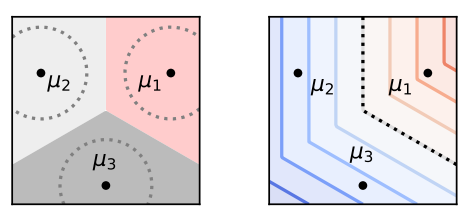

Fig. 2. Left: Decision function of a k-means clustering model with centroids $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \boldsymbol{\mu}_3$. Data points in the region highlighted in red are assigned to the first cluster. Right: Contour plot of the function $f_c(\boldsymbol{x})$ for the cluster $c = 1$.

The neural network above 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 [43]. 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 Supplement. 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 integrated gradients [29], or layer-wise relevance propagation (LRP) [17], [67]. The LRP technique, in particular, leverages the neural network structure to produce 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 are purposely designed for the task of explanation, and ensure certain desirable properties of an explanation such as conservation of predicted evidence and local consistency of explanations [17], [67].

Let us start with the output of the neural network f_c, which we wish to attribute to neurons in the intermediate layer $(h_k)_k$, by propagating through the min function. 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 β is a stiffness hyperparameter. The stiffness parameter interpolates between a uniform redistribution strategy ($\beta \to 0$) and a min-take-all strategy ($\beta \to \infty$). Note that compared to these two extreme cases, our approach allows to contextualize the explanation (i.e. not redistributing on clusters 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 it according to the simple heuristic:

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

where the expectation is computed over the whole dataset. In order 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 the 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 mid-point 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 mid-point, $\boldsymbol{x} - \boldsymbol{m}_k$, 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 Eqs. (2) and (4) by showing that their application produce, 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 a theoretically grounded solution to the problem of attributing the value of a coalition of players to individual players in the coalition. The attribution is based on the effect of removing certain players from the coalition, and can be shown to be the only solution that satisfies four basic axioms, namely *efficiency*, *symmetry*, *linearity*, and *'null player'*. 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 $\widetilde{x}_i$. The exact Shapley Value formula is provided in Appendix B in the Supplement.

Proposition 2. *Redistribution of $f_c(\boldsymbol{h})$ on cluster competitors, performed by Eq.* (2) *and parameter $\beta = 0$, corresponds to the Shapley Value obtained with the reference point $\widetilde{\boldsymbol{h}} = \boldsymbol{0}$.*

(The proof is given in Appendix B of the Supplement.) 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 Eqs.* (2) *and* (4) *corresponds to the Shapley Value obtained with the reference point $\widetilde{\boldsymbol{x}} = \boldsymbol{m}_k$.*

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

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:

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

where the centroids are also defined in feature space.

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 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 feature map), 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 Eq. (5) the decision function

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

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 non-overlapping sets of indices of support vectors that represent these clusters, and where $\mathrm{LME}^{-\gamma}$ denotes a generalized F-mean with $F(t) = e^{-\gamma t}$, i.e.

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

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 Eqs. (5) and (6) are based, are illustrated for some toy one-dimensional cluster c composed of 6 data points in Fig. 3.

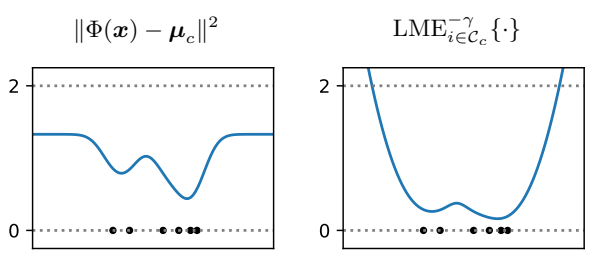

Fig. 3. Distance between some data point $\boldsymbol{x}$ and a cluster c depicted as a collection of black dots. The distance is either computed in feature space, or using the soft min-pooling of Eq. (6).

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* $\boldsymbol{\mu}_c = \frac{1}{Z_c}\sum_{i\in\mathcal{C}_c}\Phi(\boldsymbol{u}_i)$ *where* Φ *is some feature map associated to 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 Eqs.* (5) *and* (6) *can be related as:*

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

where g_c *is a monotonically increasing function defined as:*

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

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 Supplement. Formally, equivalence between the two decision functions (Eqs. (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 have 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 exactly reproduced by a neural network.

Proposition 5. *The decision function in Eq.* (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 the Appendix D of the Supplement. An example showing the equivalence between the neural network output and Eq. (6) is given in Fig. 4.

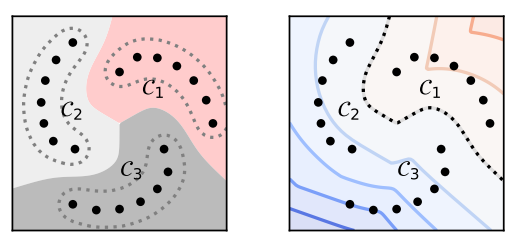

Fig. 4. Left: Partition implemented by a kernel k-means clustering with three clusters $\mathcal{C}_1, \mathcal{C}_2, \mathcal{C}_3$ supported by seven support vectors each. Right: Neural network output $f_c(\boldsymbol{x})$ associated to the cluster $\mathcal{C}_1$.

