k-Nearest Neighbor based Clustering with Shape Alternation Adaptivity

Yifeng Lu, Yao Zhang, Florian Richter, Thomas Seidl
Database Systems and Data Mining Group
LMU Munich, Germany
Email: lu@dbs.ifi.lmu.de, yao.zhang@campus.lmu.de, richter@dbs.ifi.lmu.de, seidl@dbs.ifi.lmu.de

***Abstract*—Existing clustering algorithms aim at identifying clusters from a single dataset. However, many applications generate a series of datasets. For example, scientists need to repeat an experiment many times to ensure reproducibility; sensors collect information day after day. In such scenarios, we need to identify clusters separately from a large number of datasets, which can contain an unknown number of clusters with various densities and shapes.**

Density-based clustering algorithms are commonly used in identifying arbitrary shaped clusters when the cluster number is unknown. Most density-based clustering algorithms are "DBSCAN-alike", where clusters are formed by connecting consecutive high dense regions. Therefore, points are grouped as one cluster as long as they are densely connected. When the distribution shape of points is changed across different datasets, parameter tuning on each dataset is necessary to obtain proper results, which is time-consuming.

In this work, we developed a new kNN density-based clustering algorithm, which does not adopt the DBSCAN paradigm. Instead, we identify clusters by maximizing the intra-cluster similarities, which are estimated using: 1) the probability that two points belong to the same cluster; 2) the probability that a point is a cluster center. The kNN concept and minimum spanning tree are used to compute both probabilities. Our approach is capable of extracting clusters in arbitrary shapes using the single parameter k, and can handle a series of datasets with less parameter tuning effort. Experiments on both synthetic and real-world datasets show that our approach outperforms other recent kNN clustering algorithms.

I. Introduction

Identifying clusters in arbitrary shapes is useful in many applications, such as spatial data or scientific data clustering. Usually, the number of clusters is unknown in those applications, so that density-based clustering algorithms, such as DBSCAN [1], are very popular. In recent years, novel methods, such as ISB-DBSCAN [2] and RNNDBSCAN [3], are proposed, which are "DBSCAN-alike" with the concept of k-nearest neighbor (kNN) employed to define density. The ability to identify arbitrary shaped clusters is kept. More importantly, only a single parameter k is required, which can be roughly estimated using the minimum number of points required to form a cluster. Moreover, they also handle clusters in various densities better.

Clusters in those algorithms are formed by connecting consecutive highly dense regions whose local density exceeds a given threshold, which is defined using kNNs. Thus, the connectivity is purely determined by local density. The distribution shape of data points on a global scale is not considered. However, in some applications such as scientific research, a series of datasets are generated at once, waiting for clustering. Points distribution shape in those datasets may alter, for instance, from a strip shape to a dumbbell shape, which implies cluster changing. However, existing DBSCAN-alike approaches can not adapt automatically when distribution shapes alter across different datasets. A painful parameter tuning process is necessary for each dataset.

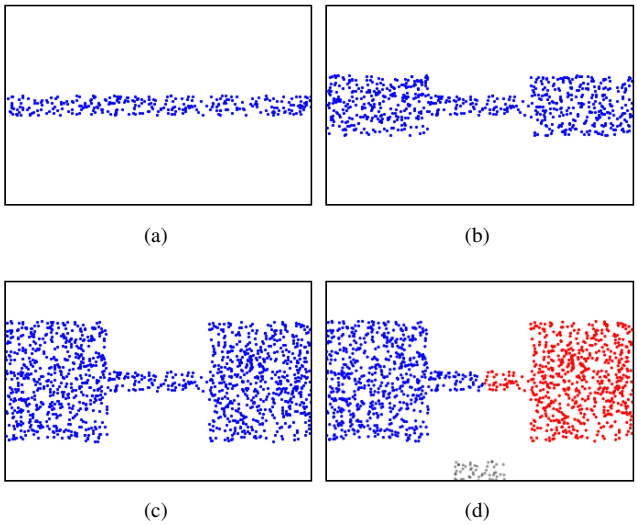


Fig. 1: RNNDBSCAN correctly detects one cluster in (a) but failed to find two clusters properly in (b) and (c), when parameters are fixed. Even if we set parameters to find two clusters, the small cluster in (d) is incorrectly labeled as noise.

