RNN-DBSCAN: A Density-Based Clustering Algorithm Using Reverse Nearest Neighbor Density Estimates

Avory Bryant and Krzysztof Cios, *Member, IEEE*

Abstract—A new density-based clustering algorithm, *RNN-DBSCAN*, is presented which uses reverse nearest neighbor counts as an estimate of observation density. Clustering is performed using a *DBSCAN*-like approach based on k nearest neighbor graph traversals through dense observations. *RNN-DBSCAN* is preferable to the popular density-based clustering algorithm *DBSCAN* in two aspects. First, problem complexity is reduced to the use of a single parameter (choice of k nearest neighbors), and second, an improved ability for handling large variations in cluster density (heterogeneous density). The superiority of *RNN-DBSCAN* is demonstrated on several artificial and real-world datasets with respect to prior work on reverse nearest neighbor based clustering approaches (*RECORD*, *IS-DBSCAN*, and *ISB-DBSCAN*) along with *DBSCAN* and *OPTICS*. Each of these clustering approaches is described by a common graph-based interpretation wherein clusters of dense observations are defined as connected components, along with a discussion on their computational complexity. Heuristics for *RNN-DBSCAN* parameter selection are presented, and the effects of k on *RNN-DBSCAN* clusterings discussed. Additionally, with respect to scalability, an approximate version of *RNN-DBSCAN* is presented leveraging an existing approximate k nearest neighbor technique.

Index Terms—Unsupervised learning, pattern analysis, clustering algorithms, pattern clustering, density estimation robust algorithm, nearest neighbor searches

1 Introduction

Clustering is an unsupervised pattern recognition problem which has applications across multiple domains such as image analysis, remote sensing, bioinformatics, and text analysis. The clustering problem can be defined generically as grouping data such that observations within a group are similar to each other while being dissimilar to observations within other groups. Clustering algorithms are traditionally divided into one of several categories which include partitioning, hierarchical, model, density, and grid based approaches.

In density-based clustering, such as the popular *DBSCAN* [1], the definition of clustering is refined as identifying dense regions in feature space which are separated by regions of low density. With respect to clustering, a dense region is defined by the group of observations lying within it. Several desirable properties of density-based clustering include an ability to handle and identify noise, discover clusters with arbitrary shapes, and automatic discovery of the number of clusters.

Solutions for the density-based clustering problem can be broken up into procedures for performing two tasks. First, a procedure for estimating the density of each observation is defined and applied to identify observations lying within dense regions (core observations). Second, a region growing procedure is defined that identifies the group of observations which are reachable from some core observation. Key to the region growing procedure is the definition of observation reachability which must be restricted to the use of dense regions (i.e., no two observations should be reachable through a region of low density).

In *DBSCAN*, observation density is defined by two parameters, min_{pts} and eps. Here an observation is defined as dense if at least min_{pts} observations are within eps distance from it. Observation reachability is defined as being within eps from an observation or reachable through some chain of dense observations which are within eps distance of each other.

In contrast to *DBSCAN*, reverse nearest neighbor approaches such as *RECORD* [2], *IS-DBSCAN* [3], *ISB-DBSCAN* [4], and *RNN-DBSCAN*, define observation density using reverse nearest neighbors. All of these approaches require the use of a single parameter k, number of nearest neighbors, which is also used to identify dense observations. For example, in *RECORD*, if an observation has k or more reverse nearest neighbors it is identified as a dense observation. Additionally, in *RECORD*, observation reachability is defined by core observation traversals of the reverse k nearest

• *A. Bryant is with the Department of Computer Science, Virginia Commonwealth University, Richmond, VA 23284, and the Naval Surface Warfare Center Dahlgren Division, Dahlgren, VA 22448.*
E-mail: bryantac@vcu.edu.
• *K. Cios is with Virginia Commonwealth University, Richmond, VA 23284 and the Polish Academy of Sciences, Lublin 20-290, Poland.*
E-mail: kcios@vcu.edu.

Manuscript received 29 Mar. 2017; revised 21 Aug. 2017; accepted 11 Dec. 2017. Date of publication 27 Dec. 2017; date of current version 27 Apr. 2018.
(Corresponding author: Avory Bryant.)
Recommended for acceptance by E. Lim.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2017.2787640

neighbor graph. More in-depth descriptions of the related approaches and *DBSCAN* can be found in Section 2.1.

Benefits of the reverse nearest neighbor approaches over *DBSCAN* are two fold. First, there is a reduction in problem complexity as they only require the use of a single parameter, k, whereas *DBSCAN* requires two parameters, min_{pts} and eps. Second, *DBSCAN*'s use of the distance-based threshold eps, under certain conditions, leads to the algorithm's inability to distinguish amongst clusters with large variations in density. Additionally, to ensure deterministic properties of *DBSCAN* clustering results, a metric (symmetric) distance measure must be used, whereas the reverse nearest neighbor approaches have no such restriction.

With respect to the second benefit, in *DBSCAN* eps and min_{pts} are used to define density which is applied globally throughout the data. In contrast, with respect to *DBSCAN*'s definition of density, density in the reverse nearest neighbor approaches is determined locally (i.e., with respect to observations local neighborhood which is defined by k). This allows these methods to identify dense regions of space, again using *DBSCAN*'s definition of density, that can vary greatly in density.

The remainder of this paper is outlined as follows. In Section 2, the *DBSCAN*, *RECORD*, *IS-DBSCAN*, and *ISB-DBSCAN* algorithms are described along with *RNN-DBSCAN*. The effect of dataset size on the choice of k is examined in Section 3. Section 4 presents two heuristics for determining an appropriate value of k for *RNN-DBSCAN*. Performance of *RNN-DBSCAN* is compared with competing approaches on several artificial and real-world datasets in Section 5. Additionally, Section 5 discusses the use of an approximate k nearest neighbor algorithm with *RNN-DBSCAN*. Finally, in Section 6 prior research in the area of density-based clustering is presented, while Section 7 summarizes the findings of this paper.

2 Methodology

2.1 Related Clustering Approaches

In this section, graph-based interpretations of the *DBSCAN*, *RECORD*, *IS-DBSCAN*, and *ISB-DBSCAN* algorithms are given. Note that a similar interpretation of *RNN-DBSCAN* is given in Section 2.3.

2.1.1 DBSCAN

DBSCAN [1] can be described with respect to the directed eps neighborhood graph $G_{eps} = (V, E)$ where V is the set of observations and $\forall(\mathbf{u}, \mathbf{v}) \in E : \mathbf{v}$ is in the neighborhood (within eps distance) of $\mathbf{u}$. The set of core (dense) observations, $Core \subseteq V$, is defined as the set of observations whose eps neighborhood size is greater than or equal to min_{pts}, $Core = \{\mathbf{v} \in V | outdegree(\mathbf{v}) \geq min_{pts}\}$.

A clustering $C = \{C_1, \ldots, C_l\}$ is identified by iterating over the set of unclustered observations, such that for an unclustered core observation $\mathbf{v}$ at cluster C_i (i.e., $\mathbf{v} \in Core$ and $\mathbf{v} \notin C_{<i}$) a breadth first traversal of G_{eps} is performed (restricted to paths whose non-terminal vertexes are core observations). Note that the order in which clusters are discovered is dependent on the order in which observations are presented (i.e., cluster C_i is discovered before cluster C_{i+1}). With this in mind, the notation $C_{<i}$ indicates the set of previously discovered clusters, and $v \notin C_{<i}$ indicates an unclustered observation at cluster i. Unclustered vertexes that are thus reachable from $\mathbf{v}$, along with $\mathbf{v}$, forming new cluster C_i. More formally, given $\mathbf{v}$, cluster $C_i \in C$ is equal to the union of v and the set of observations $\{\mathbf{u} \in V | \exists$ a $directedpathP : \mathbf{v} = \mathbf{u}_0, \ldots, \mathbf{u}_l = \mathbf{u}$ where $\forall$ edges $(\mathbf{u}_i, \mathbf{u}_{i+1}) \in P : (\mathbf{u}_i, \mathbf{u}_{i+1}) \in E, \mathbf{u}_i \in Core$, and $\mathbf{u}_i, \mathbf{u}_{i+1} \notin C_{<i}\}$.

All clustered non-core vertexes are identified as border observations, $Border = \{\mathbf{v} \in V | \mathbf{v} \in C$ and $\mathbf{v} \notin Core\}$, and the set of unclustered observations identified as noise, $Noise = \{\mathbf{v} \in V | \mathbf{v} \notin C\}$. Note that by assuming a metric (symmetric) distance measure, G_{eps} can be viewed as an undirected graph (i.e., $(\mathbf{u}, \mathbf{v}) \in E \Rightarrow (\mathbf{v}, \mathbf{u}) \in E$). Under this condition core observation cluster assignments are deterministic or independent of the order in which they are iterated over. This is due to the fact that all core observations, $\mathbf{v} \in C_i$, are guaranteed to be reachable (as defined above) from any other core observation, $\mathbf{u} \in C_i$. However, assignment of border observations to clusters is non-deterministic being dependent on the order in which clusters are discovered.

Additionally, assuming a metric distance measure, an equivalent clustering of core observations in C can be identified as strongly connected components within the subgraph $G_{eps}/(V/Core)$ (i.e., G_{eps} with non-core vertexes removed). Border observations can be added to clusters (i.e., strongly connected components), such that for cluster C_i, border observation $\mathbf{v}$ can be added to C_i iff $(\mathbf{u}, \mathbf{v}) \in E$, $\mathbf{u} \in C_i$, and $\mathbf{u} \in Core$.

As previously discussed, with respect to the reverse nearest neighbor approaches, there are two drawbacks to the use of *DBSCAN*. These drawbacks include problem complexity (requires two parameters), and an inability to distinguish amongst clusters with high variations in density. In regards to the later, *DBSCAN* fails when the distance between two or more classes is less than the minimum eps required to identify all classes (see results for the *grid* dataset in Section 5). Note that a description of all datasets used in this paper can be found in Section 5. In such a case, *DBSCAN* can be used with the minimum eps required which will incorrectly identify multiple classes as a single cluster, or *DBSCAN* can be used with some smaller value of eps which will incorrectly identify a class as noise.

With respect to the problem complexity of *DBSCAN*, it is a commonly held belief that to some degree *DBSCAN* performance is invariant with respect to the choice of min_{pts} (i.e., the parameter selection task is to identify eps for some fixed min_{pts}). While intuitively there exists a positive correlation between the choice of eps and min_{pts} (e.g., when considering a larger value of min_{pts}, a larger value of eps must likewise be chosen, and vice versa, to produce results which are similar to the original value), Fig. 1 shows that performance of *DBSCAN* is in some cases highly dependent on the choice of min_{pts}.