This neural network we have proposed 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 Eq. (2)

(and switching the sign for the soft max-pooling case)[1]. The directional redistribution in the first layer can be achieved using Eq. (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 special case can be handled by only propagating part of the relevance (and dissipating the rest), specifically, by performing the reassignment:

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

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. (A complexity analysis of the different explanation methods is given in Table II of Section VII.) For NEON to maintain its computational advantage compared to approaches based on function evaluation such as integrated gradients [29] or prediction difference analysis [56], the number of support vectors must be kept small, typically, in the order of 10 support vectors per cluster. Practical approaches to produce a limited number of points include e.g. reduced sets [70], [71], [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. the few neurons that significantly affect the output for a given input $\boldsymbol{x}$.

IV. Extension to Deep Clustering

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

Various formulations of deep k-means have been proposed in the literature. 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 Φ with the neuralized k-means model defined in Proposition 1:

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

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

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 $b_k = \|\boldsymbol{\mu}_k\|^2 - \|\boldsymbol{\mu}_c\|^2$. Note that beyond a simple application of standard k-means on top of a given layer, there has been many proposals for deep clustering.

Two quite popular formulations make use of a soft cluster assignment model, specifically, a softargmax model [23], [41], or a t-Student similarity model [12], [42]. These soft clustering approaches bring a probabilistic interpretation of cluster assignments, and build entropy-based optimization criteria. In the soft k-means models of [23], [41], the data is first projected on some direction $\boldsymbol{\mu}_c$ associated to 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.

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

Proposition 6. *The decision function of Eq.* (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 Supplement. The solution in [12], [42] is 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 above, however, it converges to hard k-means when the activations $\boldsymbol{a}$ of the different clusters become strongly separated.

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 Eq. (12), and where θ is some value between 0 and 1.

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

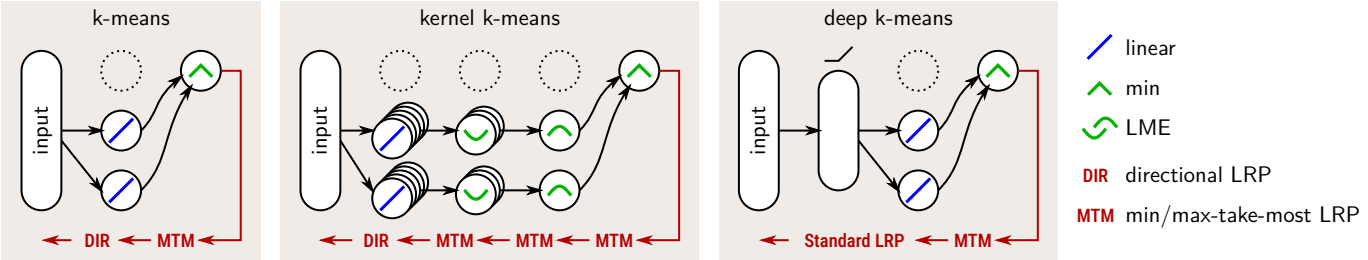

Fig. 5. Examples of clustering models whose cluster assignments can be explained with our NEON approach. The neuralized models, each of which can be expressed as combinations of detection layers and pooling layers, are depicted along with the propagation rules applied at each layer.

Neuralized deep soft clustering (absolute)

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

where $\boldsymbol{w}_k = \boldsymbol{\mu}_c - \boldsymbol{\mu}_k$, $b_k = -\log(N-1) + \log((1-\theta)/\theta)$ *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. it a log-likelihood ratio plus an offset.*

A proof is given in Appendix E of the Supplement. Like for kernel the k-means case, min-take-most can be applied to the min layers. For the last neuralized variant featuring the LME computation, one also needs to handle the case where non-zero 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 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 network are applicable. For propagation rules specific to deep neural networks, we refer to the papers [57], [74] 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 [75], hierarchical agglomerative clustering [76], or spectral clustering [77], [78], 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 cluster algorithm but that have instead been curated by humans to match their expert knowledge.

Compared to 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 explanation (cf. [79]), 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 Explainable AI 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 of 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 20newsgroups dataset [80] 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 representation $\varphi(t)$[2] (similar to word2vec [81]), and computing the average $\boldsymbol{x} = \frac{1}{|\mathcal{D}|}\sum_{t \in \mathcal{D}} \varphi(t)$. We cluster the data using a kernel k-means model with 10 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 Supplement 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

