Cluster Purging: Efficient Outlier Detection Based on Rate-Distortion Theory

Maximilian B. Toller, Bernhard C. Geiger, *Senior Member, IEEE*, and Roman Kern

Abstract—Rate-distortion theory-based outlier detection builds upon the rationale that a good data compression will encode outliers with unique symbols. Based on this rationale, we propose Cluster Purging, which is an extension of clustering-based outlier detection. This extension allows one to assess the representivity of clusterings, and to find data that are best represented by individual unique clusters. We propose two efficient algorithms for performing Cluster Purging, one being parameter-free, while the other algorithm has a parameter that controls representivity estimations, allowing it to be tuned in supervised setups. In an experimental evaluation, we show that Cluster Purging improves upon outliers detected from raw clusterings, and that Cluster Purging competes strongly against state-of-the-art alternatives.

Index Terms—Outlier detection, clustering algorithms, rate-distortion theory

1 Introduction

In present days, there exists an abundance of datasets containing individual observations that greatly deviate from the remaining observations, commonly called *outliers* or *anomalies*. The task of finding such outlying/anomalous observations in datasets is relevant in a multitude of applications and has received much attention in the last decades [1]. Traditionally, outlier detection was mostly approached from a statistical perspective, where data are modeled with distributions, while recently database-oriented methods that focus on efficiency and scalability have become more popular [2]. A major part of contemporary research concentrates on using deep learning to detect outliers in semi-supervised [3], [4] or unsupervised [5], [6], [7] settings. These approaches are well motivated for high-dimensional datasets and have yielded significantly improved outlier detection accuracy on benchmark datasets [8], [9], [10], yet deep learning techniques are also criticized for being data hungry [11] and lacking interpretability [12]. Both of these deficits gravely affect outlier detection since in many research fields large training datasets are not available [4]. Further, outlier detection techniques are commonly used in high-risk applications such as intrusion detection [1], where black-box models should generally be avoided [13].

In contrast, *clustering-based* outlier detection methods [1] resort to very intuitive concepts of what an outlier might possibly be; for instance observations that have abnormal local density [14]; or observations that do not fit well into any cluster [15], [16], [17]. A trait that these methods have in common is that they detect outliers during clustering, for instance by assigning outliers to a special outlier cluster. While this trait can be advantageous in several settings, it also has the downside that outliers are only detected as a "side-product" of clustering [1]. As a consequence, outliers detected by methods such as [14], [15], [16], [17] are observations that are irregular in the respective clustering, yet not necessarily irregular with respect to the (unclustered) data.

Another type of clustering-based methods infers outliers after the raw data were clustered. For instance, the Cluster-Based Local Outlier Factor (CBLOF) [18] scales distances between observations and cluster centers by cluster sizes, regardless of which clustering technique was used. Hence, CBLOF allows one to choose a clustering method that is well-suited for the data at hand. However, outlier detection techniques such as CBLOF [18], [19], [20] still have the same drawback as the methods mentioned above: They assume that the computed clustering is sufficient for describing outliers in raw data, which can be problematic in scenarios where it is challenging to perform a good clustering, e.g., in high-dimensional data [21].

To address this issue, one may resort to information theory. From an information-theoretic perspective, a clustering is a lossy compression of the raw data [22], where a raw observation is represented by the cluster it was assigned to. The loss (distortion) that occurs during such a clustering-compression can be combined with a cluster's degree of compression (rate) to quantify how well this cluster represents the observations that are assigned to it. Further, rate-distortion theory allows one to infer how the representivity of a clustering would change if one were to modify this clustering, and which observations would be better represented by different clusters (cf. [23], [24]). Observations that are hard to represent by a meaningful cluster and that are best represented by themselves can then be considered as outliers.

- *Maximilian B. Toller and Bernhard C. Geiger are with the Know Center GmbH, 8010 Graz, Austria. E-mail: {mtoller, bgeiger}@know-center.at.*
- *Roman Kern is with the Graz University of Technology, 8010 Graz, Austria. E-mail: rkern@tugraz.at.*

Manuscript received 7 Oct. 2020; revised 2 June 2021; accepted 27 July 2021. Date of publication 10 Aug. 2021; date of current version 10 Jan. 2023. (Corresponding author: Maximilian B. Toller.) Recommended for acceptance by P. Bogdanov. Digital Object Identifier no. 10.1109/TKDE.2021.3103571

This description outlines a technique that we refer to as *Cluster Purging*, in analogy to the act of purging in authoritarian political systems where deviating individuals that are not well-represented by such systems are removed from society.[1] In short, Cluster Purging is performed by modifying a clustering (or by analyzing a set of given clusterings), and then isolating observations that are not represented well by their cluster, regardless of how one modifies it (or which of the clusterings one considers). As such, Cluster Purging is, to the best of our knowledge, a conceptually novel approach to cluster-based outlier detection, and the main contributions of this work stem from it:

- Review of related work, outlining the differences between Cluster Purging and existing methods (Section 2).
- Theoretical formalization of Cluster Purging and description of required concepts from information theory (Section 3).
- Description of a parameter-free algorithm for Cluster Purging and discussion of various aspects that are relevant in practice, i.e., efficiency, interpretation of proposed outliers, how one can introduce parameters for improved performance, and limitations (Section 4).
- Empirical demonstration that Cluster Purging improves upon outliers detected from clustering alone, and that Cluster Purging strongly competes against state-of-the-art alternatives (Section 5).

2 Related Work

In general, cluster-based outlier detection techniques can be split into three categories depending on how they define outliers [1]:

1) Outliers are observations that do not fit into any cluster.
2) Outliers are far away from their cluster's centroid.
3) Outliers are assigned to small or sparse clusters.

Conceptually, category 1 is most closely related to Cluster Purging, since in our method outliers are observations that cannot be represented well by any cluster. There are several existing methods that fall into category 1, for instance Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [14], extensions of DBSCAN such as [25], [26], k-Means— [15] and k-Means with Outlier Removal [16]. However, a key difference between these methods and Cluster Purging is that our method is not bound to a specific clustering. Even if one bases Cluster Purging on one of the above clusterings, the results can be very different since our method does not assume that a single clustering necessarily describes outliers in the raw data.

Surprisingly, one can argue that our method should also fall into category 2, since the theoretical formulation of Cluster Purging permits setups where outliers are observations that are far away from a centroid (see Section 3). Related methods from this category are techniques that combine centroid-based clusterings with a distance threshold, for instance [20], [27]. One can distinguish Cluster Purging from these methods by the simple fact that our method does not require a distance threshold (although Cluster Purging can be adapted to require one, should an application demand this (see Section 4)).

Typical methods of the third category are Local Outlier Factor [28] and its numerous variants, e.g., [29], [30], [31]. The Cluster-Based Local Outlier Factor (CBLOF) [18] is particularly noteworthy, since this method is directly applicable to any clustering, similar to Cluster Purging. The main difference between CBLOF and Cluster Purging is that, while our method can be based on local densities, it does not require a threshold parameter to infer critical differences in local densities and does not consider a single clustering as sufficient for describing outliers.

From a theoretical perspective, the most closely related method to ours is the one-class rate-distortion model (OCRD) [32]. The brief description of Cluster Purging given above can be seen as a single (half-)step of the Blahut-Arimoto algorithm [23], [24], [33], which OCRD adapts for one-class classification. However, while OCRD is optimal in a rate-distortion theoretic sense, we here do not aim for this optimality. Instead, Cluster Purging supports arbitrary clustering techniques, allowing for a greater flexibility. In our experiments, we demonstrate that rate-distortion optimal clusterings are not necessarily optimal for detecting outliers in real data (Section 5).

3 Theoretical Formulation

In this section, the theoretical background of Cluster Purging is explained and the concept of representivity is introduced. In short, clustering can be interpreted as a form of data compression that yields cluster assignments and a representation. One can measure how representative such a representation is via its surplus complexity when compared to the most representative clustering at a given inaccuracy. Since directly finding the most representative clustering is often infeasible, we show how representivity can be efficiently estimated from a small set of available clusterings. Finally, we show how one can detect outliers under the premise that a good clustering would represent outliers by themselves, i.e., with an additional cluster.

3.1 Background

3.1.1 Data Compression

Let $\boldsymbol{x} = \{x_1, \ldots, x_n\}$ be a dataset of n observations in $\mathbb{R}^d$ consisting of $u \approx n$ unique values. A common data analysis goal is to obtain a representation of $\boldsymbol{x}$ that has fewer unique values without losing too much information [34], [35], [36]. In coding theory, the task of finding such a representation consisting of $\nu \ll u$ unique symbols is referred to as lossy data compression. Clustering can be seen as a typical example for lossy data compression. In detail, a successful compression via (non-fuzzy) clustering yields two objects

1) A list of n cluster assignments $\boldsymbol{c} = (c_1, \ldots, c_n)$, where $c_j \in 1, \ldots, \nu$ is the index of the cluster that contains observation x_j.
2) A low-dimensional representation $\boldsymbol{r} = (r_1, \ldots, r_\nu)$ describing ν different clusters.

1. None of the authors or their affiliations approve of political purges in any form.

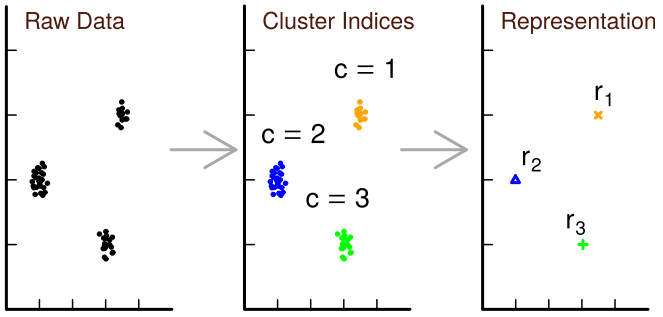

Fig. 1. Compression via k-means clustering. *Left*: A dataset consisting of 65 observations. *Middle*: Cluster assignments, indicated by color. *Right*: Symbols representing each cluster.