2.1.2 RECORD

In *RECORD* [2], the directed k nearest neighbor graph $G_{kNN} = (V, E)$ is used, such that V is the set of observations and $\forall(\mathbf{u}, \mathbf{v}) \in E : \mathbf{v}$ is a k nearest neighbor of $\mathbf{u}$. Let the directed k reverse nearest neighbor graph G_{RkNN} be defined by the transpose of G_{kNN} (i.e., edge directions are reversed).

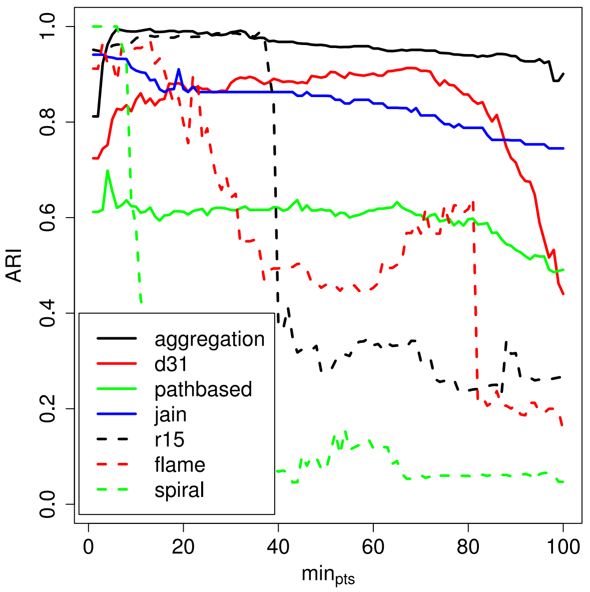

Fig. 1. Adjusted Rand Index (ARI) performance of *DBSCAN* over $1 \leq min_{pts} \leq 100$ for several artificial datasets. For each dataset and value of min_{pts}, an optimal value of eps, by maximum ARI, was selected. The search for the optimal eps was performed over the set of eps values equal to the min_{pts} nearest neighbor distance of each observation in the dataset.

The set of core observations, $Core \subseteq V$, are defined as observation whose number of reverse nearest neighbors is greater than or equal to k, $Core = \{\mathbf{v} \in V | outdegree(\mathbf{v}) \geq k\}$ in G_{RkNN}. All remaining observations are identified as outliers, $Outliers = V/Core$.

An initial Clustering $C = \{C_1, \ldots, C_l\}$ of core observations is defined by strongly connected components within the core observation subgraph, $G_{RkNN}/Outliers$. Next, outliers are incorporated into C, such that an outlier, $\mathbf{v} \in Outlier$, is inserted into cluster $C_i \in C$ iff C_i is the closest majority cluster of $\mathbf{v}$'s reverse nearest neighbors. Here majority is defined as of size at least k/d where d is the dimensionality of the data. Finally, any remaining unclustered observations are identified as noise. Note that such an approach is deterministic with respect to core observation cluster assignments, but not outlier observation cluster assignments. For example, consider the rare case where an outlier is of equal distance to two majority clusters.

Results in Section 5 show *RECORD* to be overly restrictive in identifying cluster members with many observations being incorrectly identified as noise. This is likely due to the reachability of core observations during clustering being restricted to paths which traverse reverse k nearest neighbor edges only, whereas *RNN-DBSCAN* considers both reverse and k nearest neighbor edge traversals.

2.1.3 IS-DBSCAN

IS-DBSCAN [3] is unique in its use of the influence space in defining observation density and reachability. An observation's influence space, IS, is defined as the intersect between its reverse and k nearest neighbor sets. Here the undirected influence space graph $G_{IS} = (V, E)$ is used, such that V is the set of observations and $\forall(\mathbf{u}, \mathbf{v}) \in E : \mathbf{v}$ is in the influence space of $\mathbf{u}$ ($\mathbf{v} \in IS(\mathbf{u})$). Note that graph G_{IS} can be treated as undirected as the IS relationship is symmetric ($\mathbf{u} \in IS(\mathbf{v}) \Rightarrow \mathbf{v} \in IS(\mathbf{u})$). The set of core observations, $Core \subseteq V$, is defined as observations whose influence space is of size greater than $2k/3$ (a predetermined threshold), $Core = \{\mathbf{v} \in V | \ |IS(\mathbf{v})| > 2k/3\}$.

A clustering $C = \{C_1, \ldots, C_l\}$ is identified by iterating over the set of unclustered observations, such that for an unclustered core observation $\mathbf{v}$ at cluster C_i (i.e., $\mathbf{v} \in Core$ and $\mathbf{v} \notin C_{<i}$) a depth first traversal of G_{IS} is performed (restricted to paths whose non-terminal vertexes are core observations). Unclustered vertexes that are thus reachable from $\mathbf{v}$, along with $\mathbf{v}$, forming new cluster C_i. If the size of C_i is not greater than k ($|C_i| \leq k$), then C_i is discarded and all of its member vertexes left unclustered. Note that the use of a depth fist versus a breadth first traversal has no effect on clustering results. As with *DBSCAN*, more formally, given $\mathbf{v}$, cluster $C_i \in C$ is equal to the union of $\mathbf{v}$ and the set of observations $\{\mathbf{u} \in V | \exists$ a undirected path $P : \mathbf{v} = \mathbf{u}_0, \ldots, \mathbf{u}_l = \mathbf{u}$ where $\forall$ edges $(\mathbf{u}_i, \mathbf{u}_{i+1}) \in P : (\mathbf{u}_i, \mathbf{u}_{i+1}) \in E, \mathbf{u}_i \in Core$, and $\mathbf{u}_i, \mathbf{u}_{i+1} \notin C_{<i}\}$.

IS-DBSCAN shares the same deterministic properties as *DBSCAN* (assuming a metric distance) with respect to cluster assignments of observation types. However, *IS-DBSCAN* is not dependent on the use of a metric distance as the symmetry of the *IS* relationship is independent of this condition. Also, similar to *DBSCAN*, an equivalent clustering of core-observations in C can be identified as connected components in the subgraph $G_{IS}/(V/Core)$ (i.e., G_{IS} with non-core vertexes removed), and non-core observations assigned to the same cluster as observations within their influence space.

In addition to the clustering phase, *IS-DBSCAN* uses an initial outlier detection phase based on a variant of the authors' *STRATIFY* [3] procedure. Here observations identified as outliers are removed before clustering. Also, an extra density-based dimension is added to each observation before clustering. This extra dimension being set to the sum of distances between an observation and the observations within its influence space. This last step is performed in hope of producing clusters of observations with similar density.

The main distinction between this approach and *RNN-DBSCAN* is the use of influence space in determining observation density and reachability. Results in Section 5 show *IS-DBSCAN* as being unable to identifying the underlying cluster structure in the case of several datasets. In such instances, this inability is most likely due to its use of the predetermined threshold which can be viewed as a hidden second parameter. Additionally, the outlier removal phase contributes to error by incorrectly identifying observations as noise. This last point is particularly true for datasets that do not contain noise.

2.1.4 ISB-DBSCAN

Finally, as in *IS-DBSCAN*, *ISB-DBSCAN* [4] uses the undirected influence space graph G_{IS}, along with an identical definition of core observations, $Core = \{\mathbf{v} \in V | \ |IS(\mathbf{v})| > 2k/3\}$. Clusters are identified in *ISB-DBSCAN* using a method that is similar to *IS-DBSCAN*, but limited to core observations. In other words, for all paths P in the depth-first traversal, $\forall \mathbf{v} \in P : \mathbf{v} \in Core$. Additionally, there is no requirement on the minimum size of a cluster. Finally, a phase for incorporating non-core observations into the clustering solution is introduced. For all non-core observations $\mathbf{v} \in V/Core$, $\mathbf{v}$ is assigned to the cluster of the closest core observation $\mathbf{u}$ in its influence space, $\mathbf{u} \in C_i \Rightarrow \mathbf{v} \in C_i$. All remaining unclustered observation (i.e., those with no core-observations in their influence space) are identified as noise.

Similar to the above approaches, *ISB-DBSCAN* is deterministic with respect to core observation cluster assignments, but not non-core observation assignments. For example, consider the rare case when a non-core observation has two core observations in its influence space, each belonging to different clusters, such that both core observations are of equal distance to it. Additionally, an equivalent clustering to this approach can be found by connected components within the subgraph $G_{IS}/(V/Core)$, along with an identical procedure for incorporating non-core observations.

As should be expected, drawbacks of *ISB-DBSCAN* are similar to those discussed in *IS-DBSCAN*. As previously discussed, this is most likely due to the use of the fixed predetermined threshold for determining core observation. Results in Section 5 show *ISB-DBSCAN* as being unable to identifying the underlying cluster structure in the case of several datasets. However, in most cases, *ISB-DBSCAN* is shown to out perform *IS-DBSCAN* which is likely due to an improved approach in the handling of non-core observations.

2.2 RNN-DBSCAN Definitions

Let X represent a set of observations of size $n = |X|$, such that each observation in X is drawn from a d-dimensional space of real values, $\forall \mathbf{x} \in X : \mathbf{x} \in \mathbb{R}^d$. For any two observations $\mathbf{x}, \mathbf{y} \in X$, let $dist(\mathbf{x}, \mathbf{y})$ represent a distance function (metric or non-metric) that returns the distance between observations $\mathbf{x}$ and $\mathbf{y}$. Note that the euclidean distance was used for all results produced in this paper, $dist(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{d} (\mathbf{y}_i - \mathbf{x}_i)^2}$.

For observations $\mathbf{x} \in X$, two neighborhood functions are defined based on k nearest neighbors given some integer k, $0 \leq k \leq n - 1$.

Definition 1 (k-nearest neighborhood of observation x). *The k-nearest neighborhood of observation $\mathbf{x}$ is defined by the function $N_k(\mathbf{x}) = N$ where N satisfies the following conditions:*

1) $N \subseteq X/\{\mathbf{x}\}$
2) $|N| = k$
3) $\forall \mathbf{y} \in N, \mathbf{z} \in X/(N + \{\mathbf{x}\}) : dist(\mathbf{x}, \mathbf{y}) \leq dist(\mathbf{x}, \mathbf{z})$

Definition 2 (reverse nearest neighborhood of observation x). *The reverse nearest neighborhood of observation $\mathbf{x}$ is defined by the function $R_k(\mathbf{x}) = R$ where R satisfies the following conditions:*

1) $R \subseteq X/\{\mathbf{x}\}$
2) $\forall \mathbf{y} \in R : \mathbf{x} \in N_k(\mathbf{y})$

Similar to *DBSCAN*, three types of observations are defined in X which are core, boundary, and noise. An observation $\mathbf{x} \in X$ is a core observation iff $|R_k(\mathbf{x})| \geq k$, whereas $\mathbf{x}$ is a boundary or noise observation if $|R_k(\mathbf{x})| < k$. Border observations are those of the aforementioned set which are assigned to clusters, while noise observations remain unclustered.