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

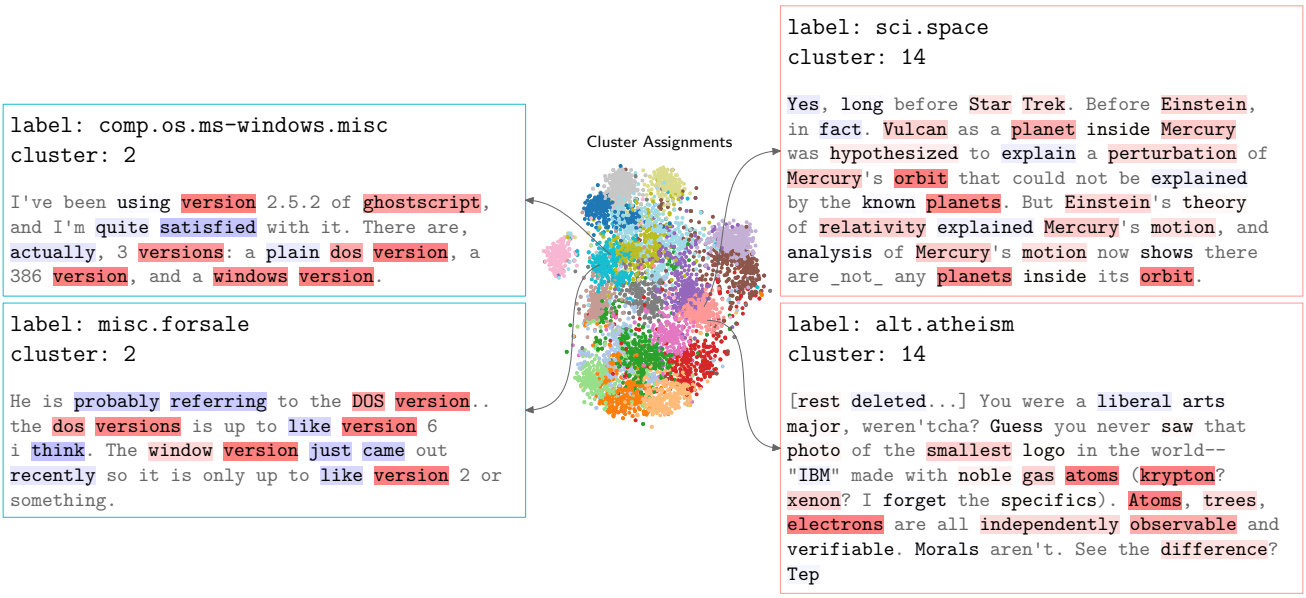

Fig. 6. Application of NEON to the clustering of newsgroup data. Newsgroup texts where words relevant for cluster membership are highlighted. Gray words are out of vocabulary.

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 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 [82], 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. 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 and justification for these two messages being assigned to the same cluster.

As a second example, consider messages on the right in Figure 6, posted on two different groups, but that are assigned to the same cluster. The top message is discussing specifics of Mercury's motion, whilst 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 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 Φ, e.g. some layer of a neural network. An 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 [79], [83], [84].

We consider the data representations built by the well-known VGG-16 convolutional network [85]. 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 image 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

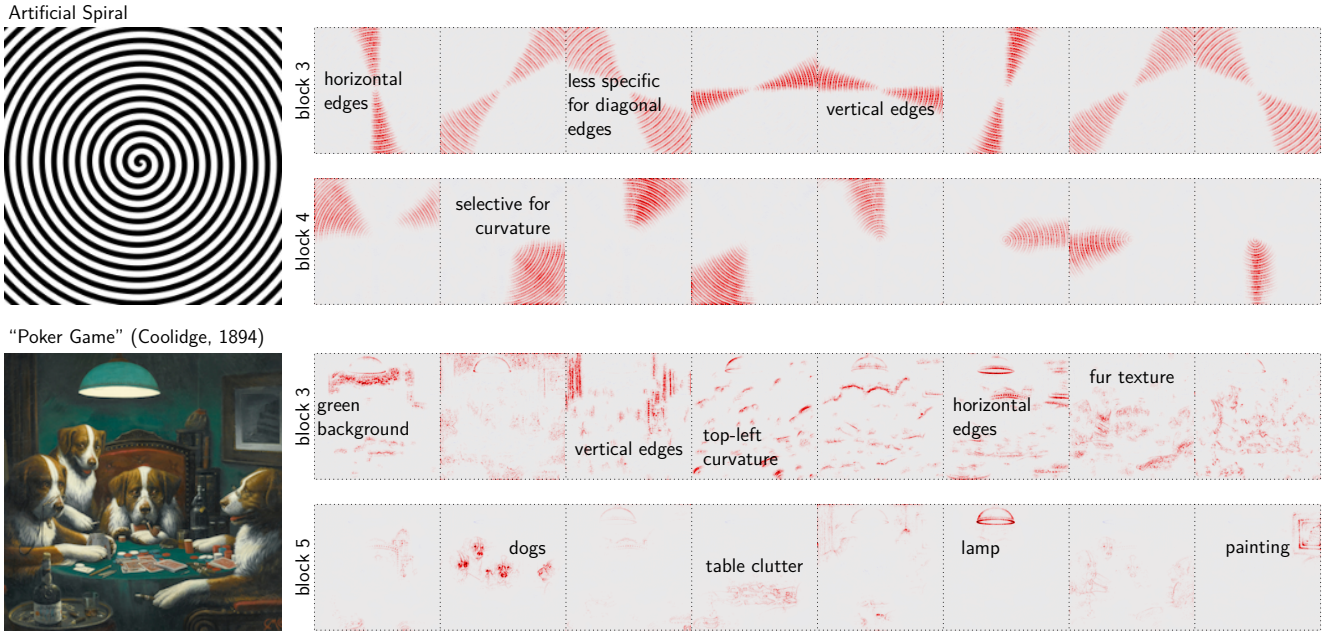

Fig. 7. NEON analysis of images represented at different layers of a deep neural network (pretrained VGG-16). K-means clustering with $K = 8$ is performed at the output of these two blocks. Each column shows the pixel-wise contributions for one of these clusters.

data point. After this, we apply k-means with $K = 8$ on these data points (rescaled to unit norm) and neuralize the model. For each cluster, we consider the model outputs f_c (the positive part), 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 pixel-wise heatmaps associated to each cluster. When computing the explanations, we set β according to our heuristic in Eq. (3), and in convolution layers, we use LRP-γ [57] with $\gamma = 0.25$ in blocks 1–3 and $\gamma = 0.1$ in blocks 4–5.