A visualization can be seen in Fig. 1. Not all clustering techniques return both of these objects, e.g., DBSCAN only gives cluster assignments $\boldsymbol{c}$ yet no representation $\boldsymbol{r}$. Details on how to obtain representations in such cases are given in Section 4.4.

Further, assume that a small subset of outliers $\boldsymbol{x_y} = \{x_{y_1}, \ldots, x_{y_m}\}$ with $m \ll n$ is part of the dataset. Since outliers are commonly assumed to deviate significantly from the remaining observations [37], compressing a dataset that contains outliers will either require additional unique symbols for outliers or else lead to a less effective compression [38]. Let

$$d(\boldsymbol{x}, \boldsymbol{r}) = \sum_{j=1}^{n} d(x_j, r_{c_j}), \tag{1}$$

be a separable distortion function, i.e., a measure describing how accurately $\boldsymbol{r}$ represents dataset $\boldsymbol{x}$. If an outlier is represented by the same symbol as an inlier, then this will increase the overall distortion since inliers and outliers are assumed to be dissimilar. Consequently, one can reduce the overall distortion by compressing outliers to unique symbols. In the context of clusterings, this translates to assigning outliers to singleton clusters, i.e., an additional cluster that only contains x_{y_j}. However, adding unique outlier clusters also increases the overall *complexity* of the compression.

3.1.2 The Empirical Rate-Distortion Function

Rate-distortion theory seeks to describe this trade-off between representation complexity (*rate*) and inaccuracy (*distortion*) in the context of random variables. Formally, the rate-distortion function $R(D)$ of a random variable X is defined as (cf. [33])

$$R(D) = \min_{P(\hat{X}|X)} H(\hat{X}) - H(\hat{X}|X) \textbf{ subject to } d(X, \hat{X}) \leq D, \tag{2}$$

where $P(\cdot)$ and $H(\cdot)$ are the probability and entropy functions, respectively, $\hat{X}$ is a stochastic compression of X, and D is a specific distortion value, e.g., the sum of squared errors in a k-means clustering. Intuitively, the rate-distortion function describes the smallest complexity one can achieve while compressing X at a given distortion, regardless of how the compression is performed.

To transfer this stochastic definition into a real-data context, let

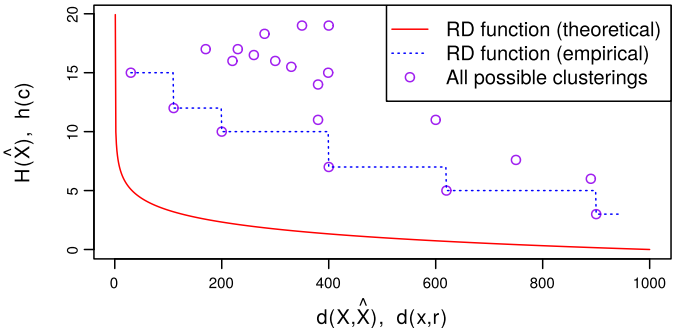

Fig. 2. Comparison of theoretical and empirical rate-distortion functions.

$$h(\boldsymbol{c}) = -\sum_{f \in \boldsymbol{f}^{\boldsymbol{c}}} \frac{f}{n} \log \frac{f}{n}, \tag{3}$$

be the empirical counterpart to the theoretical entropy $H(\hat{X})$ as per [33], where $\boldsymbol{f}^{\boldsymbol{c}} = \{f_1^c, \ldots, f_v^c\}$ are the numbers of observations assigned to each cluster. Then, inspired by (2), we define the empirical rate-distortion function of a dataset $\boldsymbol{x}$ as

$$R(D, \boldsymbol{x}, C) := \min_{\{C(\boldsymbol{x},\boldsymbol{\theta}):\boldsymbol{\theta} \in \boldsymbol{\Theta}\}} h(\boldsymbol{c}) \textbf{ subject to } d(\boldsymbol{x}, \boldsymbol{r}) \leq D, \tag{4}$$

with $C(\boldsymbol{x}, \boldsymbol{\theta}) = (\boldsymbol{c}, \boldsymbol{r})$, where $C(\cdot)$ is a deterministic compression function (i.e., a non-fuzzy clustering technique) and $\boldsymbol{\theta}$ are its parameters and where $\boldsymbol{\Theta}$ is the set of all possible parametrizations. Intuitively, the empirical rate-distortion function can be seen as the strongest degree of compression one can achieve on a dataset with a fixed compression method without exceeding the required distortion.

As such, it describes the trade-off between compression complexity and inaccuracy for a fixed dataset and a specific clustering method. The term $h(\boldsymbol{c}|\boldsymbol{x})$ was omitted from (4), since $h(\boldsymbol{c}|\boldsymbol{x}) = 0$ for all non-fuzzy clustering techniques. A visualization of theoretical and empirical rate-distortion functions is depicted in Fig. 2.

3.2 Measuring Cluster Representivity

3.2.1 Theoretical Representivity

From a rate-distortion theoretical perspective, there are two quantities that measure how "good" a clustering $(\boldsymbol{c}, \boldsymbol{r})$ represents the raw data

1) The degree of compression (the rate), computed via entropy $h(\boldsymbol{c})$;
2) How accurate the representation is (the distortion), computed via distortion $d(\boldsymbol{x}, \boldsymbol{r})$.

While the empirical rate-distortion function $R(D, \boldsymbol{x}, C)$ describes the best achievable trade-off between these quantities in a given setup, the average result of a clustering algorithm typically offers a worse trade-off. More concretely, for every clustering $C(\boldsymbol{x}, \boldsymbol{\theta}) = (\boldsymbol{c}, \boldsymbol{r})$ it holds that

$$R(d(\boldsymbol{x}, \boldsymbol{r}), \boldsymbol{x}, C) \leq h(\boldsymbol{c}), \tag{5}$$

since the rate-distortion function describes the global minimum over all parametrizations, i.e the best achievable representation at distortion $d(\boldsymbol{x}, \boldsymbol{r})$. Due to this inequality

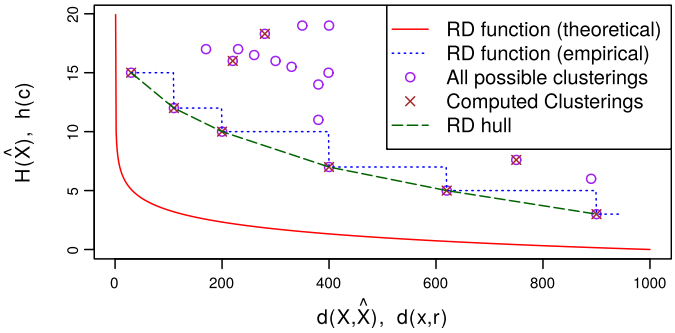

Fig. 3. Comparison of theoretical rate-distortion function, empirical rate-distortion function and rate-distortion hull. If ideal clusterings are selected for estimating the empirical rate-distortion function, then the resulting rate-distortion hull is equal to the lower convex hull of the empirical rate-distortion function.

there is always a nonnegative surplus complexity between $(\boldsymbol{c}, \boldsymbol{r})$ and (4). Thus, one can measure the theoretical representivity of a clustering via

$$\rho(\boldsymbol{x}, \boldsymbol{c}, \boldsymbol{r}, C) := R(d(\boldsymbol{x}, \boldsymbol{r}), \boldsymbol{x}, C)/h(\boldsymbol{c}). \tag{6}$$

However, computing $R(d(\boldsymbol{x}, \boldsymbol{r}), \boldsymbol{x}, C)$ and thus $\rho(\boldsymbol{x}, \boldsymbol{c}, \boldsymbol{r}, C)$ is infeasible for many clustering techniques, since this would require one to compute $C(\boldsymbol{x}, \boldsymbol{\theta})$ for all possible clustering parameters $\boldsymbol{\theta}$. Therefore, it is more practical to estimate clustering representivity relative to a small set of representations, obtained from parametrizations $\{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_t\}$. We refer to this estimate as rate-distortion *hull*.

Definition 1. Rate-distortion hull. *Let $\underline{\boldsymbol{c}} = (\boldsymbol{c}_1, \ldots, \boldsymbol{c}_t)$ and $\underline{\boldsymbol{r}} = (\boldsymbol{r}_1, \ldots, \boldsymbol{r}_t)$ be a set of clustering assignments and representations, respectively, obtained by evaluating clustering technique $C(\cdot)$ on dataset $\boldsymbol{x}$ with parametrizations $\{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_t\}$. Further, let $\boldsymbol{v} = [v_1, \ldots, v_s]$ be the indices of the lower convex hull of the arising distortion-entropy pairs $\{[d(\boldsymbol{x}, \boldsymbol{r}_1), h(\boldsymbol{c}_1)], \ldots, [d(\boldsymbol{x}, \boldsymbol{r}_t), h(\boldsymbol{c}_t)]\}$. Then, the rate-distortion hull of $\underline{\boldsymbol{c}}$ and $\underline{\boldsymbol{r}}$ is given by*

$$\begin{aligned} &\mathcal{L}(D, \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}}) := \kappa_i D + \delta_i \quad \forall i \in \{2, \ldots, s\} \\ &D \in [d(\boldsymbol{x}, \boldsymbol{r}_{v_1}), d(\boldsymbol{x}, \boldsymbol{r}_{v_s})], \end{aligned} \tag{7}$$

where

$$\kappa_i = \frac{h(\boldsymbol{c}_{v_i}) - h(\boldsymbol{c}_{v_{i-1}})}{d(\boldsymbol{x}, \boldsymbol{r}_{v_i}) - d(\boldsymbol{x}, \boldsymbol{r}_{v_{i-1}})}, \tag{8}$$

and

$$\delta_i = h(\boldsymbol{c}_{v_i}) - \kappa_i \cdot d(\boldsymbol{x}, \boldsymbol{r}_{v_i}), \tag{9}$$

are the slopes and vertical intercepts of the arising linear pieces, with $d(\boldsymbol{x}, \boldsymbol{r}_{v_1}) < \cdots < d(\boldsymbol{x}, \boldsymbol{r}_{v_s})$.