Next, observation reachability definitions are presented which will be used in defining clusters. An observation's direct reachability is limited to its set of k nearest neighbors along with its status as a core observation.

Definition 3 (directly density-reachable). *A observation $\mathbf{x}$ is directly density reachable from a observation $\mathbf{y}$ if*

1) $\mathbf{x} \in N_k(\mathbf{y})$
2) $|R_k(\mathbf{y})| \geq k$ *(core observation condition)*

Directly density reachable is non-symmetric for non-core observations, and not guaranteed to be symmetric in the case of core observations. The latter of the two cases being due to the fact that the nearest neighbor relationship is non-symmetric, and the former as no observation is reachable from a non-core observation.

Definition 4 (density-reachable). *A observation $\mathbf{x}$ is density-reachable from a observation $\mathbf{y}$ if there is a chain of observations $\mathbf{x}_1, \ldots, \mathbf{x}_m$, $\mathbf{x}_1 = \mathbf{y}$, $\mathbf{x}_m = \mathbf{x}$ such that where $|R_k(\mathbf{x})| \geq k$*

1) $\forall 1 \leq i \leq m-1 : \mathbf{x}_{i+1}$ *is directly density-reachable from $\mathbf{x}_i$ or $\mathbf{x}_i$ is directly density-reachable from $\mathbf{x}_{i+1}$*

and where $|R_k(\mathbf{x})| < k$

1) $\mathbf{x}_m$ *is directly density-reachable from* $\mathbf{x}_{m-1}$
2) $\forall 1 \leq i \leq m-2 : \mathbf{x}_{i+1}$ *is directly density-reachable from $\mathbf{x}_i$ or $\mathbf{x}_i$ is directly density-reachable from $\mathbf{x}_{i+1}$*

Density reachable is a canonical extension of directly density reachable, and is transitive though not symmetric. However, this relationship is symmetric where observations $\mathbf{x}$ and $\mathbf{y}$ are core observation. The prior statement being due to the fact that sequences of core observation within the path may be directly density reachable in either direction.

Definition 5 (density-connected). *A observation $\mathbf{x}$ is density-connected to a observation $\mathbf{y}$ if there is a observation $\mathbf{z}$ such that both, $\mathbf{x}$ and $\mathbf{y}$ are density reachable from $\mathbf{z}$.*

Density connected is a symmetric relationship over all observation types. Now using the above reachability definitions a cluster of observations is defined as follows.

Definition 6 (cluster). *A cluster C is a non-empty subset of X, $\emptyset \neq C \subseteq X$, satisfying the following conditions:*

1) $\forall \mathbf{x}, \mathbf{y} \in X$: *if $\mathbf{x} \in C$ and $\mathbf{y}$ is density-reachable from $\mathbf{x}$ then $\mathbf{y} \in C$ (maximality)*
2) $\forall \mathbf{x}, \mathbf{y} \in C$: *$\mathbf{x}$ is density-connected to $\mathbf{y}$ (connectivity)*

Note that the two conditions defining a cluster C are satisfied with respect to a soft clustering (i.e., observations may be assigned to multiple clusters). In the case of a hard clustering (i.e., observations may be assigned to only one cluster), the conditions are both satisfied for core observations only. For a non-core observation $\mathbf{x}$ in cluster C (a border observation), there may exists another cluster C' (or several clusters) such that $\mathbf{x}$ is reachable from some core observation in C'. In such a case, $\mathbf{x}$ must be assigned to only one of the clusters by some selection process (e.g., several of such processes are discussed in Section 2.1). *RNN-DBSCAN* takes an arbitrary approach to the cluster assignment of such border observations that is dependent on observation ordering.

Given cluster C, let the density of C be the maximum of the directly density-reachable distances among core observations in C.

Definition 7 (density of cluster C). *The density of cluster C is defined by the function $den(C)$:*

$$den(C) = \max_{(\mathbf{x},\mathbf{y})} dist(\mathbf{x},\mathbf{y}),$$
$$\forall \mathbf{x},\mathbf{y} \in C : |R_k(\mathbf{x})| \geq k,\ |R_k(\mathbf{y})| \geq k, \text{ and}$$
$$\mathbf{y} \text{ is directly density-reachable from } \mathbf{x}$$

Once all clusters have been identified, given cluster C and its density $den(C)$, let the extended cluster of C, C^{ex}, be a superset of C, $C^{ex} \supseteq C$, that is extended by inserting unclustered observations into C. An unclustered observation is inserted into C^{ex} if its closest core k nearest neighbor, within distance $den(C)$, is in C.

Definition 8 (extended cluster). *Given cluster C, its extended cluster C^{ex} is equal to the union of C and all observations $\mathbf{x}$ satisfying the following conditions:*

1) *$\mathbf{x} \notin C$ and there exists no other cluster $C' \neq C$ such that $\mathbf{x} \in C'$ (unclustered)*
2) *There exists an observation $\mathbf{y} \in C$ such that $|R_k(\mathbf{y})| \geq k$, $\mathbf{y} \in N_k(\mathbf{x})$, and $dist(\mathbf{x},\mathbf{y}) \leq den(C)$ (within $den(C)$ distance to a core k nearest neighbor)*
3) *There exists no observation $\mathbf{z} \in C'$, $C' \neq C$, for which the previous condition holds and $dist(\mathbf{x},\mathbf{z}) < dist(\mathbf{x},\mathbf{y})$ (closest)*

Finally, all remaining unclustered observations are identified as noise.

Definition 9 (noise). *Let $C_1^{ex},\ldots,C_l^{ex}$ be the clusters of X. Noise is defined as the set of observations in X which do not belong to any cluster $C^{ex}_{1\leq i\leq l}$, $noise = \{\mathbf{x} \in X | \forall 1 \leq i \leq l : \mathbf{x} \notin C_i^{ex}\}$.*

2.3 RNN-DBSCAN Algorithm

Algorithm 1 presents a procedure for performing an *RNN-DBSCAN* clustering $C_1^{ex},\ldots,C_l^{ex}$, as defined in Section 2.2, given dataset X and nearest neighbor parameter k. For all observations, traversed in some arbitrary order, if the current (seed) observation has yet to be assigned to a cluster and is a core observation, then it is assigned to a new cluster. This new cluster is expanded by a breadth first search of all unclustered reachable observations, density-connected, from the seed observation (Algorithm 2 and 3). Finally, the resulting clustering is expanded by Algorithm 4.

Algorithm 1. $RNN-DBSCAN(X,k)$

```
1: $assign[\forall \mathbf{x} \in X] = UNCLASSIFIED$
2: $cluster = 1$
3: for all $\mathbf{x} \in X$ do
4:   if $assign[\mathbf{x}] = UNCLASSIFIED$ then
5:     if $ExpandCluster(\mathbf{x}, cluster, assign, k)$ then
6:       $cluster = cluster + 1$
7:     end if
8:   end if
9: end for
10: $ExpandClusters(X, k, assign)$
11: return $assign$
```

Algorithm 2. $ExpandCluster(\mathbf{x}, cluster, assign, k)$

```
1: if $|R_k(\mathbf{x})| < k$ then
2:   $assign[\mathbf{x}] = NOISE$
3:   return $FALSE$
4: else
5:   initialize empty queue $seeds$
6:   $seeds.enqueue(Neighborhood(\mathbf{x}, k))$
7:   $assign[\mathbf{x} + seeds] = cluster$
8:   while $seeds \neq \emptyset$ do
9:     $\mathbf{y} = seeds.dequeue()$
10:    if $R_k(\mathbf{y}) \geq k$ then
11:      $neighbors = Neighborhood(\mathbf{y}, k)$
12:      for all $\mathbf{z} \in neighbors$ do
13:        if $assign[\mathbf{z}] = UNCLASSIFIED$ then
14:          $seeds.enqueue(\mathbf{z})$
15:          $assign[\mathbf{z}] = cluster$
16:        else if $assign[\mathbf{z}] = NOISE$ then
17:          $assign[\mathbf{z}] = cluster$
18:        end if
19:      end for
20:    end if
21:  end while
22:  return $TRUE$
23: end if
```

Algorithm 3. $Neighborhood(\mathbf{x}, k)$

```
1: $neighbors = N_k(\mathbf{x}) + \{\mathbf{y} \in R_k(\mathbf{x}) : |R_k(\mathbf{y})| \geq k\}$
2: return $neighbors$
```

Algorithm 4. $ExpandClusters(\mathbf{x}, k, assign)$

```
1: for all $\mathbf{x} \in X$ do
2:   if $assign[\mathbf{x}] = NOISE$ then
3:     $neighbors = N_k(\mathbf{x})$, $mincluster = NOISE$, $mindist = \infty$
4:     for all $\mathbf{n} \in N$ do
5:       $cluster = assign[\mathbf{n}]$, $d = dist(x, n)$
6:       if $|R_k(\mathbf{n})| \geq k$ & $d \leq den(cluster)$ & $d < mindist$ then
7:         $mincluster = cluster$, $mindist = d$
8:       end if
9:     end for
10:    $assign[\mathbf{x}] = mincluster$
11:  end if
12: end for
```

Similar to the discussion of prior approaches, Section 2.1, a graph-based interpretation of *RNN-DBSCAN* is given. Let $G_{kNN} = (V, E)$ represent the directed k nearest neighbor graph, such that V is equal to the set of observations, $V = X$, and $\forall(\mathbf{u},\mathbf{v}) \in E : \mathbf{v}$ is a k nearest neighbor of $\mathbf{u}$, $\mathbf{v} \in N_k(\mathbf{u})$. The set of core observations, $Core$, is defined as observations whose number of reverse nearest neighbors is equal to or greater than k, $Core = \{\mathbf{v} \in V | indeg(\mathbf{v}) \geq k\}$.