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, which can be interpreted as applying VGG-16 to the multiple 224×224 patches of that image. 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 the fur texture associated to 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 build invariance w.r.t. other parts of the dogs, in particular, the fur texture. This reveals to the user how VGG-16 progressively builds high-level abstractions and become invariant to certain visual features.

To summarize, our cluster explanations could extract useful insight about 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 8 heatmaps, hence easily interpretable by the user.

C. Getting Insights into the Data

While Explainable AI techniques have shown helpful to shed light into the decision strategy associated to 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 the case in scientific applications [25], [60], where the model serves to extract structures 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 [86] 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 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 heatmaps. 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 highlight that cluster models distinct objects in the image, here, the kid, the motorbike and the background. We perform an experiment where measure to what degree explained clusters match the different segmentation masks. 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.

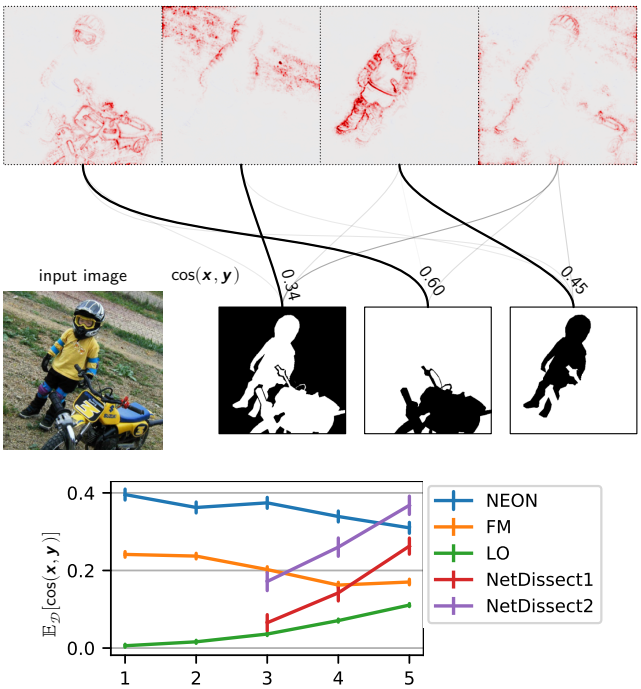

Fig. 8. Quantitative evaluation of NEON's ability to extract meaningful summaries. *Top:* The cluster explanation is matched with ground truth object segmentation masks by means of cosine similarity. *Bottom:* Comparison of NEON to other methods. For each method we show the average cosine score over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

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) feature maps (FM). The second baseline takes the top-k most activated locations (LO). In addition we consider a recently proposed method, NetDissect [84], 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.

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. The higher performance of NetDissect in higher-layer can be attributed to the smoother way it renders explanation in pixel-space (cf. Appendix J in the Supplement), thereby 'undoing' some of the invariances the neural network might have built.

Overall, our NEON approach allows to shed light into 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 the section above has demonstrated the multiple practical benefits one can get from bringing Explainable AI 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 simple 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.

In particular, we consider Integrated Gradients (IG) [29] where the explanation scores are computed by integrating the model output between the dataset mean $\bar{\boldsymbol{x}}$ and the data point $\boldsymbol{x}$ following some linear path. We then apply Prediction Difference Analysis (PDA) [56], [87] where we score the different dimension based on the effect on the decision function of removing the corresponding feature. (Missing feature is imputed using the same KDE model as used in our evaluation procedure.) Then, we consider Dimension Removal (DR), a variant of PDA, where we instead of removing the dimension i of the data point, we remove this dimension from all distance computations occurring in the computation of the neural network output. Finally, we include three simple baselines: random attribution, square difference $(\boldsymbol{x} - \widetilde{\boldsymbol{x}})^2$, which computes element-wise the square difference to some predefined reference point $\widetilde{\boldsymbol{x}}$, and sensitivity analysis $(\nabla f)^2$, which computes the square of the derivative along each input dimension.

A. Desiderata and Evaluation Metrics

In the context of explaining image classifiers, [88] proposed the 'pixel-flipping' technique for evaluating explanations. The

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

technique consists of constructing a plot that keeps track of decision function, specifically 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 [64] start from an 'empty' data point, and add the features from most to least relevant. Missing features are inpainted using a simple conditional kernel density estimation (KDE) model, the details of which we provide in Appendix G in the Supplement, 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. The higher the AUC, the better the explanation. The analysis can be extended to a whole dataset by computing by averaging the AUC obtained for each individual data point.

Algorithm 1 Area under the curve (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 $\iota \in$ `argsort`$((-R_i)_i)$ **do**
  $\mathcal{I} = \mathcal{I} \cup \{\iota\}$
  $\boldsymbol{x} \sim p_{\text{KDE}}(\boldsymbol{x} \mid \boldsymbol{z}_{\mathcal{I}})$
  `curve.append`$(g_c(\boldsymbol{x}))$
end for
return `area_under(curve)`

Consider now the five desiderata of an explanation listed in [89], 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 the few most relevant features returned by the explanation produce a substantial increase of 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 to more coarse-grained explanations.

To assess the fulfilment 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 Supplement), and where the number of clusters K is determined using the elbow method [90]. Then, we consider more complex kernel k-means models which we train on further datasets from the UCI dataset. We also consider the kernel k-means model trained on the 20newsgroup dataset [80] (`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 Supplement. Finally, we consider deep k-means models built on the popular STL-10 [91] 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 Eq. (3). For deep models, we choose β in the same way and furthermore choose the LRP rule LRP-γ [57], with the parameter γ set heuristically to 0.1.