Fig. 1(a-c) illustrate three datasets with data points distributed in different shapes but similar densities. Clusters are identified using a DBSCAN-alike approach (RNNDBSCAN [3]) under the same parameter. In the first dataset (a), it is safe to say that the result is correct as all points are in the same strip-shaped cluster. When the number of points increased on the two ends (b,c), it is likely that there are two clusters. However, RNNDBSCAN can not adapt automatically and still return a single cluster. Of course, we can also choose the parameters to return two clusters in Fig. 1(b,c), but then the single strip-shaped cluster in (a) will be separated. Therefore, users always need to manually adjust the parameter on each

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

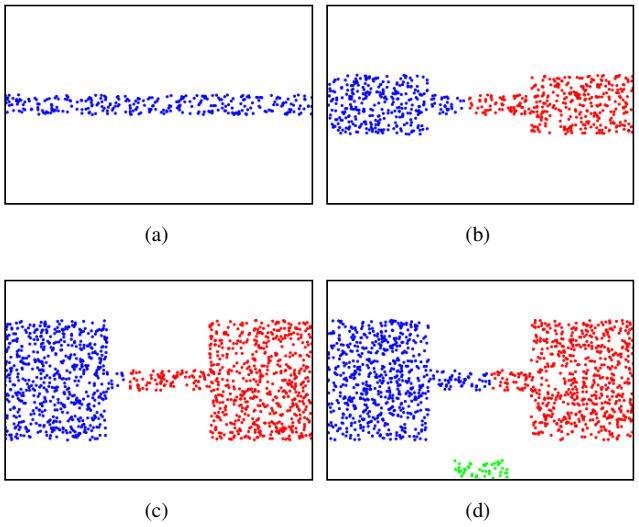


Fig. 2: Clustering results of our approach. The parameter k is set to work properly on (d). No further parameter tuning is applied to the rest of datasets.

dataset to generate the correct result, which is infeasible for applications where hundreds of datasets are waiting for clustering.

In some cases, lacking the awareness of the overall data points distribution also makes parameter tuning more difficult, even on a single dataset. For example, in Fig. 1d, the small cluster (gray) is identified as noise if we set parameters to detect the two big clusters properly since the density threshold is too large.

In this work, a new kNN clustering algorithm is proposed to tackle this problem by avoiding the DBSCAN-alike structure. It can handle arbitrary shaped clusters using a single parameter k. It is also aware of the overall data distribution so that clusters can be identified and separated even if it is densely connected to other clusters, without parameter tuning. We estimate similarities using the concept of kNN to reflect the probability that two points belonging to the same cluster and the probability that a point is a cluster center. The Affinity Propagation [4] algorithm is applied to extract cluster centers.

Fig. 2 illustrates the clustering result of our approach on toy examples given above. The parameter k is selected so that three clusters are detected in Fig. 2d. Without manual parameter tuning for the rest of the datasets, our approach can adapt itself to identify clusters properly.

II. Related Works

The concept of k-nearest neighbor is widely used in density-based clustering approaches. For example, the relationship between the connectivity of a mutual k-nearest neighbor graph and the clustering structure is studied in [5]. The SNN [6] algorithm uses the kNN to handle clusters with various densities. LDBSCAN [7] employs the local outlier factor as the metric, which is also defined using kNN. However, extra parameters are required in those approaches.

Our work is a single parameter kNN clustering algorithm. Most reported single parameter kNN clustering algorithms borrowed the idea of DBSCAN [1]. The RECORD [8] algorithm makes use of kNN graph and reverse kNN graph to define core points. Clusters are extracted from the subgraph of core observations. HDBSCAN [9] builds a minimum spanning tree on a mutual reachability graph. Edges are iteratively removed to generate optimized clusters. The IS-DBSCAN [10] approach introduced the concept of influence space of a data point, which is defined as the intersect between its reverse and k nearest neighbor sets. The influence space concept is then used to describe local density and reachability of data points. ISB-DBSCAN [2] goes a step further by using an undirected influence space graph. In RNN-DBSCAN [3], density reachability is defined by only using the concept of k-nearest neighbor and reverse k-nearest neighbor. A data point is a core point if the number of its reverse k-nearest neighbor is larger than k. KNNCLUST [11] does not follow a DBSCAN clustering style. Instead, it starts clustering by assigning different cluster labels to each data point. The cluster label is then updated recursively by computing a posterior probability concerning labels in kNN. Unfortunately, all methods above can not adapt automatically to the drift in cluster shapes. The parameter k needs to be determined separately for a series of datasets.

Of course, there are algorithms such as spectral clustering [12] that can handle cluster shape alteration. However, they need to know the cluster number in advance.

III. Preliminaries

A good clustering should have a high intra-cluster similarity. To identify cluster properly, we need a good estimation of similarities, and an algorithm that maximizes the intra-cluster similarity. Affinity Propagation (AP) [4] is designed to identify clusters that maximizing the intra-cluster similarity based on a similarity matrix and a preference vector. As suggested in AP, the similarity matrix and the preference vector should reflect the probability that two points belong to the same cluster and the probability that a point is a cluster center. Thus, the major challenge of this work is how to estimate both probabilities properly.

In this work, we use the kNN distance to measure those two probabilities. The heuristic of cluster centers proposed is DPC [13] is also employed in computing the preference vector. Those two probabilities are utilized as the similarity matrix and the preference vector for AP to generate cluster centers, which eventually produces clusters. Only one parameter k is employed in the whole process.