An initial Clustering $C_1,\ldots,C_l$ of core observations is defined by weakly connected components within the core observation subgraph $G_{kNN}/(V/Core)$. Next, for all unclustered observations $\mathbf{v}$, $\mathbf{v} \notin C_{1\leq i\leq l}$, that are k nearest neighbors of some core observation $\mathbf{u} \in C_i$, $\mathbf{u} \in Core$ and $\mathbf{v} \in N_k(\mathbf{u})$ (or $(\mathbf{u},\mathbf{v}) \in E$ in G_{kNN}), $\mathbf{v}$ is assigned to cluster C_i, $C_i = C_i + \mathbf{v}$.

Finally, the remaining unclustered observations are used to produce the expanded clustering $C_1^{ex},\ldots,C_l^{ex}$ which are

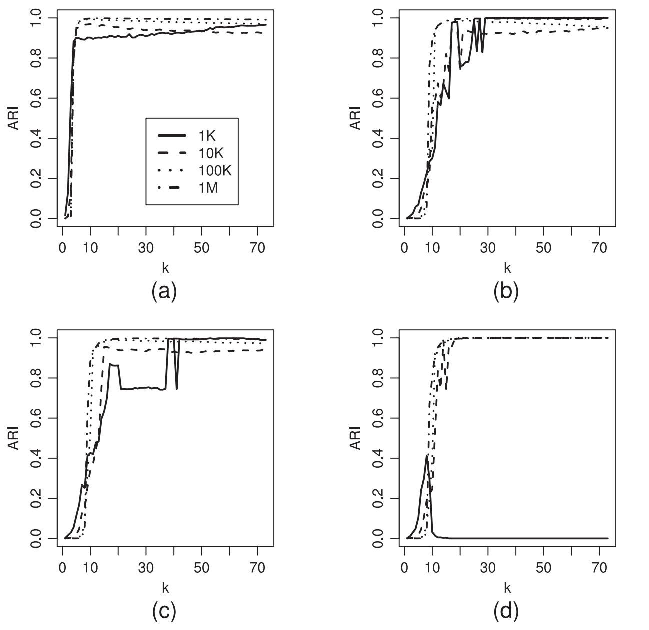

Fig. 2. ARI performance vs k on the blobs (a), circle (b), moons (c), and swissroll (d) datasets of sizes $1K$ (solid), $10K$ (dash), $100K$ (dot), and $1M$ (dot-dash).

initialized to the current clustering $C_1, \ldots, C_l$ (i.e., $C_i^{ex} = C_i$). For all unclustered observations $\mathbf{v}$, $\mathbf{v} \notin C_{1 \le i \le l}^{ex}$, $\mathbf{v}$ is assigned to the cluster C_i^{ex}, $C_i^{ex} = C_i^{ex} + \mathbf{v}$, of its closest core k nearest neighbor observation $\mathbf{u}$ whose distance from $\mathbf{v}$ is less than or equal to $den(C_i)$ (i.e., $\mathbf{u} \in Core$, $\mathbf{u} \in C_i^{ex}$, $\mathbf{u} \in N_k(\mathbf{v})$ (or $(\mathbf{v}, \mathbf{u}) \in E$ in G_{kNN}), and $dist(\mathbf{v}, \mathbf{u}) \le den(C_i)$).

2.4 RNN-DBSCAN Complexity

As with *DBSCAN*, *RECORD*, *IS-DBSCAN*, and *ISB-DBSCAN*, the computational complexity of *RNN-DBSCAN* is dependent on its solution to the all nearest neighbors problem. More specifically, *DBSCAN* is dependent on the all fixed-radius near neighbors problem, whereas the reverse nearest neighbor approaches are dependent on the all k nearest neighbors problem. A naive solution to the latter problem is $O(n^2)$, while a naive solution to the former is $O(k \times n^2)$.

In the original *DBSCAN* work [1], it is mentioned that by using an indexing structure (e.g., *R* Tree* [5] or *k-d Tree* [6]) computational complexity can be reduced to $O(n \times \log n)$, the complexity of fixed-radius near neighbors being reduced to $O(\log n)$. However, as pointed out in [7], [8] and alluded to in [1], such a solution is highly dependent on the amount of density variation in the data. This fact should not be surprising as in the case of complexity for indexing structures randomly distributed data is generally assumed. For euclidean space, [7], [8] goes on to present a $O(n \times \log n)$ solution in the two-dimensional case, and proves $\Omega(n^{4/3})$ complexity when dimensionality is greater than two. For the later case of data, in [7], [8] a ρ-approximate algorithm, *APPROXDBSCAN* is presented with $O(n)$ complexity.

Similar to the above solutions, the complexity of the k nearest neighbor problem can be improved by the above indexing structures, with the additional dependence on k, or others such as a *Cover Tree* [9] ($O(\eta \times \log n)$ where η is dependent on dimensionality). However, these methods inherit the same issues mentioned above with respect to density variation, dimensionality, and metric assumptions. As suggested above, approximate solutions can be applied to overcome these challenges. For example, *Locality-Sensitive Hashing* [10] (as seen in *ISB-DBSCAN*) represents an O(n) solution, and *NN-DESCENT* [11] (previously applied in *NG-DBSCAN* [12] to approximate *DBSCAN*) with a reported empirical complexity of $O(n^{1.3})$.

In Section 5, the use of *NN-DESCENT* is investigated with *RNN-DBSCAN*.

3 Effect of Dataset Size n on k

An interesting property to consider with respect to *RNN-DBSCAN* is the effect dataset size n has on the choice of k. In particular, one would like to know if the choice of k is independent of n. This might in fact be assumed as the expected number of reverse nearest neighbors, k, is independent with respect to the size of the dataset n. With this in mind, we assume that the choice of k should not be affected by n, except in cases where n is not sufficiently large enough to represent data. Note that a similar assumption is not true in the case of *DBSCAN* where eps and/or min_{pts} must be adjusted with respect to n. To test this assumption Fig. 2 shows plots of ARI performance over choices of k for several artificial datasets. Here the data generation process remains constant while varying the number of drawn observations ($n = 1K, 10K, 100K, 1M$).

From Fig. 2 one can observe that as n grows the plots of ARI performance over k converge (i.e., showing independence between k and n). This is particularly true at larger values of n ($1M$, $100K$, and to some extent $10K$). These results are of particular significance as they suggest that k may be selected by sampling of the dataset, assuming a large enough sample is taken (i.e., see the results at $n = 1K$). Additionally, note that when the plots converge k may be safely selected from a range of values. In other words, performance is stable with respect to k as evidences by the smoothness of the plots. This observation is carried over into the next section to develop a heuristic for choosing k.

4 Choice of k

As performance of *RNN-DBSCAN* is dependent on the choice of k, two potential heuristics for selecting an appropriate value of k are presented. Of all discussed prior reverse nearest neighbor approaches, only *RECORD* [2] addresses this issue by suggesting two distinct heuristics. One of said heuristics is based on the observation that good clustering solutions occur across multiple values of k. The first approach presented here is based on this observations.

Given dataset X, let function $l_k(X)$ return the number of clusters produced by an *RNN-DBSCAN* clustering of X with parameter k. Note that function l_k is not monotonically decreasing with respect to k (i.e., $\forall k$ the inequality $l_{k+1}(X) \le l_k(X)$ does not always hold), though a negative correlation does exist between the two (e.g., see Fig. 3 which shows plots of number of clusters versus k for *RNN-DBSCAN* on several artificial datasets).

To determine an appropriate value of k, clustering stability is examined with respect to the frequency of number of clusters, $l_k(X)$, for some range of k. Here it is assumed that a correlation exists between clustering performance and the frequency in which a clustering solution, defined by number

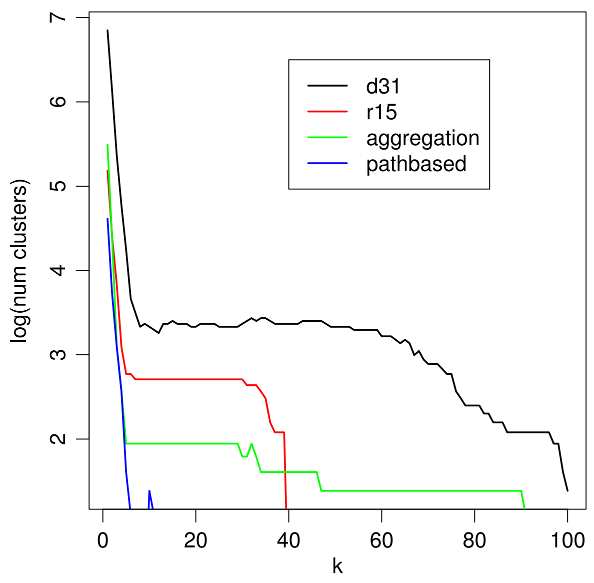

Fig. 3. Parameter k versus number of clusters (log) for *RNN-DBSCAN* clusterings produced over the range $1 \leq k \leq 100$ on the *d31*, *r15*, *aggregation*, and *pathbased* datasets.

of clusters, occurs. In particular, it is assumed that good performance can be found at spikes in the histogram of $l_k(X)$ over an appropriate range of k. Evidence in support of this assumption is shown in Fig. 4, where good performance is observed at the first frequency spike for all artificial datasets. Note that this only suggests an appropriate number of clusters which may be obtained from some set of k values.

In Fig. 4, performance at each value of $l_k(X)$ is shown with both its maximum ARI performance (solid line) and ARI performance at the minimum k (dashed line) producing a solution with $l_k(X)$ clusters. Here a positive correlation is observed between maximum ARI and minimum k ARI performance with respect to number of clusters. This suggest that given some desired number of clusters l (i.e., number of clusters at some frequency spike), k may be set to the minimum value which produces a clustering solution with l clusters.

To show the utility of this heuristic, it was applied to two unlabeled artificial datasets obtained from [13] (*t4* and *t7*, two popular shape datasets with noise). From Fig. 5 (note ARI performance is unknown), one can observe clustering results obtained by selecting the number of clusters l with maximum frequency, along with the minimum value of k which results in a solution with l clusters.

Next, the second heuristic is presented which is based on an existing internal validation measure designed for density-based clusterings, *DBCV* [14]. Note that a simplification of this measure described in [8] was used here. *DBCV* is based on the general assumption that in a good clustering, observations in a cluster should be tightly connected while observations belonging to different clusters should be far apart. With respect to tightly connected, for some cluster C_i, $DSC(C_i)$ defines the density sparseness of a cluster as the maximum distance in the minimum spanning tree of the core observations in C_i. With respect to clusters being far apart, for two clusters C_i and C_j, $DSPC(C_i, C_j)$ defines the density separation for a pair of clusters as the minimum distance between core observations in C_i and C_j.