After producing explanations for the cluster assignment of each model, we compute the AUC as follows: For k-means and kernel k-means, we iteratively add each individual input dimensions, and use kernel density estimation (KDE) inpainting for the missing features, with the KDE scale parameter chosen in a way that maximizes data likelihood (using leave-one-out cross-validation, cf. Appendix G). For deep k-means, we do not attribute directly on pixels but instead on low-level concepts as represented by the 256 feature maps at the output of block 3 of the VGG-16 network. Because the neural network ReLU activations have a natural reference point at zero (deactivated state), we inpaint by setting the missing features to zero. The results are shown in Table I.

We observe that the proposed NEON explanation method is equal or superior to all baselines on the vast majority of considered models and datasets. We note the relatively poor performance of method based on prediction difference analysis, where the removal of individual features seems insufficient to capture the more global structure of the cluster assignment.

1) Effect of K and d on NEON performance: 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

[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 [57], [62].

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 EXPLANATION METHOD. WE FIND THAT THE PROPOSED NEON METHOD SCORES THE HIGHEST FOR THE VAST MAJORITY OF MODELS. ENTRIES WHERE THE OTHER METHODS IN OUR BENCHMARK ARE INAPPLICABLE OR COMPUTATIONALLY PROHIBITIVE ARE DENOTED BY '—'.

dataset			model		methods							
name	N	D	K	type	random	$(\boldsymbol{x}-\widetilde{\boldsymbol{x}})^2$	PDA_0	PDA_{cs}	$(\nabla f_c)^2$	IG-10	DR	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	67.35	**74.92**
`sales`	811	52	6	kernel	82.30	**87.28**	82.66	82.33	86.72	86.51	83.47	87.21
`water`	527	38	5	kernel	79.30	87.30	79.13	78.89	85.01	86.82	83.55	**87.66**
`whlsl`	440	6	8	kernel	54.98	63.91	54.88	55.11	64.53	64.93	62.60	**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

training clustering model between $K = 2$ to $K = 64$, and also removing input features to generate dataset dimensions from $d = 2$ to $d = 13$. The results are shown in Fig. 9.

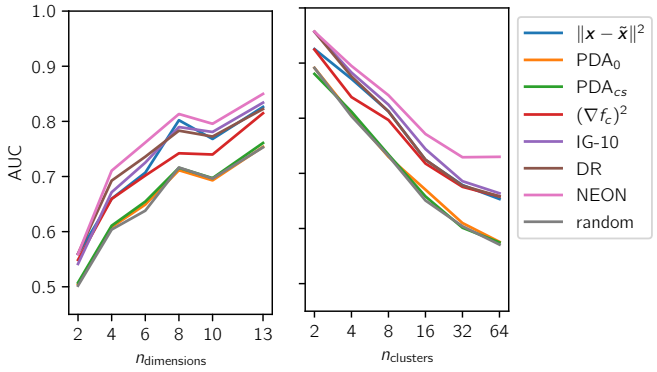

Fig. 9. Effect of the number of clusters n_{clusters} on the AUC performance of each explanation method on the `winer` dataset.

We observe that in every regime, NEON has equal or superior performance to all baselines. Anecdotally, NEON performs equivalently to IG-10 and DR for $K = 2$ (this can also be shown theoretically based on a similar argument to the one found in Proposition 3), but it start to outperform these methods as soon as the number of clusters grow.

2) Sensitivity of NEON to Hyperparameters: Unlike other baseline methods used in our benchmark, NEON comes with a 'stiffness' hyperparameter β for which we have proposed to choose heuristically following Eq. (3), and for deep clustering, with a 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).

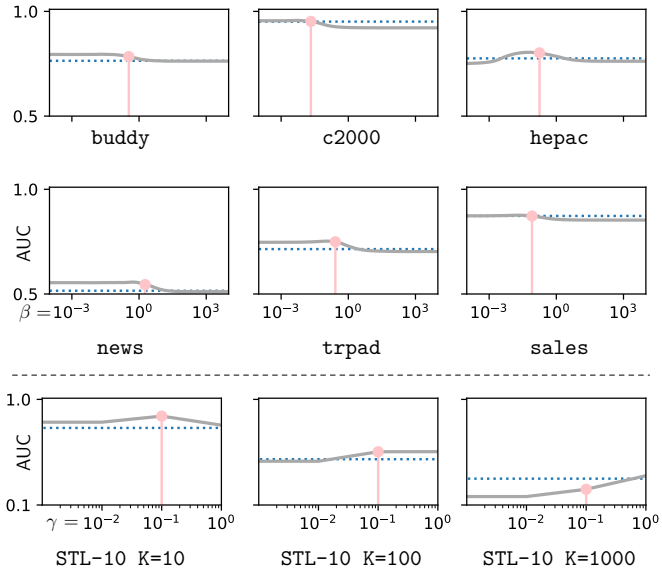

Fig. 10. Evaluating of NEON hyperparameters on a selection of clustering models. 1st row: k-means models, 2nd row: kernel k-means models, 3rd row: deep models (VGG-u / SCAN). The y-axis shows the pixel flipping AUC. The first two rows show the effect of the min-take-most parameter β, with the orange 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 $\gamma = 0.1$, and where we set $\beta = \mathbb{E}[f_c]^{-1}$.