A. Affinity Propagation

Affinity Propagation does not require the user to estimate the number of clusters or the density of points. Instead, the user has to provide a similarity matrix S in which s_{ij} is the similarity between point i and j. The diagonal of S is the

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

preference vector. s_{ii} is the *preference* of point i, denoted as pref_i. The similarity measures how likely that two points belong to the same cluster. The preference value reflects the likelihood that a point being a cluster center. Cluster centers and cluster assignments are determined by maximizing the overall intra-cluster similarity:

$$\mathcal{S} = \sum_{i=1}^{N} S_{i,e_i} \tag{1}$$

where e_i is the cluster center of point i. Obviously, the overall distribution of data points influences the final clustering result.

AP is reported to work well on certain tasks such as computational biology, where the similarity between observations is well defined. In a more general case, the similarity between points is set to the *negative* Euclidean distance. Preferences of all points are initialized to the same value, such as the minimum or the median of similarities. However, it is difficult to achieve the desired results with the default setting of AP. Figure 3 illustrates the clustering results on two datasets. Although the data only contains 2/3 simple clusters, AP does not identify these clusters successfully due to shapes and cluster proximity.

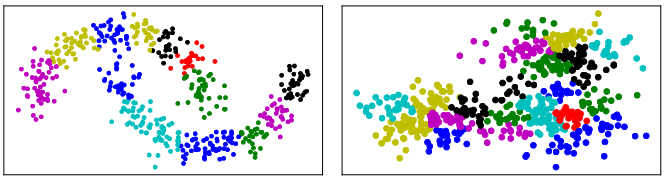


Fig. 3: Clustering results of AP under default settings.

B. Density Peaks Clustering

Density Peaks Clustering (DPC) is a semi-automated clustering approach, which makes use of two heuristics to highlight cluster centers: 1) *cluster centers are surrounded by neighbors with lower density*; 2) *cluster centers are far away from other points with a higher density*. A decision graph is derived by computing the local density ρ_i and the distance to the nearest point with a higher density δ_i (*delta distance*) for each point. Then the user identifies cluster centers by selecting points manually with both large ρ and large δ.

The DPC algorithm shows outstanding performance in a variety of clustering tasks by providing a great visualization tool for identifying cluster centers manually. There are some algorithms proposed in recent years to fully automate the cluster center selection process by analyzing the values of ρ and δ. Indeed, the original DPC algorithm also proposes to examine the product of ρ and δ for each data point. However, those approaches introduce additional parameters, which need to be precisely selected [14].

Nonetheless, the heuristic introduced by DPC is useful for our approach, which provides a good description of cluster centers. We apply it to compute the probability of a point being a cluster center, i.e., the preference of points.

IV. PROPOSED ALGORITHM

There are four major steps in our approach: 1) estimating the similarity matrix; 2) estimating the preference vector; 3) using affinity propagation to identify cluster centers; 4) assigning labels to the rest of data points.

A. Similarity Estimation

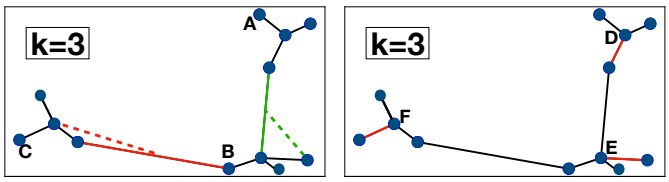
As mentioned above, similarities used by AP should reflect how likely that two points belong to the same cluster. To achieve the goal, we adopt the idea of minimax distance on graph. Minimax distance can model the underlying structures and the transitive relations nonparametrically [15]. Moreover, minimax distance on a graph is equivalent to longest edge on the path of the corresponding minimum spanning tree (MST). In this work, we built MST on our dataset. Instead of using edge length, the *minimum edge density* on the path from point i to j is used to measure similarity. Edge density of an edge e is represented using the kNN distance of the center point of e in the dataset.

For simplicity, the MST is built based on the Euclidean distance between points, known as the Euclidean Minimum Spanning Tree (EMST). Other distance function can also be used but beyond the scope of this paper.

Let T be the EMST built on our dataset, T_{ij} be the path from point i to point j, which is formed by a list of adjacent edges $e_{ii_1}, e_{i_1i_2}, \ldots, e_{i_mj}$ of T. The distance between two points i, j is:

$$dist_{ij} = \max_{e \in T_{ij}} d_k(e) \tag{2}$$

where $d_k(e)$ of edge e is the kNN distance of the center point of e with respect to points in the dataset.



(a) (b)

Fig. 4: Similarity and Preference on toy MST example.