For the clustering $C = \{C_1, \ldots, C_l\}$, the validity of cluster C_i is defined by the cluster validity index $V(C_i)$ which

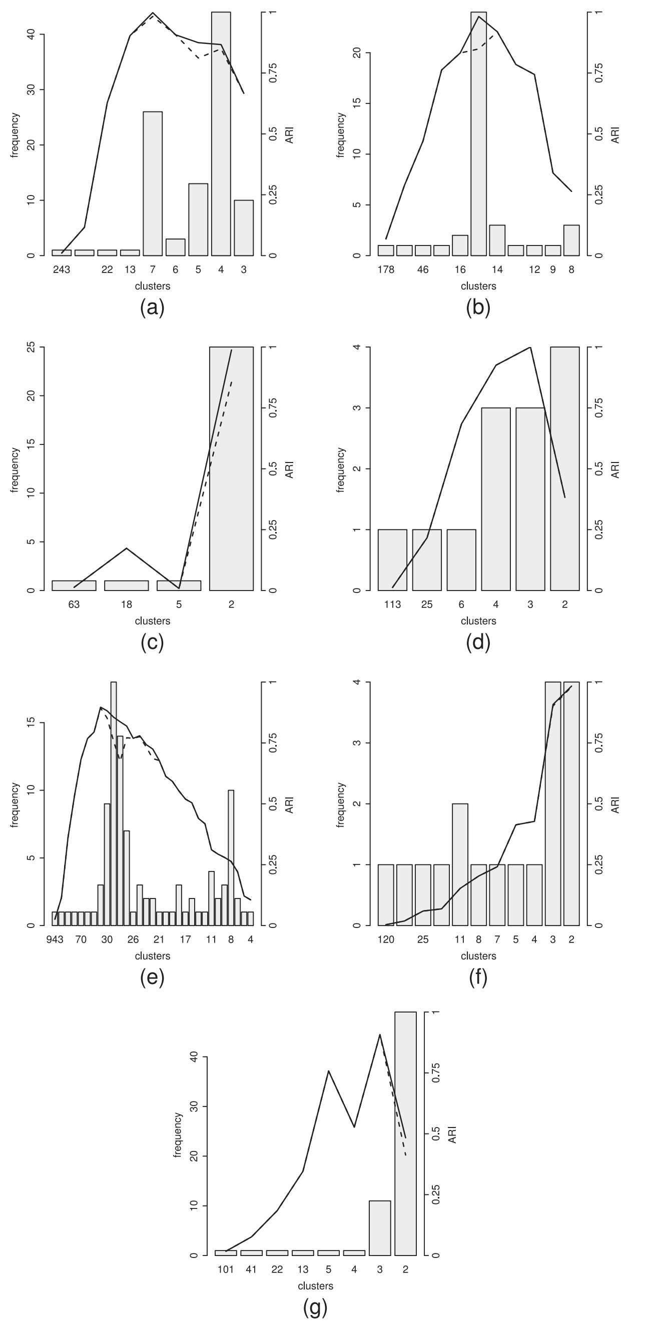

Fig. 4. Number of clusters versus frequency of occurrence (bars) and ARI performance (line) for *RNN-DBSCAN* clusterings produced over the range $1 \leq k \leq 100$ on the *aggregation* (a), *r15* (b), *flame* (c), *spiral* (d), *d31* (e), *jain* (f), and *pathbased* (g) datasets. Note that occurrences of number of clusters equal to 1 are not shown.

compares the cluster's DSC to its minimum $DSPC$ across the clustering:

$$V_C(C_i) = \frac{\min_{C_j \in C, C_j \neq C_i} DSPC(C_i, C_j) - DSC(C_i)}{\max(\min_{C_j \in C, C_j \neq C_i} DSPC(C_i, C_j), DSC(C_i))}$$

The validity index of clustering C, $DBCV(C)$, is defined as the weighted average of the cluster validity index for all clusters in C:

$$DBCV(C) = \sum_{i=1}^{l} \frac{|C_i|}{n} \times V_C(C_i)$$

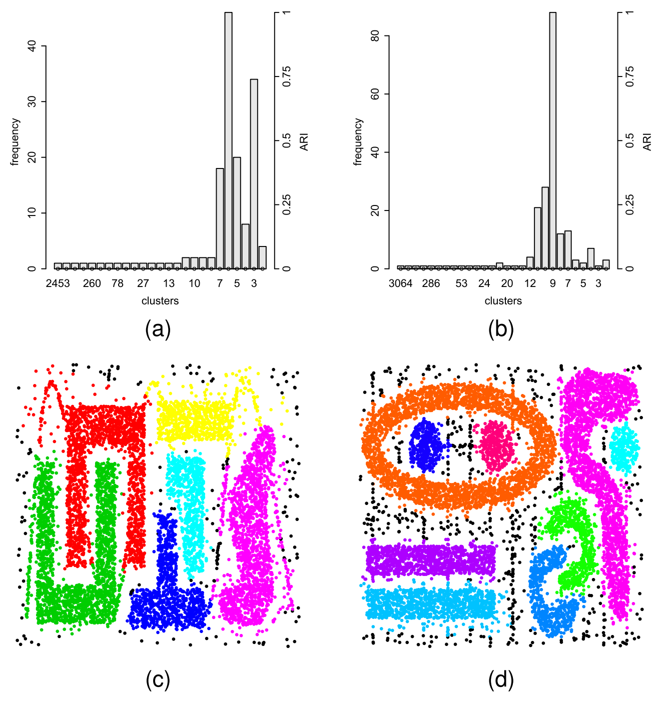

Fig. 5. Number of clusters versus frequency of occurrence for *RNN-DBSCAN* clusterings produced over the range $1 \leq k \leq 200$ on the *t4* (a) and *t7* (b) datasets. Note that occurrences of number of clusters equal to 1 are not shown. *RNN-DBSCAN* clustering results (number of clusters, 6 and 9, determined from (c) and (c) along with their corresponding minimum k values) for the *t4* (d) and *t7* (e) datasets.

Note that $DBCV(C) \in [-1, +1]$ where higher values indicate a better clustering.

Figure 6 shows ARI and *DBCV* performance over k for all artificial datasets. From these plots one observes a positive correlation between the two external (ARI) and internal *DBCV* evaluation measures. This suggests that the *DBCV* measure might be used to select an appropriate value of k. However, though positive this correlation is not exact (i.e., $+1$) where k at the maximum value of *DBCV* might not yield optimal results with respect to ARI. One potential improvement might be to examine *DBCV* results with respect to number of clusters, as with the first heuristic, though this is not investigated here.

5 Results and Discussion

To evaluate the performance of *RNN-DBSCAN*, with respect to prior approaches, several artificial, shaped-based clustering datasets from [15] were used. In addition to these, several artificial datasets of varying sizes where generated using the *scikit learn* toolkit [16], along with a grid-based dataset generated to highlight the density variation limitation of *DBSCAN*. Note that performance on the former dataset were discussed in Section 4 and are not presented below. This grid-based dataset consisted of two grids (g_1 and g_2, both with n observations) where the minimum distance between observations in g_1 is d and $d/2$ for g_2. Likewise, the minimum distance between observations in g_1 to observations in g_2 is d. Recall that given such a dataset *DBSCAN* will fail to identify the two grids as clusters, either incorrectly identifying both grids as a single cluster or identifying g_2 as a cluster and g_1 as noise. A summary of each dataset is provided in Table 1.

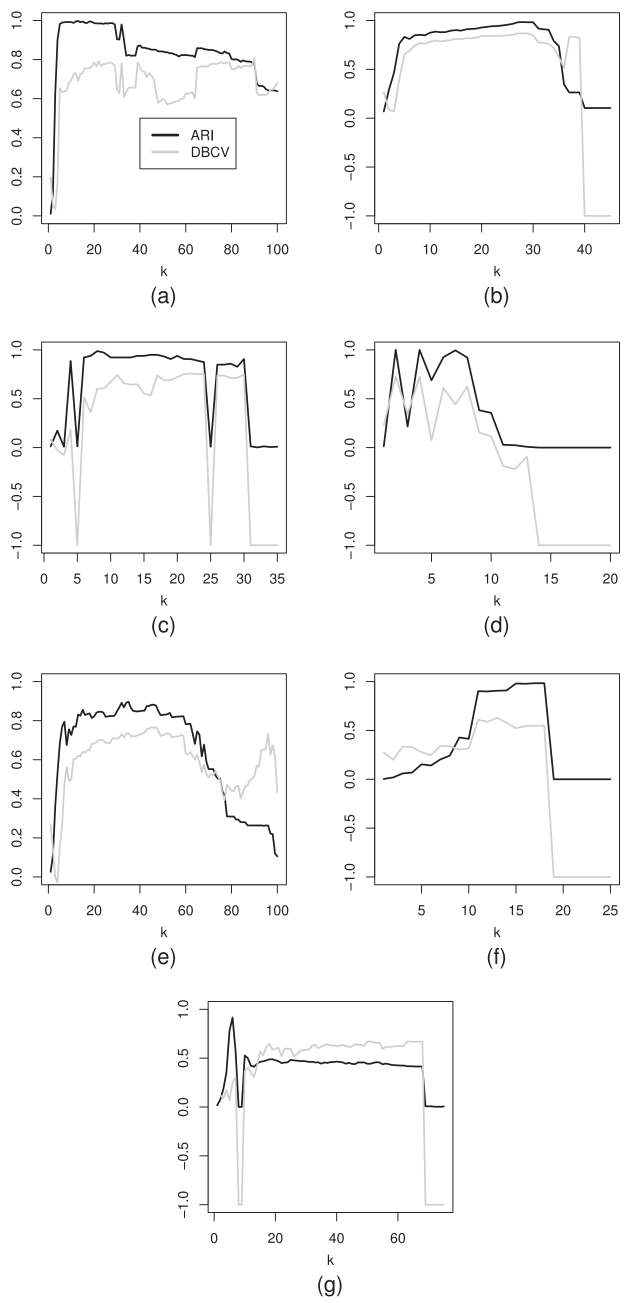

Fig. 6. Performance (ARI (black) and *DBCV* (gray)) versus k on the *aggregation* (a), *r15* (b), *flame* (c), *spiral* (d), *d31* (e), *jain* (f), and *path-based* (g) datasets.

Several real-word datasets were also included for evaluation, which were obtained from the UCI Machine Learning Repository [17] and [18]. Datasets from [18] are unlabeled, and thus used to evaluate performance of the approximate k nearest neighbor solution, along with the large ($1M$) artificial datasets. A summary of the data is shown in Table 2, and short descriptions of the domain of each set are as follows: banknote (banknote authentication), ctg (cardiotocography fetal state), digits (optical recognition of handwritten digits), ecoli (ecoli protein localization sites), htru2 (pulsar candidates), iris (iris plant type), seeds (varieties of wheat kernels), farm (farm satellite image), and house (individual household electric power consumption).