Intuitively, a rate-distortion hull is a linear interpolation of the lower convex hull of the entropy and distortion values associated with observed clusterings $(\underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$. A visualization of a rate-distortion hull is shown in Fig. 3.

Further, since $\mathcal{L}(\cdot, \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}}) = \mathcal{L}(\cdot, \underline{\boldsymbol{c}}_{\boldsymbol{v}}, \underline{\boldsymbol{r}}_{\boldsymbol{v}})$, we assume without loss of generality that $v_i = i$ and $s = t$ to keep the notation simple.

3.2.2 Representivity After Modification

Naturally, it is not possible to directly estimate the theoretical representivity of clusterings $(\underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$ based on a rate-distortion hull $\mathcal{L}(\cdot, \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$ constructed from the same clusterings. However, one can use $\mathcal{L}(\cdot, \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$ for estimating how the representivity of a particular clustering $(\boldsymbol{c}_i, \boldsymbol{r}_i) \in (\underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$ reacts to arbitrary modifications via

$$\hat{\rho}(\boldsymbol{x}, \boldsymbol{c}_i', \boldsymbol{r}_i', \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}}) := \mathcal{L}(d(\boldsymbol{x}, \boldsymbol{r}_i'), \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})/h(\boldsymbol{c}_i'), \tag{10}$$

where $\boldsymbol{c}_i'$ and $\boldsymbol{r}_i'$ are arbitrarily modified versions of $\boldsymbol{c}_i$ and $\boldsymbol{r}_i$ respectively, with

$$\boldsymbol{c}_i' \notin \underline{\boldsymbol{c}} \quad \text{and} \quad \boldsymbol{r}_i' \notin \underline{\boldsymbol{r}}.$$

Note that the error between measurements $\hat{\rho}(\boldsymbol{x}, \boldsymbol{c}_i', \boldsymbol{r}_i', \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$ and $\rho(\boldsymbol{x}, \boldsymbol{c}_i', \boldsymbol{r}_i', C)$ will not only depend on the clusterings used for constructing the rate-distortion hull. It will also depend on how many $c \in \boldsymbol{c}_i$ and $r \in \boldsymbol{r}_i$ were modified. Generally speaking, the more similar modified clustering $(\boldsymbol{c}_i', \boldsymbol{r}_i')$ is to $(\boldsymbol{c}_i, \boldsymbol{r}_i)$, the smaller the error between $\hat{\rho}(\boldsymbol{x}, \boldsymbol{c}_i', \boldsymbol{r}_i', \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$ and $\rho(\boldsymbol{x}, \boldsymbol{c}_i', \boldsymbol{r}_i', C)$ will be.

3.3 Detecting Outliers With Cluster Representivity

3.3.1 Definition of Rate-Distortion Outliers

Since $\hat{\rho}(\boldsymbol{x}, \cdot, \cdot, \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$ allows one to measure the effect of arbitrary modifications to a clustering, one can also measure how assigning an individual observation to a new, unique cluster would affect representivity. Now recall from above that an outlier is an observation that will likely need a unique symbol for an effective compression [38]. If changing the cluster assignment of observation x_j in $\boldsymbol{c}_i$ to a new additional cluster would improve $\boldsymbol{r}_i$'s representivity, then x_j should be labeled as outlier. This intuition can be formalized as follows.

Definition 2. *Rate-distortion outlier. Let $\boldsymbol{x}$ be a dataset and $(\underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$ a set of clusterings. Then observation x_j is a rate-distortion outlier if*

$$\hat{\rho}\left(\boldsymbol{x}, \boldsymbol{c}_{(i,j)}', \boldsymbol{r}_{(i,j)}', \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}}\right) \geq 1 \quad \forall i \in [2 \ldots, t], \tag{11}$$

with

$$\boldsymbol{c}_{(i,j)}' = (c_{i,1}, \ldots, c_{i,j-1}, \nu + 1, c_{i,j+1}, \ldots, c_{i,n}), \tag{12}$$

and

$$\boldsymbol{r}_{(i,j)}' = (r_{i,1}, \ldots, r_{i,\nu}, r^*), \tag{13}$$

where r^* is a representation of x_j such that $d(x_j, r^*) = 0$.

In simple terms, Definition 2 states that x_j is a rate-distortion outlier if assigning it to r^* would improve the representivity of all clusterings $(\underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$.

3.3.2 Computation of $\hat{\rho}(\boldsymbol{x}, \boldsymbol{c}_{(i,j)}', \boldsymbol{r}_{(i,j)}', \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$

A key advantage of defining outliers as in Definition 2 is that $\hat{\rho}(\boldsymbol{x}, \cdot, \cdot, \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$ can be computed for $\boldsymbol{c}_{(i,j)}'$ and $\boldsymbol{r}_{(i,j)}'$ from a set of clusterings $(\underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$ in $\mathcal{O}(n)$ time. This works, since the change in entropy from $\boldsymbol{c}_i$ to $\boldsymbol{c}_{(i,j)}'$ and the change in distortion from $\boldsymbol{r}_i$ to $\boldsymbol{r}_{(i,j)}'$ can be computed independently from the remaining clusterings in $(\underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$.

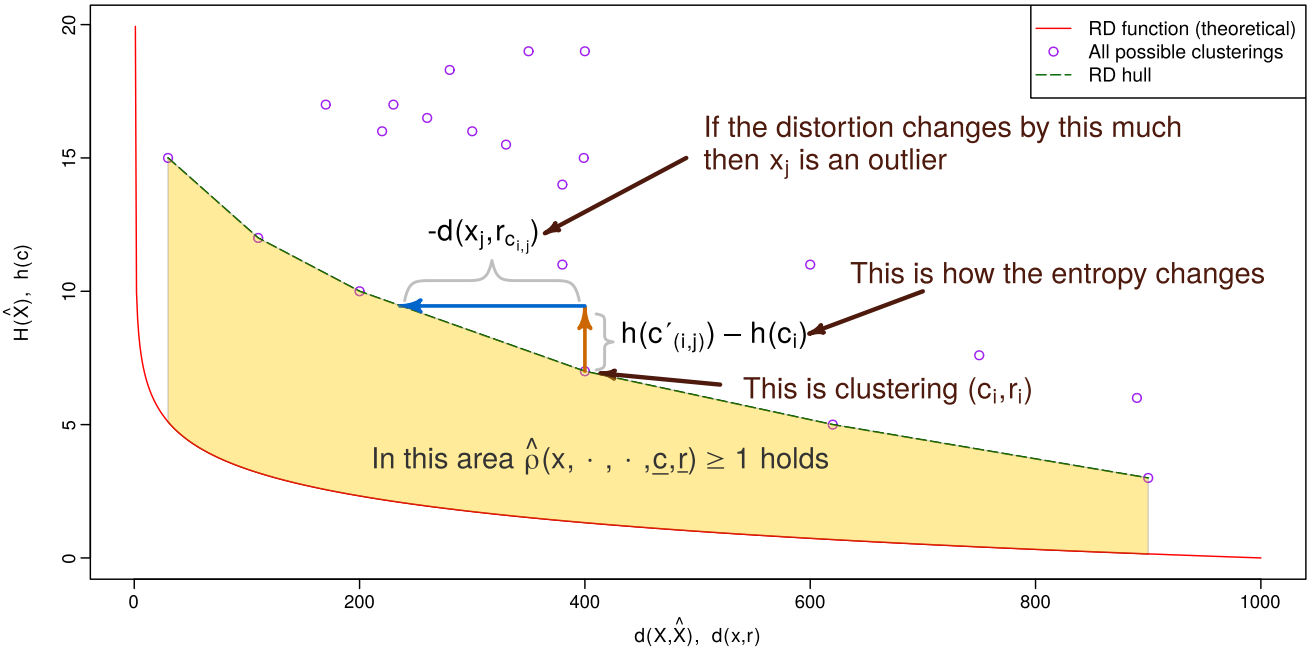

Fig. 4. Geometric interpretation of the computation of cluster representivity. For every clustering on the rate-distortion hull, one can compute how the entropy would change if x_j were represented by a new cluster. If this new clustering has a distortion that is sufficiently small to enter the area beneath the rate-distortion hull, then x_j is an outlier that needs to be represented by itself rather than a cluster.

Proposition 1. *Let $\boldsymbol{c}$ be a list of cluster assignments and let $\boldsymbol{f^c} = \{f_1^{\boldsymbol{c}}, \ldots, f_v^{\boldsymbol{c}}\}$ be the numbers of observations assigned to each cluster. Then the change in entropy caused by assigning x_j to an additional unique cluster, yielding $\boldsymbol{c}'$, depends only on $f_{c_j}^{\boldsymbol{c}}$ and is given by*

$$h(\boldsymbol{c}') - h(\boldsymbol{c}) = \frac{1}{n}\Big(f_{c_j}^{\boldsymbol{c}} \log f_{c_j}^{\boldsymbol{c}} - (f_{c_j}^{\boldsymbol{c}} - 1)\log\,(f_{c_j}^{\boldsymbol{c}} - 1)\Big). \tag{14}$$

Proof. The entropy of $\boldsymbol{c}$ as given in (3) can be rewritten as

$$h(\boldsymbol{c}) = \log n - \frac{1}{n}\sum_{f \in \boldsymbol{f^c}} f \log f, \tag{15}$$

since $\log \frac{f}{n} = \log f - \log n$. The entropy of $\boldsymbol{c}'$ is given by

$$h(\boldsymbol{c}') = \log n - \frac{1}{n}\sum_{f \neq f_{c_j}^{\boldsymbol{c}}} f \log f \quad - \frac{1}{n}\Big((f_{c_j}^{\boldsymbol{c}} - 1)\log\,(f_{c_j}^{\boldsymbol{c}} - 1)\Big), \tag{16}$$

since 1 observation is removed from cluster c_j and a unique cluster is added with entropy $1 \log 1 = 0$. Subtracting (15) from (16) yields (14). □

The change in distortion from $\boldsymbol{r}_i$ to $\boldsymbol{r}'_{(i,j)}$ is given by

$$d(\boldsymbol{x}, \boldsymbol{r}'_{(i,j)}) - d(\boldsymbol{x}, \boldsymbol{r}_i) = -d(x_j, r_{c_{i,j}}), \tag{17}$$

which follows by assumption from Definition 2. Intuitively, when one assigns x_j to a new unique symbol, then this symbol perfectly represents x_j and hence the total distortion decreases by $d(x_j, \boldsymbol{r}_{i,j})$. Note that (17) only depends on observation x_j and the cluster representative x_j is assigned to, i.e., $r_{c_{i,j}}$.