Intuitively, we measure how sparse it is on the path from point i to j. If data points stay densely around all edges on the path, then $dist_{i,j}$ is small, and i, j tend to come from the same cluster. Figure 4a shows a toy example. Dashed lines (red and green) represent the kNN distance of corresponding edges, i.e., the edge density. Thus, the point B is closer to A than C. As AP asks for negative similarity values, we normalize our distance and take the negative as the similarity value:

$$s_{ij} = -\frac{dist_{ij} - dist_{min}}{dist_{max} - dist_{min}} \tag{3}$$

where $dist_{min} = \min_{i \neq j}(dist_{ij})$, $dist_{max} = \max_{i \neq j}(dist_{ij})$

Algorithm 1 illustrates procedure for calculating the similarity matrix.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

Algorithm 1 Similarity(X, k)

$T \leftarrow$ `MinimumSpanningTree(X)`
for $x_i \in X$ **do**
  for $x_j \in X, j \neq i$ **do**
    $dist_{ij} = \max_{e \in T_{ij}} d_k(e)$
for $s_{ij} \in S, i \neq j$ **do**
  $s_{ij} \leftarrow -\frac{dist_{ij} - dist_{min}}{dist_{max} - dist_{min}}$
return S;

Algorithm 2 Preference(X, k, $dist$)

$\rho_i \leftarrow$ local density of $x_i \in X$
$j \leftarrow \arg\max_i \rho_i$
for $x_i \in X$, $i \neq j$ **do**
  $\delta_i \leftarrow$delta distance of x_i
  $\text{pref}_i \leftarrow (\delta_i - dist_{max}) \cdot \rho_i$
$\text{pref}_j \leftarrow \max_{i \neq j}(\text{pref}_i)$
$\text{pref} \leftarrow -\frac{\text{pref}}{\max_{\text{pref} \neq 0} \text{pref}}$
return pref

B. Preference Estimation

The preference value of a data point implies the likelihood of being a cluster center. According to the heuristic of DPC, the preference value of a data point i is positively correlated with two properties:

1) ρ_i, the local density of point i,
2) δ_i, the delta distance of point i (distance to a point with larger ρ).

As DPC suggests to analyzing the production of ρ and δ for automatic cluster center detection, we can assume $\text{pref}_i \propto \delta_i \cdot \rho_i$.

AP requires both similarity and preference values to be negative values. A cluster center point with large ρ_i and δ_i should have a negative preference value close to 0. Preference values of off-center points must be much smaller than 0. In consequence, we define the preference of a point i as:

$$\text{pref}_i = (\delta_i - dist_{max}) \cdot \rho_i \tag{4}$$

where the local density ρ_i is defined as the kNN distance of point i, which is similar to the edge density $d_k(e)$ defined above.

In Figure 4b, the length of the red line is the local density of points D, E, and F. δ_i and $dist_{max}$ are defined using distance values in equation 2. Under such definition, boundary points may also have high preference values (close to 0) when $\delta_i - dist_{max} \approx 0$. AP might identify those points as cluster centers of small clusters. However, their cluster size are smaller than k so that we can identify and correct those "outlier clusters" easily.

Furthermore, pref_j is not defined in equation 4 if $\rho_j = \max(\rho)$, since δ_j is not defined. Thus, we let $\text{pref}_j = \max_{i \neq j}(\text{pref}_i)$ as j is very likely to be a cluster center. Moreover, we need to normalize the preference value by dividing by the maximum nonzero preference value:

$$\text{pref}_i = -\frac{\text{pref}_i}{\max_{\text{pref} \neq 0} \text{pref}} \tag{5}$$

Such normalization reflects how likely a point i is to be a cluster center, when compared with the most possible one. Algorithm 2 presents the procedure for calculating the preference vector.

C. Generating Clusters

After estimating similarities and preferences, we add the preference vector **pref** into the diagonal of the similarity matrix S and let AP determine cluster centers. As AP tends to maximize the overall intra-cluster similarity, the global distribution of data points is also taken into consideration when identifying cluster centers.

Cluster labels of the rest of the points are assigned in a similar way to DPC. We traverse unlabeled points in descending order according to the local densities. The label of the nearest cluster is attached. A refinement step is introduced by comparing the label of each point to its k nearest neighbors. Labels will be updated to the majority label among kNN. Such a refinement step is useful since AP may generate "outlier clusters", as mentioned above. Algorithm 3 illustrates the overall procedure of our approach.

Algorithm 3 Generating Clusters

Require: Dataset X, Parameter k
Ensure: Cluster labels $l = [l_1, ..., l_N]^T$;
$N \times N$ similarity matrix: $s \leftarrow$ Similarity(X, k)
$N \times 1$ preference vector: pref $\leftarrow$ Preference($X, k, -s$)
$diag(s) \leftarrow$ pref
$\forall x_i \in X, l_i \leftarrow -1$ ▷ initial cluster label to -1
Exemplars $E \leftarrow$ `AffinityPropagation(s)`
for $x_i \in E$ **do**
  $l_i \leftarrow$ A unique cluster label
Sort X in descending order of local density ρ
for all $x_i \in X, l_i = -1$ **do**
  $l_i = \underset{(l_j \neq -1)}{\arg\max}(s_{i,j})$
for all $x_i \in X$ **do**
  $l_i \leftarrow$ majority label among kNN of x_i ▷ refinement
return c;

D. Computational Complexity

Our approach contains four primary steps: 1) EMST generation; 2) Similarity and Preference estimation; 3) AP for cluster centers generation and 4) Cluster labels assignment. The complexity of the first step, generating EMST, can achieve $O(N \log N)$ by utilizing index structure and the Prim's algorithm [16]. Similarity and preference computation involves the EMST traversing and the similarity matrix filling, which takes $O(N^2)$. The time complexity of Affinity Propagation makes $O(N^2 T)$, where N is the number of data points, T is the number of iterations. The last cluster assignment step is mainly

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

about nearest neighbor search, which takes $O(N \log N)$ in total. In summary, our k-nearest neighbor density-based clustering approach has a complexity of $O(N^2T)$.