In addition to the clustering approaches discussed in Section 2, performance of the *OPTICS* [21] algorithm is also

TABLE 1
Artificial Datasets

Data	Observations	Classes	Dimensions
aggregation [15]	788	7	2
d31 [15]	3100	31	2
flame [15]	240	2	2
jain [15]	373	2	2
pathbased [15]	300	3	2
r15 [15]	600	15	2
spiral [15]	312	3	2
grid [15]	1250	2	2
blobs [16]	1K,10K,100K,1M	5	3
circle [16]	1K,10K,100K,1M	2	2
moons [16]	1K,10K,100K,1M	2	2
swissroll [16]	1K,10K,100K,1M	2	3

given for each datasets. *OPTICS* produces an augmented ordering of a dataset which can be used to extract any *DBSCAN* clustering with respect to a fixed min_{pts} and maximum value of eps. Note that the primary output of *OPTICS* is the dataset ordering with corresponding reachability distances which when plotted reveals the cluster structure of the data (e.g., clusters can be identified as dents in the reachability plot). While *OPTICS* does not explicitly produce a clustering, an automatic technique for extracting a clusterings from the reachability plots is also presented in [21] which we utilize here.

With respect to performance, Adjusted Rand Index (ARI) [22] and Normalized Mutual Information (NMI) [23], [24] were used for evaluation. ARI represents the similarity measure between two clusterings that is adjusted for chance and is related to accuracy, while NMI quantifies the amount of information obtained about one clustering, through the other clustering (i.e., the mutual dependence between the two). In the case of observations being identified as noise, each noise observation was treated as a distinct singleton cluster for both ARI and NMI. For ARI results, clustering Purity is also presented which is a weighted average of the percentage of observations belonging to the dominant class in each cluster. Noise observations were ignore in the calculation of purity.

With respect to the parameter search space of each clustering approach, nearest neighbor parameter k, for each of the reverse nearest approaches was chosen from the range $1 \leq k \leq 100$. For *DBSCAN*, min_{pts} was selected from the set $\{1, 5, 10, 20\}$, and eps selected over the set of eps values equal to the min_{pts} nearest neighbor distance of each observation. Similar to *DBSCAN*, *OPTICS* min_{pts} was selected from the same set of values with the maximum eps value set to the maximum min_{pts} nearest neighbor distance, and steepness parameter ξ taken from the range $0 \leq \xi \leq 1$. In all cases, two sets of parameters were selected which maximized both ARI and NMI respectively. Note that the observation ordering was fixed for each dataset and in all cases euclidean distance was used.

TABLE 2
Real-World Datasets

Data	Observations	Classes	Dimensions
banknote [17]	1372	2	4
ctg [17]	2126	10	19
digits [17]	1,797	10	64
ecoli [17]	336	8	7
htru2 [17], [19]	17,898	2	8
iris [17]	150	3	4
seeds [17]	210	3	7
farm [18], [20]	3,627,086	-	4
house [17], [18]	2,049,280	-	6

TABLE 3
ARI Performance on Artificial Datasets

Data		RNN	REC	IS	ISB	DBS	OPT
aggr	ari	**0.998**	0.752	0.872	0.914	0.994	0.979
	clu	7	7	6	6	7	8
	pur	0.999	1.0	0.956	0.956	0.999	0.987
	noi	0	163	34	0	2	0
d31	ari	**0.896**	0.539	0.71	0.739	0.868	0.874
	clu	31	38	34	43	31	60
	pur	0.975	0.928	0.901	0.861	0.982	0.95
	noi	167	1051	492	244	286	0
flam	ari	**0.971**	0.631	0.682	0.215	0.944	0.928
	clu	2	2	2	23	2	3
	pur	0.996	0.995	0.981	1.0	0.992	0.983
	noi	2	43	31	33	4	0
jain	ari	0.983	0.417	0.819	**1.0**	0.941	**1.0**
	clu	2	2	2	2	4	2
	pur	1.0	1.0	1.0	1.0	1.0	1.0
	noi	2	115	34	0	1	0
path	ari	**0.917**	0.763	0.759	0.789	0.655	0.684
	clu	3	3	5	5	10	7
	pur	0.99	1.0	0.986	0.989	0.986	0.957
	noi	11	50	21	16	11	0
r15	ari	0.984	0.751	0.807	**0.993**	0.979	0.956
	clu	15	14	15	15	15	16
	pur	0.995	0.932	0.986	0.997	0.995	0.977
	noi	3	103	91	0	6	0
spir	ari	**1.0**	**1.0**	0.947	**1.0**	**1.0**	0.653
	clu	3	3	3	3	3	6
	pur	1.0	1.0	1.0	1.0	1.0	0.888
	noi	0	0	11	0	0	0
grid	ari	**1.0**	0.922	0.994	0.997	0.5	0.997
	clu	2	2	2	2	1	3
	pur	1.0	1.0	0.999	0.999	1.0	0.999
	noi	0	50	2	0	625	0

Table 3 shows the ARI performance, Purity, number of cluster, and number of observation identified as noise for *RNN-DBSCAN* (RNN), *RECORD* (REC), *IS-DBSCAN* (IS), *ISB-DBSCAN* (ISB), *DBSCAN* (DBS), and *OPTICS* (OPT) on the artificial datasets. From these results, *RNN-DBSCAN* is shown to rank first in six of eight datasets (tied for first in one case), while ranking second in the other two datasets (two methods tied for first in one case). More importantly, in each case *RNN-DBSCAN* was able to identify the underlying classes of each dataset (i.e., at the maximum ARI solution number of clusters equals the number of expected classes), whereas each of the other approaches fail at this task in at least one case.

By design, *DBSCAN* is shown to perform poorly on the *grid* dataset, whereas each of the other approaches, in varying degrees, are able to identify each grid in the dataset. Fig. 7 shows that *DBSCAN* incorrectly identifies one of the

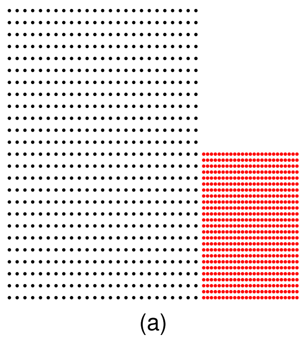

Fig. 7. *DBSCAN* (a) and *RNN-DBSCAN* (b) clustering results (maximum ARI solution) for the *grid* dataset. Note that observations colored black were identified as noise by the clustering.

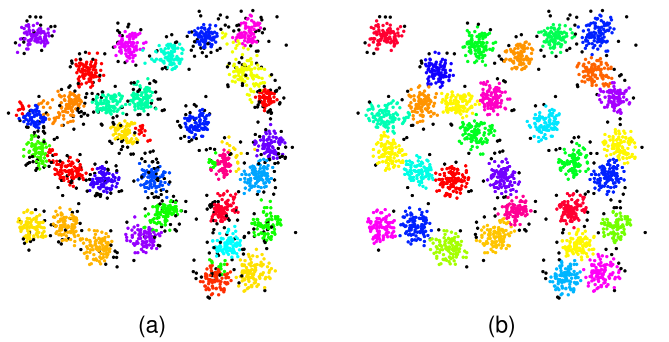

Fig. 9. *IS-DBSCAN* (a) and *RNN-DBSCAN* (b) clustering results (maximum ARI solution) for the *d31* dataset. Note that observations colored black were identified as noise by the clustering.

grids as noise, while *RNN-DBSCAN* is able to correctly identify both grids. Overall, *DBSCAN* performs well across all the datasets, excluding *grid* and *pathbased*, though it does require the use of two parameters (min_{pts} and eps). Note that the *pathbased* dataset consisted of three classes (two blobs and one low chain with varying densities).

IS-DBSCAN and *ISB-DBSCAN* fail in several cases that can be attributed to one of two reasons. First, with respect to both *IS-DBSCAN* and *ISB-DBSCAN*, the influence space based approach is unable to uncover the underlying class structure as shown in Fig. 8 (splits classes amongst multiple clusters) and Fig. 9 (splits classes amongst multiple clusters and merges classes into a single clusters). Second, with respect to *IS-DBSCAN*, many observations are incorrectly identified as outliers as shown in Fig. 9 and Table 3.

Similar to *IS-DBSCAN*, the main shortcoming of *RECORD* is that many observations are incorrectly identified as noise as seen in Table 3.

For *OPTICS*, the use of the automated technique to extract clusters by identifying dents in the reachability should be considered. In fact, *OPTICS* performance could be identical to *DBSCAN* where clusters are identified by a simple cut in the reachability plot (i.e., the reachability plot contains information to extract *DBSCAN* clusterings over a range of eps values). The automated technique essentially introduces the additional problem of identifying an appropriate number of clusters, the benefit of which is an improved ability to handle density variations (i.e., see *OPTICS* results on the *grid* dataset as compared to *DBSCAN*). In Table 3 one can see that the automated version of *OPTICS* tends to overestimate the number of clusters

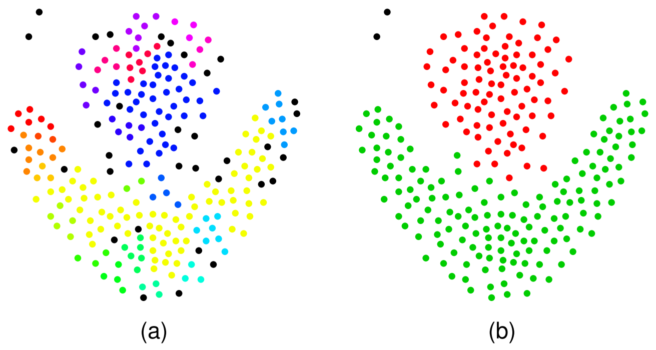

Fig. 8. *ISB-DBSCAN* (a) and *RNN-DBSCAN* (b) clustering results (maximum ARI solution) for the *flame* dataset. Note that observations colored black were identified as noise by the clustering.

(also, note that the number of noise observation is always zero due to the use of the maximum k nearest neighbor distance as the maximum eps value).

Purity performance is good overall for each clustering approach, indicating that each approach is at least able to identify clusters which consist primarily of observations from a single class. However, such a measure must be considered with respect to the number of identified clusters and noise observations. Hence, the discrepancies in ARI versus Purity performance.