To evaluate $\hat{\rho}(\boldsymbol{x}, \boldsymbol{c}'_{(i,j)}, \boldsymbol{r}'_{(i,j)}, \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$, one can combine (14) and (17) in the following way:

Proposition 2. *Let $\boldsymbol{x}$ be a dataset and $(\underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$ a set of clusterings. If $\boldsymbol{c}'_{(i,j)}$ and $\boldsymbol{r}'_{(i,j)}$ are defined as in (12) and (13), respectively, then it holds that*

$$\begin{aligned} &\hat{\rho}\Big(\boldsymbol{x}, \boldsymbol{c}'_{(i,j)}, \boldsymbol{r}'_{(i,j)}, \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}}\Big) \geq 1 \\ \Leftrightarrow \\ &d(x_j, r_{c_{i,j}}) \geq \frac{h(\boldsymbol{c}'_{(i,j)}) - h(\boldsymbol{c}_i)}{-\kappa_i}, \end{aligned} \tag{18}$$

where κ_i is the slope of the rate-distortion hull between $d(\boldsymbol{x}, \boldsymbol{r}_{i-1})$ and $d(\boldsymbol{x}, \boldsymbol{r}_i)$, with $i \neq 1$.

Note that $i \neq 1$ in Proposition 2 is necessary since there is no slope κ_0 left of $\boldsymbol{r}_1$ in the rate-distortion hull.

Proof. Inserting (7) into the left expression of (18) gives

$$\Big(\kappa_\ell \cdot d(\boldsymbol{x}, \boldsymbol{r}'_{(i,j)}) + \delta_\ell\Big) / h(\boldsymbol{c}'_{(i,j)}) \geq 1, \tag{19}$$

where ℓ is the index of the slope and vertical intercept at $d(\boldsymbol{x}, \boldsymbol{r}'_{(i,j)})$. Since it holds that $d(\boldsymbol{x}, \boldsymbol{r}'_{(i,j)}) \leq d(\boldsymbol{x}, \underline{\boldsymbol{r}}_i)$ and due to the convexity of $\mathcal{L}(\cdot)$, we can assume without loss of generality that $\ell = i$. Then, inserting (9) into (19) and factorizing κ_i gives

$$\Big(\kappa_i \cdot \Big(d(\boldsymbol{x}, \boldsymbol{r}'_{(i,j)}) - d(\boldsymbol{x}, \boldsymbol{r}_i)\Big) + h(\boldsymbol{c}_i)\Big) / h(\boldsymbol{c}'_{(i,j)}) \geq 1. \tag{20}$$

Finally, after inserting (17) into (20), the resulting expression can easily be rearranged into the right side of (18). □

The main point of Proposition 2 is that $\hat{\rho}(\boldsymbol{x}, \boldsymbol{c}'_{(i,j)}, \boldsymbol{r}'_{(i,j)}, \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$ can be easily computed from the available clusterings. A visual intuition of how $\hat{\rho}(\cdot)$ is computed can be seen in Fig. 4. A concrete algorithm is described in Section 4.2. Computational speedups implied by (14) and (18) are discussed in Section 4.3.

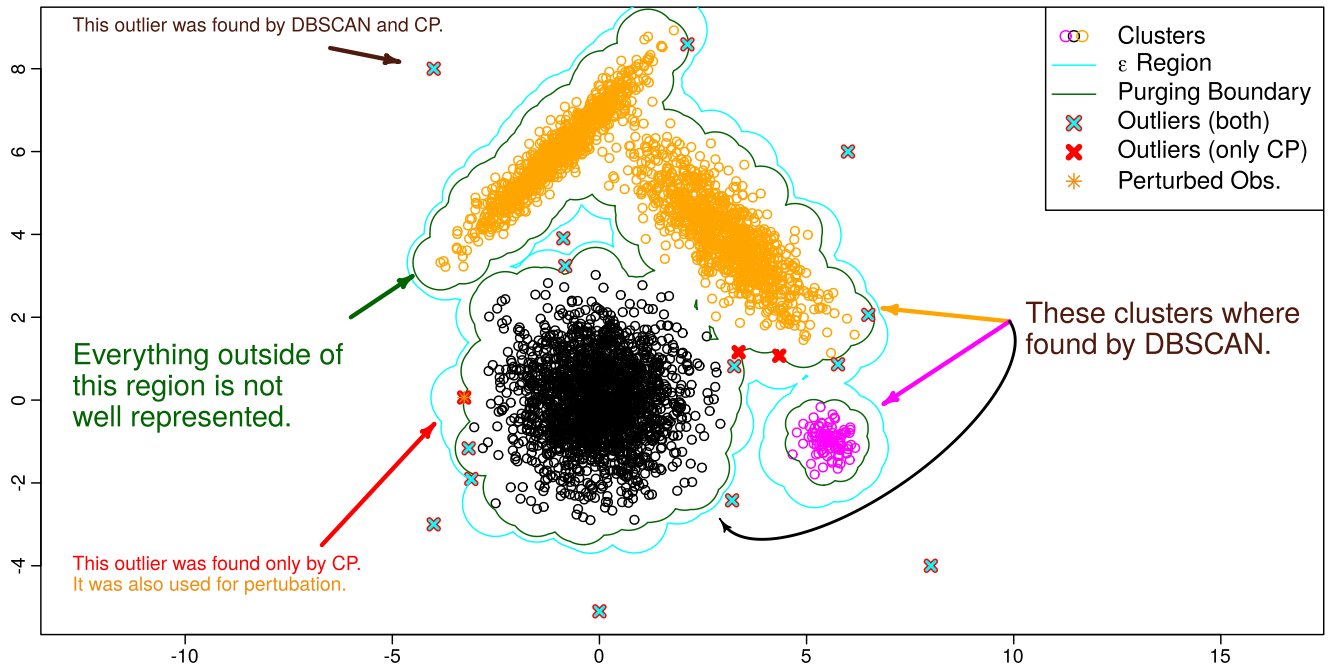

Fig. 5. Cluster Purging (CP) based on DBSCAN with $\varepsilon = 0.8$, minPts $= 20$ and a max-max perturbation (cf. Section 5.1). The parametrization of DBSCAN is suboptimal, and the clustering representation can be improved by purging (i.e., uniquely encoding) outliers detected by CP. Overlapping purging boundaries were depicted as union of discs for readability. Note that observations within the ε region and purging boundary may also be outliers if they are alone in their cluster (cluster size = 1).

4 Practical Aspects

After formalizing the theoretical background needed to efficiently perform Cluster Purging, we now address several practical issues and formulate concrete algorithms for an efficient computation.

4.1 Interpretation

Recall that any clustering is a representation of the raw data, and that a cluster is a representation of the data assigned to it. In essence, the theoretical foundation of Cluster Purging concerns itself with the representivity of clusterings. If a cluster would represent its data better if one of them were removed (purged), then that deviating observation is considered an outlier. To make the concept of representivity more tangible, we address four critical questions that may be non-obvious to the reader.

4.1.1 How can Rate-Distortion Outliers be Interpreted?

In simple terms, a rate-distortion outlier is an observation that is "far away" from its cluster. How "far" this needs to be is determined by a threshold that we call *purging boundary*. This purging boundary is inferred from cluster sizes and distortions across multiple clusterings, as well as from the raw dataset (see Eq. (18)). Hence, an accurate interpretation of rate-distortion outliers depends on how these quantities are measured. For example, under Manhattan distances and a k-means clustering, all purging boundaries are hypercubes that are centered at the cluster's centroid and enclose inliers. For DBSCAN and Euclidean distance, every observation within a specific cluster is surrounded by a hypersphere that encloses its nearest neighbor unless it is an outlier. See Fig. 5 for a visualization.

In the context of high-dimensional data, interpretability is often addressed via dimensionality reductions such that every outlier can be described by a small subset of the original dimensions, see [39], [40]. Similarly, rate-distortion outliers can be characterized by their low-entropy representation: They are observations that make the representation unnecessarily complicated.

4.1.2 How is Cluster Purging Different From Distance-Based Outlier Detection With Clustering?

Cluster Purging permits setups, e.g., centroid-based clustering and euclidean distortion, that are very similar to conventional distance-based outlier detection methods such as [20], [27]. The main difference between Cluster Purging and such methods is that purging boundaries are inferred based on a different clustering, and not based on a parameter. Further, Cluster Purging is not limited to distance-based setups and is compatible with any well-defined dissimilarity measure and clustering technique, e.g., Kullback-Leibler divergence [41] paired with fuzzy C-means clustering [42].

4.1.3 Isn't Cluster Purging Just Another Clustering-Based Outlier Detection Technique That Fails if the Clustering is Bad?

Not necessarily. Cluster Purging considers the original raw data via (18) in addition to all available clusterings. Further, the rate-distortion hull (7) allows one to determine which clusterings among the available ones are best in terms of rate-distortion theory. If all available clusterings are "bad", then Cluster Purging may fail to find correct outliers, yet if a single "good" clustering is available, then Cluster Purging will identify this clustering and use it for outlier detection.