V. EXPERIMENTAL RESULTS

We investigate our approach on both synthetic and real-world datasets from the UCI Machine Learning Repository [17]. Additionally, several artificial datasets of varying sizes, densities, and shapes were generated to highlight the effectivity of our approach. A summary of each dataset is provided in Table I.

TABLE I: Dataset Statistics

(a) Synthetic Datasets

Data	N	N_c	d
spiral [17]	312	3	2
aggregation [17]	788	7	2
flame [17]	240	2	2
d31 [17]	3100	31	2
moon (Fig 5a)	600	2	2
gaussian (Fig 5b)	800	2	2
blobs (Fig 5c)	1K, 5K, 10K	2	2
strip (Fig 5d)	1K, 5K, 10K	2	2

(b) Real-world Datasets

Data	N	N_c	d
iris [17]	150	3	4
digits [17]	1797	10	64
seeds [17]	210	3	7
segments [17]	2310	7	19
seismic-bumps [17]	210	3	8
satimage [17]	6430	6	37
banknote [17]	1372	2	4

Affinity Propagation algorithm under default settings (as described in Section III-A) is conducted. RNNDBSCAN algorithm is included to represent recent DBSCAN-alike kNN clustering algorithms. KNNCLUST stands for the performance of kNN clustering approaches that are not DBSCAN-alike. Furthermore, two density-based algorithms, DBSCAN and OPTICS, and a manifold-based algorithm, Spectral Clustering, are also conducted as a baseline.

Adjusted Rand Index (ARI) [18] and Normalized Mutual Information (NMI) [19] are reported. For each kNN based algorithm, we vary k from 1 to 100 and report the result of k with the best ARI score. Clusters generated by KNNCLUST are inconsistent across multiple runs on the same dataset due to its random access to data points. Thus, the average score of several runs is reported while the best run is used for visualization. For DBSCAN, min_{pts} is selected from the set $\{2, 5, 10, 20\}$, and eps is selected over a set of values equal to the min_{pts} nearest neighbor distance of each observation. OPTICS has the same parameter pool as DBSCAN, while the schema mentioned in [20] is used for cluster generation. The affinity matrix we used in Spectral Clustering (SC) is constructed using the kNN method, where k also varies from 1 to 100. Table II and III show the ARI and NMI score on synthetic and real datasets. The best values in each row are marked in bold.

TABLE II: ARI & NMI Score on Synthetic Datasets

Dataset		**Our**	**AP**	**RNN**	**KNN**	**DBS**	**OPT**	**SC**
spir	ari	**1**	0.101	0.179	**1**	**1**	0.162	0.388
	nmi	**1**	0.538	0.454	**1**	**1**	0.395	0.466
aggr	ari	**0.996**	0.177	0.991	0.809	0.992	0.984	0.809
	nmi	**0.988**	0.681	0.987	0.895	0.98	0.977	0.895
flame	ari	**0.97**	0.086	0.949	0.208	**0.97**	0.967	0.650
	nmi	**0.93**	0.483	0.891	0.517	**0.93**	0.927	0.741
d31	ari	**0.954**	0.529	0.855	0.457	0.884	0.641	0.943
	nmi	**0.97**	0.854	0.917	0.081	0.927	0.879	0.962
moon	ari	**0.984**	0.076	0.976	0.485	0.980	0.976	0.802
	nmi	**0.97**	0.458	0.946	0.379	0.970	0.951	0.754
gaus	ari	0.962	0.042	0.183	0.544	0.526	0.034	**0.98**
	nmi	0.931	0.409	0.156	0.554	0.501	0.195	**0.97**
blobs	ari	**0.941**	0.093	0.511	0.245	0.765	0.469	0.921
	nmi	**0.908**	0.508	0.569	0.493	0.717	0.619	0.887
strip	ari	**0.783**	0.097	0.530	0.355	0.750	0.621	0.709
	nmi	**0.780**	0.487	0.544	0.477	0.723	0.626	0.697
Average	ari	**0.949**	0.150	0.647	0.443	0.791	0.607	0.672
	nmi	**0.935**	0.552	0.640	0.485	0.781	0.696	0.711

Our approach has the best ARI and NMI scores in almost all synthetic datasets. It is only defeated by the Spectral Clustering on the `gaus` dataset with a narrow margin. On real-world datasets, our method is also very competitive. It has the best ARI score in 4 out of 7 datasets. It also ranks second or third place in the rest of the datasets. In terms of NMI score, our approach also outperforms competitors in real datasets. It has the highest NMI score in 5 real datasets and takes the second place in `sati` dataset with a marginal gap. `bank` dataset is the only one that our approach is not among top-2, but still better than AP, KNNCLUST, and OPTICS. In summary, our approach provides solid clustering quality compared with traditional methods, as well as novel kNN based methods.