Table 4 shows NMI performance results on the same set of artificial datasets and clustering approaches. Here *RNN-DBSCAN* ranking performance is identical to those discussed with respect to ARI (i.e., ranks first in six of eight datasets and second in the other two).

Table 5 shows the ARI performance, Purity, number of cluster, and number of observation identified as noise for *RNN-DBSCAN* (RNN), *RECORD* (REC), *IS-DBSCAN* (IS), *ISB-DBSCAN* (ISB), *DBSCAN* (DBS), and *OPTICS* (OPT) on the real-world datasets. Here *RNN-DBSCAN* ranks first in three, second in three, and third in one of the datasets. *DBSCAN* edges out *RNN-DBSCAN* ranking first in four of the seven datasets, though ranking no better then third (fourth in one case) in the remaining datasets. Additionally, for two of the dataset *DBSCAN* ranks first in, both methods perform relatively well (*ctg*) or poorly (*htru*).

Similar to results in the artificial datasets, *RECORD* performs the worst, and *ISB-DBSCAN* slightly outperforms its predecessor *IS-DBSCAN*. In the case of *RECORD* and *IS-DBSCAN*, from Table 5 one can again see that both methods have issues with identifying noise observations.

Finally, Table 6 shows NMI performance results on the same set of artificial datasets and clustering approaches.

TABLE 4
NMI Performance on Artificial Datasets

Data	RNN	REC	IS	ISB	DBS	OPT
aggr	**0.996**	0.742	0.888	0.954	0.991	0.969
d31	**0.934**	0.772	0.844	0.879	0.911	0.921
flam	**0.931**	0.54	0.567	0.362	0.869	0.875
jain	0.97	0.376	0.709	**1.0**	0.862	**1.0**
path	**0.872**	0.706	0.735	0.772	0.704	0.686
r15	0.988	0.881	0.871	**0.994**	0.984	0.964
spir	**1.0**	**1.0**	0.917	**1.0**	**1.0**	0.685
grid	**1.0**	0.824	0.983	0.991	0.301	0.991

TABLE 5
ARI Performance on Real-World Datasets

Data		RNN	REC	IS	ISB	DBS	OPT
bank	ari	**0.771**	0.086	0.596	0.594	0.558	0.225
	clu	3	4	2	3	8	35
	pur	0.985	0.828	0.894	0.896	0.896	0.98
	noi	34	580	25	10	1	0
ctg	ari	0.951	0.057	0.883	0.902	**0.992**	0.892
	clu	10	14	6	9	13	17
	pur	1.0	0.372	0.999	0.999	1.0	0.995
	noi	91	796	409	179	5	0
digi	ari	**0.739**	0.011	0.462	0.695	0.684	0.315
	clu	34	3	25	18	21	29
	pur	0.936	0.245	0.957	0.977	0.983	0.733
	noi	104	1524	564	298	355	0
ecol	ari	0.526	0.14	0.474	0.46	**0.639**	0.591
	clu	8	2	4	3	3	5
	pur	0.736	0.538	0.714	0.711	0.582	0.708
	noi	10	89	63	55	100	0
htru	ari	0.334	0.146	0.147	0.166	**0.552**	0.146
	clu	204	56	310	204	4	26
	pur	0.976	0.915	0.981	0.949	0.977	0.976
	noi	236	1909	4206	2270	2289	0
iris	ari	0.644	0.289	0.566	0.568	**0.703**	0.643
	clu	4	2	2	2	7	4
	pur	0.963	0.674	0.671	0.667	0.978	0.847
	noi	16	55	1	0	16	0
seed	ari	**0.617**	0.416	0.383	0.361	0.491	0.498
	clu	4	3	2	9	4	6
	pur	0.898	0.903	0.653	0.888	0.95	0.857
	noi	4	65	34	22	51	0

Again, we see that these results correlate with the ARI observations.

It is important to note that an overall trend with respect to performance is observed across both the artificial and real-world datasets. Specifically, *RNN-DBSCAN* out performs all of the prior reverse nearest neighbor approached while being competitive with *DBSCAN*.

5.1 Approximate k Nearest Neighbor Results

Given the reliance of *RNN-DBSCAN* on k nearest neighbors (kNN), and its corresponding high computational complexity ($O(n^2)$), an approximate solution is investigated below.

The *NN-DESCENT* [11] algorithm produces an approximate kNN solution relying on the principle that a neighbor of a neighbor is also likely a neighbor. Here it is assumed that a kNN approximation can be improved by exploring each observation's neighbors' neighbors as defined by the current kNN approximation. Reusing previous notation, let $N_k(\mathbf{x})$ and $R_k(\mathbf{x})$ define the set of kNNs and reverse nearest neighbors of observation $\mathbf{x}$ given the current kNN approximation. Additionally, let $B(x)$ define the neighborhood of $\mathbf{x}$ as the union of kNNs and reverse nearest neighbors, $B(\mathbf{x}) = N_k(\mathbf{x}) \cup R_k(\mathbf{x})$. The basic implementation of *NN-DESCENT* is described as follows. Beginning with some random kNN approximation, for each observation $\mathbf{x}$, $\mathbf{x}$ is compared with each of its neighbors' neighbors $\mathbf{z}$ such that $\mathbf{z} \in B(\mathbf{y})$ and $\mathbf{y} \in B(\mathbf{x})$. Each comparison, attempts to update the k nearest neighbors of $\mathbf{x}$ with $\mathbf{z}$. This process is repeated over all observations until no updates of the current kNN approximation are found (i.e., no new nearest neighbors are discovered).

In addition to the above basic implementation of $NN - DESCENT$, improvements are also presented in [11] based on local join, incremental search, sampling, and early termination. These improvements reduce the number of comparison and improve data locality with respect to distributed implementations. Two parameters, ρ and δ, are introduced in this improved implementation with ρ defining the neighborhood sampling rate and δ defining early termination (i.e., minimum number of updates). The complexity of each iteration of *NN-DESCENT* is $O(\rho \times n \times k^2)$ with the number of required iterations to convergence being low (empirically observed to be less then 12 iterations in [11]).

Table 7 shows the results of *NN-DESCENT* along with ARI performance of *DBSCAN* using *NN-DESCENT* on several artificial and real-world datasets. *NN-DESCENT* parameters of $k = 100$, $\rho = 0.1$, and $\delta = 0.001$ were used to produce these results. Scan rate is defined as the number of observation comparisons (distance calculations) made relative to the number of comparisons required to compute an exact solutions, $(n \times (n-1))/2$. Recall is the average percentage of correctly identified kNNs in the approximate solution with respect to the exact solution. Similarly, ARI performance is the *RNN-DBSCAN* performance of the approximate kNN solution using the *RNN-DBSCAN* clustering of the exact solution as ground truth.

Reasonable ARI performance at $k = 100$ is observed, with a slight performance decrease observed at $k = 10$. However, recall the approximate solutions where generated for $k = 100$ where better results might be obtained from approximating $k = 10$ directly. In particular, different values of ρ and δ may improve performance. With respect to complexity, scan rates indicate performance at around $n^{1.5}$, though again this might be improved through ρ and δ. Note that scan rate does appear to decrease with respect to the dataset size.

TABLE 6
NMI Performance on Real-World Datasets

Data	RNN	REC	IS	ISB	DBS	OPT
bank	**0.68**	0.213	0.59	0.585	0.579	0.363
ctg	0.934	0.399	0.803	0.886	**0.99**	0.902
digi	**0.824**	0.47	0.648	0.775	0.77	0.67
ecol	0.569	0.538	0.551	0.571	**0.6**	0.55
htru	0.195	0.116	0.117	0.109	**0.25**	0.109
iris	0.683	0.445	0.723	**0.734**	**0.734**	**0.734**
seed	**0.618**	0.48	0.487	0.495	0.533	0.525

TABLE 7
Approximate k Nearest Neighbors Results

Data	Scan Rate	Recall	$\mathbf{ARI}_{k=10}$	$\mathbf{ARI}_{k=100}$
blobs	0.0038	0.996	0.998	0.999
circle	0.0042	0.997	0.973	0.998
moons	0.0042	0.997	0.982	0.998
swissroll	0.0038	0.997	0.982	0.997
farm	0.0013	0.999	0.942	0.982
house	0.0022	0.996	0.978	0.988

6 Related Work

While a complete review of density-based clustering is beyond the scope of this paper, much of the work in this area can be described as building upon *DBSCAN* [1]. In particular, this involves some variant of the original algorithm which addresses some perceived shortcoming of *DBSCAN*. For example, techniques for identifying appropriate value (s) of eps (*OPTICS* [21]), techniques for handling data with heterogeneous density (*OPTICS* [21] and *LDBSCAN* [26]), parameter-free techniques (*APSCAN* [27]), techniques which are more computationally efficient (*APPROX-DBSCAN* [7], [8], *PARTDBSCAN* [28], *MR-DBSCAN* [29], and *MR. SCAN* [30]), techniques for handling high-dimensional data (*NG-DBSCAN* [12] and *MAFIA* [31]), online or streaming techniques (*DENSTREAM* [32]), and techniques aimed at specific domains (*TOSCA* [33]).

One popular technique for improving the computational efficiency of density-based clustering is to combine it with grid-based clustering. Here feature space is partitioned into grids which observations fall into. Density-based clustering is then performed on the grids such that observation are assigned to the cluster of their grid. An example of this is the *DENCLUE* [34] algorithm, which is of further interest due to its use of kernel density estimation. Here the influence of an observation is modeled as a kernel with the overall density of data being calculated as the sum of kernels.

In [35], the *SNN* algorithm is introduced which performs a *DBSCAN*-like clustering using the k nearest neighbor graph. Here edges within said graph are weighted based on a shared nearest neighbor metric, and core/noise observations identified using the sum of an observation edge weights along with some threshold. Clusters are identified by connected components (i.e., connected components of core observations which are extended with non-core observations as discussed throughout Section 2) within a subgraph of the k nearest graph where edges with weight below some threshold are discarded.

HDBSCAN [36] considers the weighted mutual reachability graph where the edge weight between two observations is equal to their mutual reachability distance. Mutual reachability distance between two observations being defined as the maximum distance amongst their actual distance and their two k nearest neighbor distances. Given a Minimum Spanning Tree MST of this graph, a hierarchical clustering is produced by iteratively removing edges from the MST (from largest to smallest weight). In addition to the hierarchical clustering, a method for producing a flat partitioning from the hierarchical solution is introduced.