We observe that the simple heuristic proposed in Eq. (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 hyperpa-

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	h_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	$\mathcal{O}(10Kd)$	$\mathcal{O}(10Kdp)$
DR	$(\boldsymbol{\mu}_c)_c$	$\mathcal{O}(Kd^2)$	$\mathcal{O}(Kd^2p)$
NEON	NN	$\mathcal{O}(Kd)$	$\mathcal{O}(Kdp^2)$

rameters brings slight additional gains on the AUC score. Notably, the seemingly limited performance of NEON on deep clustering 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.

C. Construction Overhead and Runtime

Lastly, we would like to study the fulfillment by NEON of desiderata **D4** (low construction overhead) and **D5** (runtime efficiency), comparatively to other methods in our benchmark. We resort to a qualitative analysis for **D4**, where we categorize methods according to what needs to be constructed 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, '$(\boldsymbol{u}_c)_c$' indicates that we need the cluster centroids, or the cluster support vectors for kernel k-means, 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 section, there is no significant obstacle to apply NEON on the studied models (k-means, kernel k-means, deep k-means, 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, NEON computational cost is lower than most explanation methods, by only require a single forward and backward pass, whereas many other explanation methods typically 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 Supplement.) For kernel k-means, results are more balanced, with NEON being slower an simple sensitivity analysis, but running faster than the more advanced PDA/DR and IG competitors if the number 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 need to additionally ensure that the number of support vectors remains small, typically less than 10.

Overall, we have demonstrated in our benchmark 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 Explainable AI 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.

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 a rich interpretable feedback into the nature of the clusters 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.

ACKNOWLEDGEMENTS

This work was supported by the German Ministry for Education and Research under Grant Nos. 01IS14013A-E, 01GQ1115, 01GQ0850, 01IS18025A, 031L0207D, and 01IS18037A, and the German Research Foundation (DFG) in the DAEDALUS graduate school. KRM was partly supported by the Institute for Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (No. 2017-0-00451, No. 2017-0-01779).

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, 1999.
[2] R. Xu and D. C. W. II, "Survey of clustering algorithms," *IEEE Trans. Neural Networks*, vol. 16, no. 3, pp. 645–678, 2005.
[3] A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1988.
[4] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. Springer New York, 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, 2004.

[6] H. Celiker and J. Gore, "Clustering in community structure across replicate ecosystems following a long-term bacterial evolution experiment," *Nature Communications*, 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 *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017, pp. 659–669.

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

[9] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*. Berkeley, Calif.: University of California Press, 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, 2000.

[11] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means: spectral clustering and normalized cuts," in *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2004, pp. 551–556.

[12] J. Xie, R. B. Girshick, and A. Farhadi, "Unsupervised deep embedding for clustering analysis," in *Proceedings of the 33nd International Conference on Machine Learning*, 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 *IEEE International Conference on Acoustics, Speech and Signal Processing*, 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 *ICML*, ser. Proceedings of Machine Learning Research, vol. 70. PMLR, 2017, pp. 3861–3870.

[15] M. Caron, P. Bojanowski, A. Joulin, and M. Douze, "Deep clustering for unsupervised learning of visual features," in *15th European Conference on Computer Vision*, 2018, pp. 139–156.

[16] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *ECCV (1)*, ser. Lecture Notes in Computer Science, vol. 8689. 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, p. e0130140, 2015.

[18] M. T. Ribeiro, S. Singh, and C. Guestrin, ""Why should I trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and 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 *IEEE International Conference on Computer Vision*, 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*. Springer Nature, 2019, vol. 11700.

[21] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digital Signal Processing*, vol. 73, pp. 1–15, 2018.

[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," *Proceedings of the IEEE*, vol. 109, no. 3, pp. 247–278, 2021.

[23] W. V. Gansbeke, S. Vandenhende, S. Georgoulis, M. Proesmans, and L. V. Gool, "SCAN: learning to classify images without labels," in *ECCV (10)*, ser. Lecture Notes in Computer Science, vol. 12355. 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 Genetics*, 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 Genetics*, vol. 45, no. 10, pp. 1127–1133, Sep. 2013.

[26] A. K. Kau, Y. E. Tang, and S. Ghose, "Typology of online shoppers," *Journal of Consumer 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 *IEEE International Symposium on Circuits and Systems*, 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," *Journal of Machine Learning Research*, vol. 11, pp. 1803–1831, 2010.

[29] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp. 3319–3328.

[30] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in Neural Information Processing Systems 30*, 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 Recognition*, vol. 65, pp. 211–222, 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 *IEEE Symposium on Computational Intelligence and Data Mining*, 2013, pp. 32–38.

[33] A. Shrikumar, P. Greenside, and A. Kundaje, "Learning important features through propagating activation differences," in *Proceedings of the 34th International Conference on Machine Learning*, 2017.

[34] J. Kauffmann, K.-R. Müller, and G. Montavon, "Towards explaining anomalies: A deep Taylor decomposition of one-class models," *Pattern Recognit.*, vol. 101, p. 107198, 2020.

[35] L. Arras, G. Montavon, K.-R. Müller, and W. Samek, "Explaining recurrent neural network predictions in sentiment analysis," in *Proceedings of the 8th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, 2017, pp. 159–168.

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

[37] D. Bertsimas, A. Orfanoudaki, and H. M. Wiberg, "Interpretable clustering via optimal trees," *CoRR*, vol. abs/1812.00539, 2018.