Fig. 5 visualizes the clustering result of our approach and other kNN clustering algorithms on a series synthetic datasets. Our approach provides the best clustering result, especially in Fig. 5b/c, where two or three clusters are mutually overlapped. Our approach can identify clusters from densely connected data, while other kNN methods fail.

The last column of Fig. 5 compares the clustering quality (ARI score) of our approach with other algorithms. The available range of k with a high ARI score is much more extensive than competitors, which means that the user can estimate the parameter roughly. Indeed, accepting roughly estimated parameters is an essential feature of kNN clustering algorithms since users do not need to search for the best parameter value accurately as $k$means or spectral clustering required, which is time-saving.

More importantly, the more extensive available range of k also means that our approach can handle a series of datasets without parameter tuning on each dataset. In this example, our

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

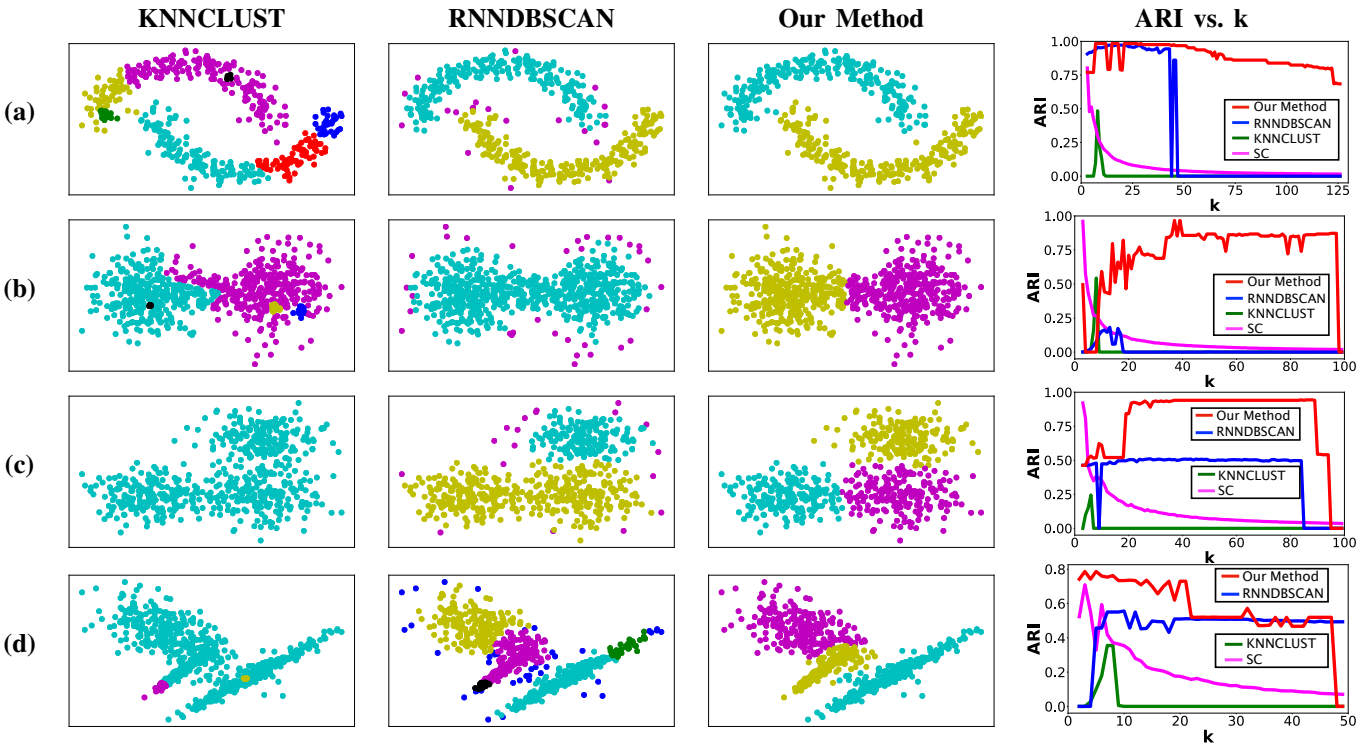


Fig. 5: Visualization of kNN clustering approaches on synthetic datasets and the ARI score under different k values.