In *LDBSCAN* [26], core observations are identified using the local outlier factor metric, a measurement of the degree to which an observation is a local outlier. This metric being based on the ratio between an observations reachability distance, and the reachability distances of its k nearest neighbors. This concept is extended into the clustering phase where an observation reachable neighbors is restricted to its set of nearest neighbors whose reachability distance ratios are within some bounds of the observation's reachability distance ratio.

Finally, in *SCAN* [37], a variant of *DBSCAN* is introduced for clustering networks. The main difference between the two algorithms being *SCAN*'s use of structural similarity in defining distances between observations.

7 Conclusions

A novel density-based clustering algorithm, *RNN-DBSCAN*, was presented using reverse nearest neighbor based core observation and observation reachability definitions. The superiority of *RNN-DBSCAN* over prior reverse nearest neighbor approaches, with respect to ARI and NMI performance, was shown using several artificial and real-world datasets. *RNN-DBSCAN* performance was also shown to be comparable to that of *DBSCAN*. This last result is of significance given the reduced problem complexity of *RNN-DBSCAN* with respect to *DBSCAN* (i.e., requires the use of a single parameter, k, as compared with the two parameters, eps and min_{pts}).

Two heuristic for selecting *RNN-DBSCAN*'s parameter k were presented. The validity of these heuristics being demonstrated on several artificial datasets with respect to ARI performance. Additionally, parameter k was shown to be independent of the size of the dataset with respect to large values of n. Again, this is of particular significance with respect to *DBSCAN* as the values of min_{pts} and eps must be appropriately adjusted with respect to n.

A common graph-based interpretation of *RNN-DBSCAN*, the prior reverse nearest neighbor approaches, and *DBSCAN* was presented. Such an interpretation makes it easier to distinguish among the approaches with respect to several key components which include the graph definition, core observation identification, clustering by identifying connected components in some subgraph, and extending clustering results.

Finally, a scalable *RNN-DBSCAN* solution was investigated using an approximate k nearest neighbors method. These results suggest that such an approach could be used to improve the complexity of *RNN-DBSCAN*.

Acknowledgments

This work was funded in part by the Naval Surface Warfare Center Dahlgren Division's In-house Laboratory Independent Research Program.

References

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

[2] S. Vadapalli, S. R. Valluri, and K. Karlapalem, "A simple yet effective data clustering algorithm," in *Proc. 6th Int. Conf. Data Mining*, Dec. 2006, pp. 1108–1112.

[3] C. Cassisi, A. Ferro, R. Giugno, G. Pigola, and A. Pulvirenti, "Enhancing density-based clustering: Parameter reduction and outlier detection," *Inf. Syst.*, vol. 38, no. 3, pp. 317–330, May 2013.

[4] Y. Lv, et al., "An efficient and scalable density-based clustering algorithm for datasets with complex structures," *Neurocomput.*, vol. 171, no. C, pp. 9–22, Jan. 2016.

[5] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger, "The r*-tree: An efficient and robust access method for points and rectangles," *SIGMOD Rec.*, vol. 19, no. 2, pp. 322–331, May 1990.

[6] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, Sep. 1975.

[7] J. Gan and Y. Tao, "Dbscan revisited: Mis-claim, un-fixability, and approximation," in *Proc. ACM SIGMOD Int. Conf. Manag. Data*, 2015, pp. 519–530. [Online]. Available: http://doi.acm.org/10.1145/2723372.2737792

[8] J. Gan and Y. Tao, "On the hardness and approximation of euclidean dbscan," *ACM Trans. Database Syst.*, vol. 42, no. 3, pp. 14:1–14:45, Jul. 2017. [Online]. Available: http://doi.acm.org/10.1145/3083897

[9] A. Beygelzimer, S. Kakade, and J. Langford, "Cover trees for nearest neighbor," in *Proc. 23rd Int. Conf. Mach. Learning*, 2006, pp. 97–104. [Online]. Available: http://doi.acm.org/10.1145/1143844.1143857

[10] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proc. 30th Annu. ACM Symp. Theory Comput.*, 1998, pp. 604–613.

[11] W. Dong, C. Moses, and K. Li, "Efficient k-nearest neighbor graph construction for generic similarity measures," in *Proc. 20th Int. Conf. World Wide Web*, 2011, pp. 577–586. [Online]. Available: http://doi.acm.org/10.1145/1963405.1963487

[12] A. Lulli, M. Dell'Amico, P. Michiardi, and L. Ricci, "Ng-dbscan: Scalable density-based clustering for arbitrary data," *Proc. VLDB Endow.*, vol. 10, no. 3, pp. 157–168, Nov. 2016. [Online]. Available: https://doi.org/10.14778/3021924.3021932

[13] C. Cassisi, R. Giugno, P. Montalto, A. Pulvirenti, M. Aliotta, and A. Cannata, "Dbstrata: A system for density-based and outlier detection based on stratification," 2011. [Online]. Available: http://www.dmi.unict.it/cassisi/DBStrata/

[14] D. Moulavi, P. A Jaskowiak, R. J. G. B. Campello, A. Zimek, and J. Sander, "Density-based clustering validation," in *Proc. SIAM Int. Conf. Data Mining*, 2014, pp. 839–847.

[15] U. of Eastern Finland. Clustering datasets: Shape sets, (2017). [Online]. Available: https://cs.joensuu.fi/sipu/datasets/

[16] F. Pedregosa, et al., "Scikit-learn: Machine learning in Python," *J. Mach. Learning Res.*, vol. 12, pp. 2825–2830, 2011.

[17] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[18] J. Gan and Y. Tao, Approxdbscan datasets, 1999. [Online]. Available: https://sites.google.com/view/approxdbscan/datasets

[19] R. J. Lyon, B. W. Stappers, S. Cooper, J. M. Brooke, and J. D. Knowles, "Fifty years of pulsar candidate selection: From simple filters to a new principled real-time classification approach," *Monthly Notices Roy. Astron. Soc.*, vol. 459, pp. 1104–1123.

[20] M. Varma and A. Zisserman, "Texture classification: are filter banks necessary?" in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2003, pp. II–691–8 vol. 2.

[21] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: Ordering points to identify the clustering structure," *SIGMOD Rec.*, vol. 28, no. 2, pp. 49–60, Jun. 1999.

[22] L. Hubert and P. Arabie, "Comparing partitions," *J. Classification*, vol. 2, no. 1, pp. 193–218, 1985.

[23] A. Strehl and J. Ghosh, "Cluster ensembles — a knowledge reuse framework for combining multiple partitions," *J. Mach. Learn. Res.*, vol. 3, pp. 583–617, Mar. 2003. [Online]. Available: http://dx.doi.org/10.1162/153244303321897735

[24] T. M. Cover and J. A. Thomas, *Elements of Information Theory*. New York, NY, USA: Wiley-Interscience, 1991.

[25] K. S. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is "nearest neighbor" meaningful?" in *Proc. 7th Int. Conf. Database Theory*, 1999, pp. 217–235. [Online]. Available: http://dl.acm.org/citation.cfm?id=645503.656271

[26] 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.

[27] X. Chen, W. Liu, H. Qiu, and J.-H. Lai, "Apscan: A parameter free algorithm for clustering." *Pattern Recognition Lett.*, vol. 32, no. 7, pp. 973–986, 2011.

[28] X. Xu, J. Jäger, and H.-P. Kriegel, "A fast parallel clustering algorithm for large spatial databases," *Data Min. Knowl. Discov.*, vol. 3, no. 3, pp. 263–290, Sep. 1999.

[29] Y. He, et al., "Mr-dbscan: An efficient parallel density-based clustering algorithm using MapReduce," in *Proc. IEEE 17th Int. Conf. Parallel Distrib. Syst.*, Dec. 2011, pp. 473–480.

[30] B. Welton, E. Samanas, and B. P. Miller, "Mr. scan: Extreme scale density-based clustering using a tree-based network of gpgpu nodes," in *Proc. Int. Conf. High Perform. Comput. Netw. Storage Anal.*, 2013, pp. 84:1–84:11. [Online]. Available: http://doi.acm.org/10.1145/2503210.2503262

[31] S. Goil, H. Nagesh, and A. Choudhary, "Mafia: Efficient and scalable subspace clustering for very large data sets," Nothwestern University, Evanston, IL, USA, 1999.

[32] F. Cao, M. Ester, W. Qian, and A. Zhou, "Density-based clustering over an evolving data stream with noise," in *Proc. SIAM Conf. Data Mining*, 2006, pp. 328–339.

[33] R. Guidotti, R. Trasarti, and M. Nanni, "Tosca: Two-steps clustering algorithm for personal locations detection," in *Proc. 23rd SIGSPATIAL Int. Conf. Advances Geographic Inform. Syst.*, 2015, pp. 38:1–38:10. [Online]. Available: http://doi.acm.org/10.1145/2820783.2820818

[34] A. Hinneburg and D. A. Keim, "A general approach to clustering in large databases with noise." *Knowl. Inf. Syst.*, vol. 5, no. 4, pp. 387–415, 2003.

[35] L. Ertoz, M. Steinbach, and V. Kumar, "A new shared nearest neighbor clustering algorithm and its applications," in *Proc. Workshop Clustering High Dimensional Data Appl. 2nd SIAM Int. Conf. Data Mining*, 2002, pp. 105–115.

[36] R. J. G. B. Campello, D. Moulavi, and J. Sander, *Density-Based Clustering Based on Hierarchical Density Estimates*. Berlin, Germany: Springer, 2013, pp. 160–172.

[37] X. Xu, N. Yuruk, Z. Feng, and T. A. J. Schweiger, "Scan: a structural clustering algorithm for networks," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2007, pp. 824–833. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1281192.1281280

Avory Bryant is a software engineer in the Naval Surface Warfare Center Dahlgren Division, and working toward the PhD degree in the Computer Science Department at Virginia Commonwealth University. His interest include data clustering, information retrieval, and visual analytics.

Krzysztof Cios received the PhD, DSc, and MBA degrees. He is a professor and chair of the Computer Science Department at the Virginia Commonwealth University, Virginia. His research interests include the areas of machine learning, data science, and applications in medicine and biology. His research has been funded by NIH, NASA, NSF, NATO, and the US Air Force. He served as primary advisor to 15 doctoral students who are now university professors in the US, Australia, Vietnam, and Thailand, or work at companies such as Google, NASA, General Motors, and Proctor and Gamble. He published three books and about 200 journal and conference papers. He has been the recipient of the Norbert Wiener Outstanding Paper Award, the Neurocomputing Best Paper Award, and the Fulbright Senior Scholar Award. He is a foreign member of the Polish Academy of Arts and Sciences (PAU), and a fellow of the American Institute for Medical and Biological Engineering (AIMBE).