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

[39] M. Moshkovitz, S. Dasgupta, C. Rashtchian, and N. Frost, "Explainable k-means and k-medians clustering," in *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, ser. Proceedings of Machine Learning Research, vol. 119. PMLR, 2020, pp. 7055–7065.

[40] P. Geurts, N. Touleimat, M. Dutreix, and F. d'Alché-Buc, "Inferring biological networks with output kernel trees," *BMC Bioinform.*, vol. 8, no. S-2, 2007.

[41] K. G. Dizaji, A. Herandi, C. Deng, W. Cai, and H. Huang, "Deep clustering via joint convolutional autoencoder embedding and relative entropy minimization," in *ICCV*. IEEE Computer Society, 2017, pp. 5747–5756.

[42] X. Guo, X. Liu, E. Zhu, and J. Yin, "Deep clustering with convolutional autoencoders," in *ICONIP (2)*, ser. Lecture Notes in Computer Science, vol. 10635. Springer, 2017, pp. 373–382.

[43] X. Peng, J. T. Zhou, and H. Zhu, "k-meansnet: When k-means meets differentiable programming," *CoRR*, vol. abs/1808.07292, 2018.

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

[45] 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, 2004.

[46] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "On clustering validation techniques," *J. Intell. Inf. Syst.*, vol. 17, no. 2-3, pp. 107–145, 2001.

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

[48] M. Meila, "How to tell when a clustering is (approximately) correct using convex relaxations," in *Advances in Neural Information Processing Systems 31*, 2018, pp. 7418–7429.

[49] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to information retrieval*. Cambridge University Press, 2008.

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

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

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

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

[54] D. Sculley, "Web-scale k-means clustering," in *WWW*. ACM, 2010, pp. 1177–1178.

[55] J. O. Hanson, J. R. Rhodes, S. H. M. Butchart, G. M. Buchanan, C. Rondinini, G. F. Ficetola, and R. A. Fuller, "Global conservation of species' niches," *Nature*, vol. 580, no. 7802, pp. 232–234, Mar. 2020.

[56] L. M. Zintgraf, T. S. Cohen, T. Adel, and M. Welling, "Visualizing deep neural network decisions: Prediction difference analysis," in *ICLR (Poster)*. OpenReview.net, 2017.

[57] G. Montavon, A. Binder, S. Lapuschkin, W. Samek, and K.-R. Müller, "Layer-wise relevance propagation: An overview," in *Explainable AI*, ser. Lecture Notes in Computer Science. Springer, 2019, vol. 11700, pp. 193–209.

[58] S. Lapuschkin, A. Binder, K.-R. Müller, and W. Samek, "Understanding and comparing deep neural networks for age and gender classification," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 1629–1638.

[59] Y. Ding, Y. Liu, H. Luan, and M. Sun, "Visualizing and understanding neural machine translation," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics*, 2017, pp. 1150–1159.

[60] F. Horst, S. Lapuschkin, W. Samek, K.-R. Müller, and W. I. Schöllhorn, "Explaining the unique nature of individual gait patterns with deep learning," *Scientific Reports*, vol. 9, p. 2391, Feb. 2019.

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

[62] O. Eberle, J. Büttner, F. Krautli, K.-R. Müller, M. Valleriani, and G. Montavon, "Building and interpreting deep similarity models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[63] 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," *Information Fusion*, vol. 77, pp. 261–295, 2022.

[64] T. Schnake, O. Eberle, J. Lederer, S. Nakajima, K. T. Schütt, K.-R. Müller, and G. Montavon, "Higher-order explanations of graph neural networks via relevant walks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021.

[65] D. H. Hubel and T. N. Wiesel, "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex," *The Journal of Physiology*, 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 studies*, vol. 34, pp. 43–98, 1956.

[67] G. Montavon, "Gradient-based vs. propagation-based explanations: An axiomatic comparison," in *Explainable AI*, ser. Lecture Notes in Computer Science. Springer, 2019, vol. 11700, pp. 253–265.

[68] L. S. Shapley, "17. a value for n-person games," in *Contributions to the Theory of Games (AM-28), Volume II*. Princeton University 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, 2010.

[70] B. Schölkopf, S. Mika, C. J. C. Burges, P. Knirsch, K.-R. Müller, G. Rätsch, and A. J. Smola, "Input space versus feature space in kernel-based methods," *IEEE Trans. Neural Networks*, vol. 10, no. 5, pp. 1000–1017, 1999.

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

[72] B. Schölkopf and A. J. Smola, *Learning with Kernels: support vector machines, regularization, optimization, and beyond*. MIT Press, 2002.

[73] R. Zhang and A. I. Rudnicky, "A large scale clustering scheme for kernel k-means," in *16th International Conference on Pattern Recognition*, 2002, pp. 289–292.

[74] L. Arras, J. A. Arjona-Medina, M. Widrich, G. Montavon, M. Gillhofer, K.-R. Müller, S. Hochreiter, and W. Samek, "Explaining and interpreting LSTMs," in *Explainable AI*, ser. Lecture Notes in Computer Science. Springer, 2019, vol. 11700, pp. 211–238.

[75] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 1996, pp. 226–231.

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

[77] M. Meila and J. Shi, "Learning segmentation by random walks," in *Advances in Neural Information Processing Systems 13*, 2000, pp. 873–879.

[78] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems 14*, 2001, pp. 849–856.