TABLE III: ARI & NMI Score on Real-World Datasets

Dataset		Our	AP	RNN	KNN	DBS	OPT	SC
iris	ari	**0.882**	0.344	0.548	0.562	0.739	0.578	0.767
	nmi	**0.867**	0.633	0.688	0.664	0.722	0.731	0.811
digi	ari	**0.799**	0.126	0.574	0.783	0.677	0.116	0.756
	nmi	**0.854**	0.667	0.742	0.848	0.805	0.495	**0.854**
seed	ari	**0.753**	0.177	0.534	0.434	0.582	0.470	0.602
	nmi	**0.712**	0.510	0.562	0.543	0.554	0.550	0.629
segm	ari	0.428	0.080	**0.500**	0.116	0.401	0.175	0.473
	nmi	**0.699**	0.544	0.622	0.570	0.610	0.531	0.651
seis	ari	**0.738**	0.186	0.436	0.477	0.499	0.558	0.616
	nmi	**0.711**	0.526	0.555	0.565	0.531	0.618	0.644
sati	ari	0.529	0.169	0.432	0.389	0.389	0.090	**0.550**
	nmi	0.628	0.540	0.575	0.537	0.551	0.364	**0.656**
bank	ari	0.516	0.027	**0.763**	0.029	0.668	0.342	0.531
	nmi	0.569	0.394	**0.707**	0.391	0.638	0.439	0.451
Average	ari	**0.664**	0.158	0.541	0.399	0.565	0.333	0.614
	nmi	**0.720**	0.545	0.636	0.588	0.630	0.533	0.671

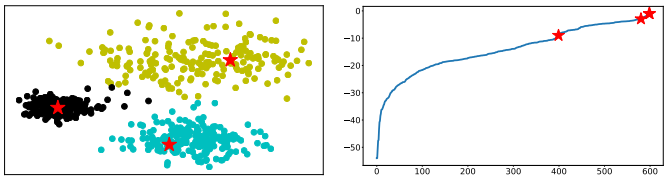
method can identify all clusters properly by fixing k at around 30. In contrast, KNNCLUST is able to find clusters correctly only if the parameter is accurately selected. Spectral clustering works only if the number of clusters is chosen appropriately. RNNDBSCAN has a wider parameter range, but parameter tuning is still necessary across different datasets.

VI. Further Properties and Discussions

A. Automatic cluster centers detection for DPC

Many approaches are proposed to detect cluster centers for DPC automatically by proposing new density functions [21] or analyzing the scalar value, such as $\rho_i \cdot \delta_i$, of each point. Points with properties larger than a threshold are returned as cluster centers.

Our approach is different since we keep the information of both ρ and δ, and use AP to find cluster centers that maximize intra-cluster similarity. Thus, cluster centers are determined not only by the properties of each point, but also by the relationship between points. Fig. 6 illustrates cluster centers found by our approach and their corresponding preference values. We can see that three centers are not those points with the highest preference value.



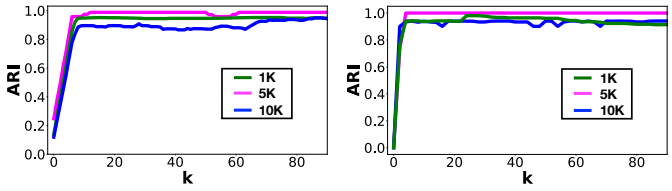
(a) Proposed Algo. (b) Preference Plot

Fig. 6: Cluster centers (red stars) and their preference values in the dataset.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

B. Various Dataset Size

Despite less complexity in parameter selection, another essential property of kNN clustering over DBSCAN is the dataset size invariant [3]. Fig. 7 shows the clustering qualities of our approach on two datasets with different dataset sizes. We can see that, although the DBSCAN structure is not used, the quality does not differ a lot under different sizes.



(a) Dataset *blobs* (b) Dataset *strip*

Fig. 7: ARI Performance vs k on *blobs* and *strip* dataset with different sizes.

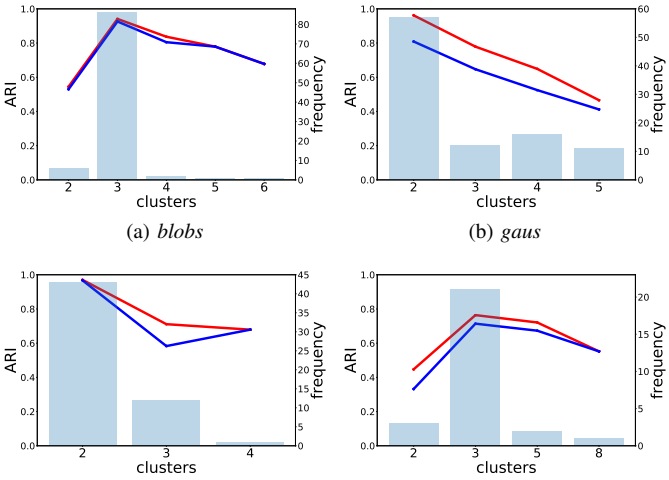
C. Cluster Number and Parameter k Estimation

There is no ground truth in the real-word for unsupervised learning. Thus, even if we can try different settings, we do not know which one is the best. Some unsupervised cluster quality measures, such as the silhouette coefficient [22], are introduced to address the problem. However, those methods can only be applied to model-based algorithms such as $k$means, and it is challenging to estimate the qualities of arbitrary shaped clusters.