4.1.4 Can Outliers Really be Detected via Representivity? It Seems Strange That Whether Data are Outliers Depends on the Size of Their Cluster

We describe a short example where rate-distortion theory-based representivity is intuitive for outlier detection: A group of 100 people is asked to form small "parties" to represent their political opinions. 95 people consider themselves *moderate* and form a moderate party, whereas 4 people form an *extremist* party and 1 person has no opinion. If this 1 person joined the small extremist party (cluster A), then this would have a more noticeable (outlying) effect on this party's political orientation than if the 1 person joined the large moderate party (cluster B). Likewise, purging boundaries grow logarithmically as clusters become larger (see Eq. (14)).

4.2 Algorithms for Cluster Purging

4.2.1 Parameter-Free Cluster Purging

From the theoretical formulations in Section 3, one can directly derive an algorithm for Cluster Purging. This algorithm takes a dataset $\boldsymbol{x}$ and a set of clusterings $(\underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$ as input and returns a set of outliers without requiring any additional parameters. In simple terms, this algorithm can be summarized as

1) Compute the entropy and distortion of all clusterings.
2) Find the lower convex hull of the resulting entropy-distortion pairs to construct a rate-distortion hull.
3) For every cluster in every clustering on this rate-distortion hull, compute how the entropy would change if an observation in this cluster were removed.
4) Based on the resulting changes of entropy and the slope of the rate-distortion hull, compute how much the distortion must change to pass the "purging boundary".
5) Data that, when purged, would be outside of the purging boundary, as well as clusters of size 1, are outliers.

A visual intuition of how this computation is performed is depicted in Figs. 4 and 5, whereas pseudo-code for this algorithm is listed in Algorithm 1. An R implementation can be found online.[2]

Note that the selected distortion measure $d(\cdot)$ should be equal to the distortion measure that was used to compute clusterings, e.g., for k-means clustering $d(\cdot)$ should be Euclidean distance, for DBSCAN it should be nearest neighbor distance. We confirmed this insight in preliminary experiments, where it turned out that heterogeneous distortion pairs were inferior to homogeneous distortion pairs in all settings we tested.

4.2.2 Parametric Cluster Purging

In some settings, it may be desirable to tune cluster purging to a specific dataset. While the parameter-free nature of the theoretical formulation of Cluster Purging prevents this, one can "cheat" by replacing the estimate of cluster representivity $\hat{\rho}(\cdot)$ with its true value $\rho(\cdot)$. Of course, $\rho(\cdot)$ is not known, yet in supervised settings it can be learned from a training set, or a user may simply guess its value or use a default parametrization.

2. https://tinyurl.com/f59ezjhk

Algorithm 1. Parameter-Free Cluster Purging

Require: $\boldsymbol{x}, \underline{\boldsymbol{c}}, \underline{\boldsymbol{r}}$
1: outliers $\leftarrow \emptyset$;
2: **for** clustering $(\boldsymbol{c}, \boldsymbol{r}) \in (\underline{\boldsymbol{c}}, \underline{\boldsymbol{r}})$ **do**
3: Compute $h(\boldsymbol{c})$ according to (3);
4: Compute $d(\boldsymbol{x}, \boldsymbol{r})$ according to (1);
5: **end for**
6: Set $\mathcal{L}$ to the lower convex hull of all h and d;
7: Compute κ (the slopes of $\mathcal{L}$) via linear interpolation;
8: Drop clusterings that are not on $\mathcal{L}$;
9: Sort clusterings increasingly according to $d(\boldsymbol{x}, \boldsymbol{r})$;
10: Drop clustering with highest entropy (cf. Proposition 2);
11: **for all** $(c, \boldsymbol{r})$ **do**
12: **for** cluster $g \in (\boldsymbol{c}, \boldsymbol{r})$ **do**
13: Compute change of entropy according to (14);
14: **end for**
15: **end for**
16: **for** $j \in (1, \ldots, n)$ **do**
17: **if** any side of (18) holds for all $(\boldsymbol{c}, \boldsymbol{r})$ **then**
18: outliers $\leftarrow$ outliers $\cup\, x_j$;
19: **end if**
20: **end for**
21: **return** outliers

In particular, the concrete value of $\rho(\cdot)$ at a specific clustering $(\boldsymbol{c}, \boldsymbol{r})$ is not even needed. According to (18), it is sufficient if slope κ of the rate-distortion function at $d(\boldsymbol{x}, \boldsymbol{r})$ is passed as parameter, since the remaining quantities needed to perform Cluster Purging can be easily inferred from κ. A concrete algorithm is listed in Algorithm 2.

Algorithm 2. Parametric Cluster Purging

Require: $\boldsymbol{x}, \boldsymbol{c}, \boldsymbol{r}, \kappa$
1: outliers $\leftarrow \emptyset$;
2: **for** cluster $g \in (\boldsymbol{c}, \boldsymbol{r})$ **do**
3: Compute change of entropy Δ_g according to (14);
4: **end for**
5: **for** $j \in (1, \ldots, n)$ **do**
6: **if** $d(x_j, r_{c_j}) \cdot \kappa \leq \Delta_{c_j}$ **then**
7: outliers $\leftarrow$ outliers $\cup\, x_j$;
8: **end if**
9: **end for**
10: **return** outliers

A clear advantage of this parametric variant of Cluster Purging is that, if the true slope is passed to the algorithm, it will necessarily be superior to the parameter-free variant. Further, this variant only needs a single clustering, and is very simple overall. However, we believe that the parameter-free algorithm should generally be preferred over its parametric counterpart (cf. [43]).

4.3 Efficiency

In the pseudo-code of Algorithms 1 and 2 there are several verbose instructions whose computational complexity might be non-obvious. In Algorithm 1, lines 3 and 4 require $\mathcal{O}(n)$ steps, whereas all remaining verbose steps in both

algorithms require at most $O(vtd)$ steps. Asymptotically, v is the largest number of clusters, t the number of clusterings, and d the dimensionality of the dataset. Since all three of these quantities were assumed to be constant, these steps can hence be performed in $\mathcal{O}(1)$ time. Consequently, the time complexity of both Algorithms can be reduced to $\mathcal{O}(n)$.

In terms of space complexity, one will naturally require at least $\mathcal{O}(tn)$ space to store all clusterings. The remaining memory overhead of both algorithms is constant.

4.4 Obtaining Multiple Clusterings $(\underline{c}, \underline{r})$

In recent years, datasets have become increasingly large and "*in many situations, the knowledge extraction process has to be very efficient and close to real time because storing all observed data is nearly infeasible*" [44]. Consequently, it may occur in practice that computing multiple good clusterings of a dataset may be too costly, although the above formulation of rate-distortion hulls would require this. To address this issue, we here discuss methods for efficiently obtaining similar clusterings, i.e., perturbations, from a single "seed" clustering.

In general the theoretical formulations of Cluster Purging permit arbitrary perturbations. However, the quality of a clustering representivity estimate depends on how "strongly" the seed clustering was perturbed. Hence, from a rate-distortion theoretic perspective, it is desirable that clustering $(\boldsymbol{c}, \boldsymbol{r})$ and its perturbation $(\tilde{\boldsymbol{c}}, \tilde{\boldsymbol{r}})$ are as similar as possible, yet not identical. To achieve this, it is typically sufficient to modify the cluster assignment and representation of a single observation x_j, given that this change results in a different entropy-distortion pair, i.e., $[h(\boldsymbol{c}), d(\boldsymbol{x}, \boldsymbol{r})] \neq [h(\tilde{\boldsymbol{c}}), d(\boldsymbol{x}, \tilde{\boldsymbol{r}})]$. A concrete change that causes this is typically given by selecting the cluster with the largest size, i.e., $\operatorname{argmax} \boldsymbol{f}^{\boldsymbol{c}}$, and removing the observation that causes the largest distortion in this cluster. At first glance, this may seem counterintuitive, since the aim of a perturbation is to cause a small yet sufficiently large change in the clustering, and hence removing the observation from the smallest cluster with the smallest distortion would seem better. We elaborate on this and empirically compare other perturbation strategies in Section 5.1.

4.5 Nearest Neighbor Representations

A further issue may occur when the selected clustering technique, e.g., DBSCAN, yields cluster assignments $\boldsymbol{c}$ yet no representations $\boldsymbol{r}$. In such cases, one can jointly infer $\boldsymbol{r}$ from $\boldsymbol{x}$ and $\boldsymbol{c}$ based on the following intuition: Since clustering techniques group data according to some similarity measure [45], this similarity measure implicitly contains information on what a representation for such a clustering technique might be. In the case of DBSCAN, which clusters data according to nearest neighbor distances, one can simply represent every x_j by its nearest neighbor within the cluster of x_j. While using such representations leads to no compression of the data, this is still meaningful if one wants to detect outliers. We demonstrate this empirically in Section 5.2, whereas a visualization can be seen in Fig. 6.

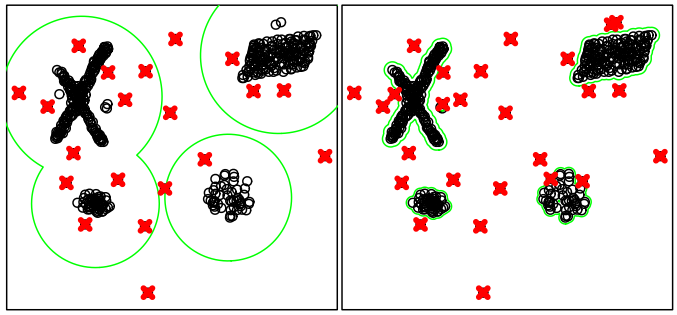

Fig. 6. Cluster Purging applied to a synthetic dataset [38] clustered with DBSCAN. Detected outliers are depicted in red ($\times$). *Left*: For every cluster, a single euclidean centroid was used as representative, resulting in large, spherical purging boundaries. *Right*: For every observation, its nearest neighbor within the same cluster was used as representative, resulting in tight boundaries that fit the data well.