[79] 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 Communications*, vol. 10, p. 1096, 2019.

[80] T. Joachims, "A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization," in *Proceedings of the Fourteenth International Conference on Machine Learning*, 1997, pp. 143–151.

[81] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *1st International Conference on Learning Representations*, 2013.

[82] 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, p. e0181142, 2017.

[83] 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 *Advances in Neural Information Processing Systems 29*, 2016, pp. 3387–3395.

[84] B. Zhou, D. Bau, A. Oliva, and A. Torralba, "Interpreting deep visual representations via network dissection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1–1, 2018.

[85] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations*, 2015.

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

[87] I. Covert, S. Lundberg, and S. Lee, "Explaining by removing: A unified framework for model explanation," *CoRR*, vol. abs/2011.14878, 2020.

[88] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K.-R. Müller, "Evaluating the visualization of what a deep neural network has learned," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 11, pp. 2660–2673, 2017.

[89] W. R. Swartout and J. D. Moore, *Explanation in Second Generation Expert Systems*. Berlin, Heidelberg: Springer-Verlag, 1993, p. 543–585.

[90] V. Satopaa, J. Albrecht, D. Irwin, and B. Raghavan, "Finding a "kneedle" in a haystack: Detecting knee points in system behavior," in *2011 31st International Conference on Distributed Computing Systems Workshops*, 2011, pp. 166–171.

[91] A. Coates, A. Y. Ng, and H. Lee, "An analysis of single-layer networks in unsupervised feature learning," in *AISTATS*, ser. JMLR Proceedings, vol. 15. JMLR.org, 2011, pp. 215–223.

Jacob Kauffmann received a Bachelors degree in Computer Science from TU Berlin in 2014 and a Masters degree in Computer Science from TU Berlin in 2017. He is currently a Ph. D. student in the Machine Learning Group at TU Berlin.

Malte Esders received a Bachelors degree in Cognitive and Neurobiological Psychology from Utrecht University in 2014, and a Masters degree in Computational Neuroscience from TU Berlin in 2017. He is currently a Ph. D. student in the Machine Learning Group at TU Berlin.

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 (HU Berlin), Berlin, Germany, the Technische Universität Berlin (TU Berlin), Berlin, and Freie Universität Berlin (FU Berlin), Berlin, in 2017. He finished his Ph.D. degree in 2021 at the Machine Learning Group, TU Berlin specializing on anomaly detection. He is now with AIGNOSTICS a Berlin-based start-up applying ML methods for digital pathology .

Grégoire Montavon is a senior researcher in the Machine Learning Group at the Technische Universität Berlin, and in the Berlin Institute for the Foundations of Learning and Data (BIFOLD). He received a Masters degree in Communication Systems from École Polytechnique Fédérale de Lausanne in 2009, and a Ph.D. degree in Machine Learning from the Technische Universität Berlin in 2013. He is a member of the ELLIS Unit Berlin, and an editorial board member of Pattern Recognition. He is recipient of the 2020 Pattern Recognition Best Paper Award. His research interests include explainable machine learning, deep neural networks, and unsupervised learning.

Wojciech Samek (M'13) is head of the Department of Artificial Intelligence and the Explainable AI Group at Fraunhofer Heinrich Hertz Institute, Berlin, Germany. He studied computer science at Humboldt University of Berlin, from 2004 to 2010, and received the Ph.D. degree with distinction from the Technical University of Berlin in 2014. During his studies he was awarded scholarships from the German Academic Scholarship Foundation and the DFG Research Training Group GRK 1589/1, and was a visiting researcher at NASA Ames Research Center, Mountain View, USA. He is associated faculty at the Berlin Institute for the Foundation of Learning and Data (BIFOLD), the ELLIS Unit Berlin and the DFG Graduate School BIOQIC. Furthermore, he is an editorial board member of Pattern Recognition, PLoS ONE and IEEE TNNLS, and an elected member of the IEEE MLSP Technical Committee. He has been serving as an AC for NAACL'21, and was a recipient of multiple best paper awards, including the 2020 Pattern Recognition Best Paper Award. His research interest include deep learning, explainable AI, neural network compression, and federated learning.

Klaus-Robert Müller (M'12) has been a professor of computer science at Technische Universität Berlin since 2006; at the same time he is co-directing the Berlin Big Data Center. He studied physics in Karlsruhe from 1984 to 1989 and obtained his Ph.D. degree in computer science at Technische Universität Karlsruhe in 1992. After completing a postdoctoral position at GMD FIRST in Berlin, he was a research fellow at the University of Tokyo from 1994 to 1995. In 1995, he founded the Intelligent Data Analysis group at GMD-FIRST (later Fraunhofer FIRST) and directed it until 2008. From 1999 to 2006, he was a professor at the University of Potsdam. He was awarded the Olympus Prize for Pattern Recognition (1999), the SEL Alcatel Communication Award (2006), the Science Prize of Berlin by the Governing Mayor of Berlin (2014), and the Vodafone Innovations Award (2017). In 2012, he was elected member of the German National Academy of Sciences-Leopoldina, in 2017 of the Berlin Brandenburg Academy of Sciences and also in 2017 external scientific member of the Max Planck Society. Form 2019 on he became ISI Highly Cited Researcher. His research interests are intelligent data analysis and machine learning with applications in neuroscience (specifically brain-computer interfaces), medicine, physics, chemistry and the humanities.