Note that the quality of our kNN clustering approaches is relatively high on a wide range of parameters. Such property can be used to estimate cluster numbers and the best parameter setting in the real-world [3]. Our approach also keeps such benefits. Fig. 8 illustrates the histogram of cluster number been identified with respect to the value of parameter k for $k \in [0, 100]$. Red and blue lines are the corresponding best and worst ARI scores in each cluster number. As shown in Fig. 8, for $k \in [1, 100]$, the number of clusters concentrates to a few values. Thus, the real cluster number can be estimated as the most frequent cluster number, and the value of k that leads to the most frequent cluster number can be considered as a good setting.

VII. CONCLUSION

In this work, we present a novel kNN based clustering algorithm. Although we are not DBSCAN-alike, we still keep the ability of arbitrary shape clustering using the only parameter k. Similarities and preferences of points are measured as probabilities based on heuristics from DPC and kNN distance. Affinity Propagation algorithm is employed to maximize the overall intra-cluster similarity. In consequence, our kNN based clustering approach is also aware of the overall distribution in the dataset. In cases where we need to cluster a series of datasets with similar density but different shapes, our approach is particularly useful since it can adapt to changes in shapes automatically. Furthermore, with all those functional benefits, our approach still provides a reliable clustering quality.



(a) *blobs* (b) *gaus*

(c) *flame* (d) *seeds*

Fig. 8: Histogram of cluster number identified by our approach, $k \in [1, 100]$. The best (red) and the worst (blue) ARI score are illustrated for each case.

REFERENCES

[1] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96. AAAI Press, 1996, pp. 226–231.

[2] Y. Lv, T. Ma, M. Tang, J. Cao, Y. Tian, A. Al-Dhelaan, and M. Al-Rodhaan, "An efficient and scalable density-based clustering algorithm for datasets with complex structures," *Neurocomputing*, vol. 171, pp. 9–22, 2016.

[3] A. Bryant and K. Cios, "Rnn-dbscan: A density-based clustering algorithm using reverse nearest neighbor density estimates," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 6, pp. 1109–1121, June 2018.

[4] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *science*, vol. 315, no. 5814, pp. 972–976, 2007.

[5] M. Brito, E. Chavez, A. Quiroz, and J. Yukich, "Connectivity of the mutual k-nearest-neighbor graph in clustering and outlier detection," *Statistics & Probability Letters*, vol. 35, no. 1, pp. 33–42, 1997.

[6] L. Ertoz, M. Steinbach, and V. Kumar, "A new shared nearest neighbor clustering algorithm and its applications," in *Workshop on clustering high dimensional data and its applications at 2nd SDM*, 2002, pp. 105–115.

[7] L. Duan, L. Xu, F. Guo, J. Lee, and B. Yan, "A local-density based spatial clustering algorithm with noise," *Information systems*, vol. 32, no. 7, pp. 978–986, 2007.

[8] S. Vadapalli, S. R. Valluri, and K. Karlapalem, "A simple yet effective data clustering algorithm," in *Sixth International Conference on Data Mining (ICDM'06)*. IEEE, 2006, pp. 1108–1112.

[9] R. J. Campello, D. Moulavi, and J. Sander, "Density-based clustering based on hierarchical density estimates," in *Pacific-Asia conference on knowledge discovery and data mining*. Springer, 2013, pp. 160–172.

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

[11] T. N. Tran, R. Wehrens, and L. M. Buydens, "Knn-kernel density-based clustering for high-dimensional multivariate data," *Computational Statistics & Data Analysis*, vol. 51, no. 2, pp. 513–525, 2006.

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

[13] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

[14] H. Yan, Y. Lu, H. Ma *et al.*, "Density-based clustering using automatic density peak detection." in *ICPRAM*, 2018, pp. 95–102.
[15] M. H. Chehreghani, "Efficient computation of pairwise minimax distance measures," in *2017 IEEE International Conference on Data Mining (ICDM)*. IEEE, 2017, pp. 799–804.
[16] Bentley and Friedman, "Fast algorithms for constructing minimal spanning trees in coordinate spaces," *IEEE Transactions on Computers*, vol. C-27, no. 2, pp. 97–105, Feb 1978.
[17] D. Dua and C. Graff, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml
[18] L. Hubert and P. Arabie, "Comparing partitions," *Journal of classification*, vol. 2, no. 1, pp. 193–218, 1985.
[19] T. O. Kvalseth, "Entropy and correlation: Some comments," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 17, no. 3, pp. 517–519, 1987.
[20] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, "Optics: Ordering points to identify the clustering structure," in *Proceedings of the ACM SIGMOD International Conference on Management of Data*. ACM, 1999, pp. 49–60.
[21] Z. Guo, T. Huang, Z. Cai, and W. Zhu, "A new local density for density peak clustering," in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 2018, pp. 426–438.
[22] P. J. Rousseeuw, "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis," *Journal of computational and applied mathematics*, vol. 20, pp. 53–65, 1987.

978-1-7281-6926-2/20/$31.00 ©2020 IEEE