4.6 Rules of Thumb

Since Cluster Purging allows highly diverse setups, we formulate three rules of thumb for guiding practitioners:

First, different clusterings offer different entropy-distortion trade-offs, e.g., a clustering with n clusters leads to a lossless representation yet no compression, whereas a representation with a single cluster leads to good compression yet large distortion. Since purging boundaries depend on cluster sizes, they will adapt to different entropy-distortion trade-offs. Generally speaking, Cluster Purging will work well under many different trade-offs as long as one avoids the extremes of the empirical rate-distortion function.

Second, it is desirable that the selected clusterings and/or perturbations have similar entropy-distortion trade-offs. The reason for this is that the estimated rate-distortion slope between two clusterings becomes less accurate the further these clusterings are apart in rate-distortion space. Hence, it is generally not a good idea to combine different clustering techniques, e.g., k-means and DBSCAN. Pairing similar clusterings is usually better, e.g., 7-means with 8-means. Fixing a single clustering $(\boldsymbol{c}, \boldsymbol{r})$ and computing a slight perturbation $(\tilde{\boldsymbol{c}}, \tilde{\boldsymbol{r}})$ by changing the cluster assignment of a single observation is likely best.

Third, the selected distortion measure should be related to the selected clustering technique. For instance, it is often better to pair k-means with euclidean distortion than with Hamming distortion, and for hierarchical clusterings one should use the same distance function for computing the clustering and for measuring distortion. For probabilistic clustering techniques, distortion should likely be measured via Kullback-Leibler divergence.

4.7 Limitations

The concept of rate-distortion outliers describes *individual* observations that are outlying. Collective outliers [1] and outlying clusters are not covered and will be addressed in future work. Further, in rare cases it may occur that the computed rate-distortion hull has an increasing segment. In such an increasing region (18) does not hold, and it is best to ignore this region of the rate-distortion hull. Finally, while Algorithms 1 and 2 can be computed in $\mathcal{O}(n)$ time, the computation of the clusterings they are based on may be more costly.

TABLE 1
Case Study: Average Class-Wise F_1-Scores

Measure	Perturbation Strategy			
	min-min	min-max	max-min	max-max
Outlier F_1-score	**0.17**	0.08	0.00	0.16
Inlier F_1-score	0.43	0.94	0.16	**0.97**
Combined F_1-score	0.30	0.51	0.08	**0.56**

5 Experimental Evaluation

To evaluate the practical applicability and correctness of rate-distortion theory for outlier detection, we conduct a case study in which different perturbation strategies are analyzed (Section 5.1). In Section 5.2, we compare our method Cluster Purging (CP) with other state-of-the-art outlier detection methods in an experimental evaluation on benchmark datasets. Further, we also analyze how frequently Cluster Purging improves upon outliers detected by an existing clustering. Throughout all experiments, we use Euclidean distance as distance measure in all clustering techniques, and consequently also as distortion measure. We avoid using non-distance distortion measures such as Kullback-Leibler divergence, since this would make a fair comparison of Cluster Purging with distance-based outlier detectors difficult. Centroids are computed as the arithmetic mean of all observations in a cluster whenever needed. The *source code* for reproducing all results, as well as all data can be accessed online.[3]

5.1 Case Study: Perturbation for Map Denoising

From the elaborations made in Section 4.4, one can derive four different perturbation strategies[4]

1) min-min: Select smallest cluster, purge least distorted observation.
2) min-max: Select smallest cluster, purge most distorted observation.
3) max-min: Select largest cluster, purge least distorted observation.
4) max-max: Select largest cluster, purge most distorted observation.

We compare all four strategies in a case study, where the goal is to denoise a dataset via k-means clustering and outlier detection. The dataset contains coordinates of a map of the continent Europe [46] with 100 artificially added noise points. Since k-means clustering algorithms are sensitive to the selected initial centers, we fix the number of centroids to $k = 225$, and compute 1000 different initializations, each for 10 different initial random seeds. For every computed clustering, we perform Cluster Purging based on all 4 perturbation strategies with noise points considered as outliers. As evaluation measure, we use $F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$. Further, since inlier and outlier classes are heavily imbalanced $(169673 : 100)$ we compute average class-wise F_1-scores in addition to average raw F_1-scores. The results of this case study are reported in Table 1, whereas a visualization can be seen in Fig. 7.

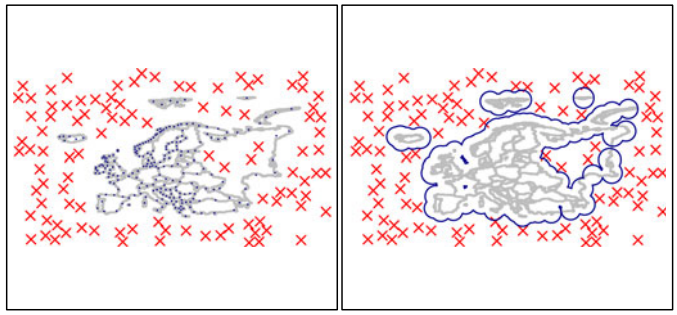

Fig. 7. Case Study: Comparison of Purging Boundaries (blue) with min-min perturbation and max-max perturbation. True noise points are depicted in red ($\times$), while detected outliers are not depicted for readability (the left plot would be covered in outliers). *Left*: Purging boundaries derived from a min-min perturbation are so small that they are barely visible. *Right*: Purging boundaries derived from a max-max perturbation are ≈ 67 times larger than min-min purging boundaries, almost fully covering the map of Europe.

5.2 Competitive Evaluation on Benchmark Datasets

5.2.1 Setup

We compare both variants of our method, Cluster Purging (CP) and Parametric Cluster Purging (CPP) against closely related outlier detection methods mentioned in Section 2:

- The one-class rate-distortion model (OCRD) [32].
- Variants of k-means that detect outliers, i.e., k-means$--$ (KM—) [15] and k-means with outlier removal (KMOR) [16].
- Raw clusterings, i.e., k-means clustering, Hierarchical Agglomerative Clustering (HAC) with complete linkage and DBSCAN [14], with singelton clusters considered as outliers (these variants are referred to as *Vanilla* detectors)
- Cluster-based local outlier factor (CBLOF) [19] based on all vanilla clusterings and raw local outlier factor (LOF) [28].
- Outlier detection for high-dimensional data via Local Projection Score (LPS) [47].
- Cluster Purging (CP) with a single max-max perturbation and Parametric Cluster Purging (CPP), both based on all vanilla clustering techniques ($t = 1$ clustering each). Other perturbation methods are addressed in Section 5.2.4.

We omit [25], [26] since they use soft clusterings; [20] and [27] because they have high computational cost and are not reproducible, respectively; [29], [30], [31] since we found that two variants of the Local Outlier Factor are sufficient. To enable a comparison with LOF, CBLOF and LPS, which return outlier scores instead of outliers indices, we take the top $m = |\boldsymbol{y}|$ scores of these methods, where m is the true number of outliers in dataset $\boldsymbol{x}$. As evaluation measure, we use F_1-score. Further, since all clustering algorithms under consideration (and most outlier detectors) have parameters, it is difficult to generalize outlier detection performances based on a single arbitrarily selected parametrization. Hence, the parameters of all clustering

3. https://tinyurl.com/f59ezjhk
4. In all four perturbation strategy descriptions, "purge" is short for "reassign to additional unique cluster".

TABLE 2
Compared Outlier Detection Methods and Their Parametrizations

Methods	Grid searched parameters	Hard coded parameters
OCRD	β: $[0.1, \ldots, 10]$ (n steps)	$q(0) = 0.5$ uniform prior
	k-means	
Vanilla	$k = [2, \ldots, 10]$	n_{start}=1000
KM--	$k = [2, \ldots, 10]$	$n_{\text{outlier}} = m$
KMOR	$k = [2, \ldots, 10]$, $\gamma = [0.1, \ldots, 10]$ (n steps)	$\delta = 1$
CBLOF	(vanilla parameters)	$b = \min(k$ - $1, 5)$
CP	(vanilla parameters)	
CPP	(vanilla parameters), $\kappa = [0.1, \ldots, 10]$ (n steps)	
	HAC	
Vanilla	$k = [1, \ldots, n]$	
CBLOF	(vanilla parameters)	$b = \min(k$ - $1, 5)$
CP	(vanilla parameters)	
CPP	(vanilla parameters), $\kappa = [0.1, \ldots, 10]$ (n steps)	
	DBSCAN	
Vanilla	$\min_p$=$[d + 1, \ldots, d + 10]$ ε = unique $\min_p$-NN dists.	
CBLOF	(vanilla parameters)	$b = \min(k$ - $1, 5)$
CP	(vanilla parameters)	
CPP	(vanilla parameters), $\kappa = [0.1, \ldots, 10]$ (n steps)	
	No Clustering	
LOF	$k = [1, \ldots, n - 1]$	
LPS	$k = [2, \ldots, \lceil \frac{d}{2} \rceil]$	$n_{\text{outlier}} = m$

techniques (and outlier detection methods) are grid searched over their respective parameter space towards maximizing F_1- score. For methods having several parameters where a grid search would be infeasible, some parameters are set according to literature recommendations. The detailed grid search setups and parametrizations are listed in Table 2.

Additionally, to evaluate the claimed computational efficiency of CP and CPP, we track the average runtime of each method per call. We report this quantity instead of overall runtime since the total number of needed calls to each outlier detection method varies for each grid search.

5.2.2 Datasets

The experimental evaluation of all detectors is performed on 13 publicly available benchmark datasets, taken from [48]. These datasets come from diverse domains such as medicine, space, and telecommunications, and were commonly used as benchmarks in literature. More detailed descriptions of the domain background of these datasets can be found in [48]. For this experimental evaluation, dataset Arrhymthia is particularly noteworthy since it is high-dimensional with $n \approx d$, and Heart, Pima and Ionosphere since they have an outlier ratio $\frac{m}{n}$ close to 50 percent.

5.2.3 Main Results

The main results of the competitive evaluation are depicted in Table 3. Overall, detectors based on k-means clusterings performed worse than detectors based on other clusterings. The overall highest average F_1-score was achieved by CBLOF based on HAC clustering. For other clustering methods, CPP performed best. The average performance of OCRD, which is bound to a Blahut-Arimoto-like clustering, was competitive with detectors based on k-means clusterings, yet lower than that of detectors based on HAC and DBSCAN.

Regarding computational efficiency, vanilla clusterings were faster than methods based on these clusterings. The fastest method was vanilla k-means, while CPP had the overall lowest surplus runtime after its clustering was computed. The slowest method was LOF followed by LPS.

When considering on how many datasets detectors with exchangeable clusterings did not perform worse than the respective vanilla clustering, there is a clear ranking. Our method CPP performed best (100 percent), followed by CP (85 percent), followed by CBLOF (62 percent).

5.2.4 Detailed Results per Perturbation Method

In the bottom of Table 3, average F_1-scores and runtimes of all four considered perturbation strategies are listed per clustering. In terms of average F_1-scores, the max-max perturbation scored highest most often, whereas differences in runtime between perturbation strategies are negligible. For this reason and due to lack of space, only the detailed scores per dataset of CP with max-max perturbations are listed in Table 3.

6 Discussion

The results of the case study indicate that the max-max perturbation is slightly superior over the other considered perturbation strategies. This is in accordance with the results of the competitive evaluation, and hence we overall argue that max-max perturbations should be preferred.

In the benchmark evaluation, the parameter-free variant of Cluster Purging seems to be competitive with other detectors, yet does not demonstrate superior detection performances. However, this lack of superiority may be tolerable when one considers that a parameter-free algorithm was compared against parametric ones—where CBLOF, the strongest competitor, received information on how many outliers are present in the dataset. Of course, one may argue that Cluster Purging is not truly parameter-free if only a single clustering is provided, since the selected perturbation strategy can also be seen as a parameter. Yet, when one considers that multiple different perturbation strategies may lead to similar detection results (cf. Table 2 min-max and max-max), then it can be argued that Cluster Purging is still "less" parameter-dependent than other competing methods. Further, if a single parameter is allowed (rate-distortion hull slope κ), then one can use the parametric variant of Cluster Purging, which overall seems to compete strongly against the state-of-the-art. The slow runtime of the seemingly efficient method LOF can be explained by the need of computing up to $n - 1$ nearest neighbors during parameter optimization.

It is also noteworthy that Cluster Purging—especially its parametric variant—performed (or was tied for) best on high-dimensional and outlier heavy datasets Arrhymthia, Heart, Pima and Ionosphere. Hence, one can expect Cluster Purging to tolerate high-dimensional data or high outlier ratios even if clustering such data is challenging.

TABLE 3
Competitive Evaluation Results

Clustering	B-A	k-means						HAC				DBSCAN				None	
Detector	OCRD	Vanilla	KM—	KMOR	CBLOF	CP	CPP	Vanilla	CBLOF	CP	CPP	Vanilla	CBLOF	CP	CPP	LOF	LPS
Adapts #outlier?	✓	✓	×	✓	×	✓	✓	✓	×	✓	✓	✓	×	✓	✓	×	×
Parameter-free?	×	×	×	×	×	✓	×	×	×	✓	×	×	×	✓	×	×	×
								F_1-score									
Arrhymthia	0.68	0.01	0.67	0.63	0.63	0.20	0.69	0.68	0.67	0.70	0.71	0.62	0.66	0.62	0.69	0.69	0.60
Heart	0.65	0.00	0.57	0.62	0.53	0.16	0.63	0.64	0.56	0.64	0.65	0.63	0.54	0.63	0.67	0.55	0.48
Hepatitis	0.43	0.00	0.23	0.41	0.31	0.24	0.34	0.31	0.31	0.32	0.36	0.35	0.31	0.35	0.35	0.31	0.23
Parkinson	0.86	0.00	0.80	0.86	0.78	0.12	0.79	0.86	0.86	0.86	0.86	0.81	0.82	0.81	0.86	0.78	0.73
Pima	0.60	0.00	0.49	0.56	0.47	0.20	0.55	0.52	0.50	0.52	0.56	0.54	0.47	0.53	0.54	0.54	0.43
Stamps	0.59	0.00	0.29	0.51	0.45	0.16	0.38	0.24	0.94	0.33	0.52	0.64	0.42	0.65	0.65	0.39	0.65
Glass	0.18	0.00	0.11	0.24	0.44	0.24	0.34	0.32	0.22	0.33	0.36	0.33	0.44	0.33	0.33	0.33	0.11
Ionosphere	0.69	0.00	0.82	0.77	0.67	0.51	0.80	0.86	0.75	0.84	0.87	0.77	0.85	0.77	0.88	0.83	0.67
Lympho	0.86	0.00	0.33	0.40	0.17	0.67	0.80	0.67	0.33	0.83	0.83	0.29	0.67	0.55	0.62	0.83	0.33
Shuttle	0.32	0.00	0.23	0.21	0.15	0.11	0.20	0.21	0.85	0.21	0.27	0.32	0.15	0.32	0.34	0.31	0.31
WBC	0.70	0.00	0.70	0.78	0.60	0.74	0.78	0.53	1.00	0.64	0.78	0.82	0.50	0.82	0.82	0.80	0.60
WDBC	0.67	0.00	0.80	0.84	0.80	0.80	0.84	0.84	0.90	0.78	0.90	0.84	0.90	0.90	0.90	0.80	0.70
WPBC	0.40	0.00	0.23	0.40	0.34	0.19	0.41	0.39	0.43	0.41	0.42	0.44	0.38	0.44	0.44	0.36	0.28
Average	0.59	0.00	0.48	0.56	0.49	0.33	**0.58**	0.54	**0.64**	0.57	0.62	0.57	0.55	0.59	**0.62**	0.58	0.47
								Average runtime per method call (milliseconds)									
Arrhymthia	25.48	6.04	274.02	109.27	6.25	6.25	6.27	2.25	28.64	22.11	11.91	97.20	173.57	263.24	109.76	543.70	4065.01
Heart	9.39	0.21	78.56	22.09	0.21	0.21	0.24	0.86	10.88	5.85	2.77	1.06	17.57	23.08	1.65	278.49	29.91
Hepatitis	2.86	0.09	18.03	5.21	0.10	0.09	0.11	0.35	3.48	3.15	1.12	0.40	5.70	7.54	0.65	136.42	11.62
Parkinson	9.97	0.15	29.96	16.91	0.16	0.16	0.19	0.62	8.53	5.43	2.28	0.85	15.26	17.67	1.48	222.19	26.45
Pima	41.37	0.47	405.15	97.74	0.49	0.49	0.56	5.12	33.45	16.09	7.50	2.24	62.45	62.80	4.34	969.16	76.47
Stamps	22.08	0.31	146.76	31.91	0.32	0.32	0.35	1.31	13.58	7.21	3.28	1.86	26.04	29.72	2.04	349.49	34.01
Glass	13.47	0.15	65.98	18.01	0.15	0.15	0.18	0.68	7.96	5.29	2.05	0.80	16.07	18.33	1.32	238.27	24.91
Ionosphere	21.40	0.62	101.73	43.48	0.63	0.63	0.67	1.45	14.15	8.87	4.10	6.81	44.66	44.89	3.00	358.27	123.49
Lympho	5.85	0.10	77.28	11.33	0.10	0.10	0.13	0.38	6.62	3.85	1.75	0.41	7.55	12.89	1.07	172.90	17.45
Shuttle	54.22	0.98	56.49	138.08	1.02	1.02	1.11	9.74	46.86	21.30	10.45	10.19	111.05	112.32	6.25	1447.65	95.85
WBC	11.92	0.18	81.76	20.14	0.19	0.19	0.21	0.70	8.37	5.60	2.23	1.38	16.19	22.96	1.42	250.97	25.09
WDBC	21.38	0.52	140.29	40.72	0.53	0.53	0.58	1.53	16.75	9.67	4.42	2.35	36.57	36.50	2.89	375.74	120.32
WPBC	8.82	0.38	71.80	20.27	0.38	0.38	0.41	0.71	9.19	5.58	2.57	3.00	16.15	23.31	1.75	223.21	33.84
Total average	19.09	**0.79**	119.06	44.24	0.81	0.81	0.85	**1.98**	16.04	9.23	4.34	**9.89**	42.22	51.94	10.59	428.19	360.34

Perturbation Specific Results

	k-means				HAC				DBSCAN			
	min-min	min-max	max-min	max-max	min-min	min-max	max-min	max-max	min-min	min-max	max-min	max-max
Average F_1	**0.45**	0.11	0.33	0.33	0.55	0.54	0.35	**0.57**	0.33	**0.59**	0.29	**0.59**
Average Runtime	0.81	0.81	0.81	0.81	9.22	9.24	9.23	9.23	51.89	51.90	51.64	51.94

Consequently, we expect Cluster Purging to perform well in a variety of domains under the premise that a reasonably-working clustering technique is known. Further, our proposed algorithms, especially the parametric variant, are efficient in terms of computational complexity, requiring only $\mathcal{O}(n)$ time. While at least one clustering is still required as input, this efficiency can be a key advance in scenarios where prior clusterings of the data are available.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable feedback. This work was supported in part by the ECSEL Joint Undertaking (JU) through iDev40 Project under Grant 783163, in part by the European Union's Horizon 2020 Research and Innovation Programme, and in part by the consortium members, Austria, Germany, Belgium, Italy, Spain, and Romania. The Know-Center is funded within the Austrian COMET Program - Competence Centers for Excellent Technologies - under the auspices of the Austrian Federal Ministry for Climate Action, Environment, Energy, Mobility, Innovation and Technology, the Austrian Federal Ministry of Digital and Economic Affairs, and by the State of Styria. COMET is managed by the Austrian Research Promotion Agency FFG.

References

[1] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv. (CSUR)*, vol. 41, no. 3, 2009, Art. no. 15.

[2] A. Zimek and P. Filzmoser, "There and back again: Outlier detection between statistical reasoning and data mining algorithms," *Wiley Interdisciplinary Rev.: Data Mining Knowl. Discov.*, vol. 8, no. 6, 2018, Art. no. e1280.
[3] N. Görnitz, M. Kloft, K. Rieck, and U. Brefeld, "Toward supervised anomaly detection," *J. Artif. Intell. Res.*, vol. 46, pp. 235–262, 2013.
[4] G. Pang, C. Shen, and A. van den Hengel, "Deep anomaly detection with deviation networks," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2019, pp. 353–362.
[5] J. Chen, S. Sathe, C. Aggarwal, and D. Turaga, "Outlier detection with autoencoder ensembles," in *Proc. SIAM Int. Conf. Data Mining*, 2017, pp. 90–98.
[6] H. Zenati, M. Romain, C.-S. Foo, B. Lecouat, and V. Chandrasekhar, "Adversarially learned anomaly detection," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, 2018, pp. 727–736.
[7] L. Ruff *et al.*, "Deep one-class classification," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 4393–4402.
[8] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[9] D. Kwon, H. Kim, J. Kim, S. C. Suh, I. Kim, and K. J. Kim, "A survey of deep learning-based network anomaly detection," *Cluster Comput.*, vol. 22, pp. 949–961, 2019.
[10] R. Chalapathy and S. Chawla, "Deep learning for anomaly detection: A survey," 2019, *arXiv: 1901.03407*.
[11] G. Marcus, "Deep learning: A critical appraisal," 2018, *arXiv: 1801.00631*.
[12] C. Rudin, "Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead," *Nat. Mach. Intell.*, vol. 1, no. 5, pp. 206–215, 2019.
[13] C. Rudin and B. Ustun, "Optimized scoring systems: toward trust in machine learning for healthcare and criminal justice," *Interfaces*, vol. 48, no. 5, pp. 449–466, 2018.
[14] M. Ester *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proc. KDD*, vol. 96, no. 34, 1996, pp. 226–231.
[15] S. Chawla and A. Gionis, "K-means–: A unified approach to clustering and outlier detection," in *Proc. SIAM Int. Conf. Data Mining*, 2013, pp. 189–197.
[16] G. Gan and M. K.-P. Ng, "K-means clustering with outlier removal," *Pattern Recognit. Lett.*, vol. 90, pp. 8–14, 2017.
[17] H. Liu, J. Li, Y. Wu, and Y. Fu, "Clustering with outlier removal," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 6, pp. 2369–2379, Jun. 2021.
[18] Z. He, X. Xu, and S. Deng, "Discovering cluster-based local outliers," *Pattern Recognit. Lett.*, vol. 24, no. 9/10, pp. 1641–1650, 2003.
[19] S.-y. Jiang and Q.-b. An, "Clustering-based outlier detection method," in *Proc. 5th Int. Conf. Fuzzy Syst. Knowl. Discov.*, 2008, pp. 429–433.
[20] R. Pamula, J. K. Deka, and S. Nandi, "An outlier detection method based on clustering," in *Proc. 2nd Int. Conf. Emerg. Appl. Inf. Technol.*, 2011, pp. 253–256.
[21] H.-P. Kriegel, P. Kröger, and A. Zimek, "Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering," *ACM Trans. Knowl. Discov. (TKDD)*, vol. 3, no. 1, pp. 1–58, 2009.
[22] I. S. Dhillon, S. Mallela, and D. S. Modha, "Information-theoretic co-clustering," in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2003, pp. 89–98.
[23] R. Blahut, "Computation of channel capacity and rate-distortion functions," *IEEE Trans. Inf. Theory*, vol. 18, no. 4, pp. 460–473, Jul. 1972.
[24] S. Arimoto, "An algorithm for computing the capacity of arbitrary discrete memoryless channels," *IEEE Trans. Inf. Theory*, vol. 18, no. 1, pp. 14–20, Jan. 1972.
[25] C. Ruiz, M. Spiliopoulou, and E. Menasalvas, "C-DBSCAN: Density-based clustering with constraints," in *Proc. Int. Workshop Rough Sets, Fuzzy Sets, Data Mining, Granular-Soft Comput.*, 2007, pp. 216–223.
[26] A. Smiti and Z. Eloudi, "Soft DBSCAN: Improving DBSCAN clustering method using fuzzy set theory," in *Proc. 6th Int. Conf. Hum. Syst. Interact. (HSI)*, 2013, pp. 380–385.
[27] Z. He, S. Deng, and X. Xu, "Outlier detection integrating semantic knowledge," in *Proc. Int. Conf. Web-Age Inf. Manage.*, 2002, pp. 126–131.
[28] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *ACM SIGMOD Rec.*, vol. 29, no. 2, pp. 93–104, 2000.
[29] S. Papadimitriou, H. Kitagawa, P. B. Gibbons, and C. Faloutsos, "LOCI: Fast outlier detection using the local correlation integral," in *Proc. 19th Int. Conf. Data Eng.*, 2003, pp. 315–326.
[30] L. Duan, L. Xu, F. Guo, J. Lee, and B. Yan, "A local-density based spatial clustering algorithm with noise," *Inf. Syst.*, vol. 32, no. 7, pp. 978–986, 2007.
[31] H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek, "Loop: Local outlier probabilities," in *Proc. 18th ACM Conf. Inf. Knowl. Manage.*, 2009, pp. 1649–1652.
[32] K. Crammer, P. P. Talukdar, and F. Pereira, "A rate-distortion one-class model and its applications to clustering," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 184–191.
[33] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. Somerset, U.K.: Wiley, 2006.
[34] B.-K. Yi and C. Faloutsos, "Fast time sequence indexing for arbitrary Lp norms," in *Proc. 26th Int. Conf. Very Large Data Bases*, 2000, pp. 385–394.
[35] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A symbolic representation of time series, with implications for streaming algorithms," in *Proc. 8th ACM SIGMOD Workshop Res. Issues Data Mining Knowl. Discov.*, 2003, pp. 2–11.
[36] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra, "Dimensionality reduction for fast similarity search in large time series databases," *Knowl. Inf. Syst.*, vol. 3, no. 3, pp. 263–286, 2001.
[37] F. E. Grubbs, "Procedures for detecting outlying observations in samples," *Technometrics*, vol. 11, no. 1, pp. 1–21, 1969.
[38] C. Böhm, K. Haegler, N. S. Müller, and C. Plant, "Coco: Coding cost for parameter-free outlier detection," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2009, pp. 149–158.
[39] X. H. Dang, I. Assent, R. T. Ng, A. Zimek, and E. Schubert, "Discriminative features for identifying and interpreting outliers," in *Proc. IEEE 30th Int. Conf. Data Eng.*, 2014, pp. 88–99.
[40] N. Liu, D. Shin, and X. Hu, "Contextual outlier interpretation," 2017, *arXiv:1711.10589*.
[41] S. Kullback and R. A. Leibler, "On information and sufficiency," *Ann. Math. Statist.*, vol. 22, no. 1, pp. 79–86, 1951.
[42] J. C. Dunn, "A fuzzy relative of the isodata process and its use in detecting compact well-separated clusters," *J. Cybern.*, vol. 3, pp. 32–57, 1973.
[43] E. Keogh, S. Lonardi, and C. A. Ratanamahatana, "Towards parameter-free data mining," in *Proc. 10th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2004, pp. 206–215.
[44] X. Wu, X. Zhu, G.-Q. Wu, and W. Ding, "Data mining with big data," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 1, pp. 97–107, Jan. 2014.
[45] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Comput. Surv. (CSUR)*, vol. 31, no. 3, pp. 264–323, 1999.
[46] P. Fränti and S. Sieranoja, "K-means properties on six clustering benchmark datasets," *Appl. Intell.*, vol. 48, no. 12, pp. 4743–4759, 2018.
[47] H. Liu, X. Li, J. Li, and S. Zhang, "Efficient outlier detection for high-dimensional data," *IEEE Trans. Syst., Man, Cybern.: Syst.*, vol. 48, no. 12, pp. 2451–2461, 2017.
[48] G. O. Campos *et al.*, "On the evaluation of unsupervised outlier detection: Measures, datasets, and an empirical study," *Data Mining Knowl. Discov.*, vol. 30, no. 4, pp. 891–927, 2016.

Maximilian B. Toller is currently working toward the PhD degree with the Graz University of Technology, Austria. He is currently a researcher with Know-Center GmbH, Graz, Austria. His research interests include outlier detection, time series data mining, theoretical foundations of data mining, and computational complexity theory.

Bernhard C. Geiger (Senior Member, IEEE) received the Dipl.-Ing. degree (with distinction) in electrical engineering and the Dr. techn. degree (with distinction) in electrical and information engineering from the Graz University of Technology, Austria, in 2009 and 2014, respectively. In 2009, he joined the Signal Processing and Speech Communication Laboratory, Graz University of Technology, as a project assistant, where he became a research and teaching associate in 2010. He was a senior scientist and an Erwin Schrödinger fellow with the Institute for Communications Engineering, Technical University of Munich, Germany, from 2014 to 2017 and a postdoctoral researcher with Signal Processing and Speech Communication Laboratory, Graz University of Technology, Austria, from 2017 to 2018. He is currently a senior researcher with Know-Center GmbH, Graz, Austria. His research interests include information theory for machine learning, theory-assisted machine learning, and information-theoretic model reduction for Markov chains and hidden Markov models.

Roman Kern is currently an assistant professor with the Institute for Interactive Systems and Data Science, Technical University of Graz and head of Knowledge Discovery at the Know-Center, a competence centre for Big Data analytics and data-driven business. His research interests include natural language processing, machine learning, with a focus on data science and big data analytics. He applies these methods in fields like scientific publication mining, intelligent transportation systems, and smart production.