High Dimensional Clustering: A Strongly Connected Component Clustering Solution (SCCC)

by

mihir.shekhar@research.iiit.ac.in , Lini Thomas, Kamalakar Karlapalem

in

IEEE International Conference on Data Mining
(*ICDM-2018*)

Singapore

Report No: IIIT/TR/2018/-1

Centre for Data Engineering
International Institute of Information Technology
Hyderabad - 500 032, INDIA
November 2018

High Dimensional Clustering: A Strongly Connected Component Clustering Solution (SCCC)

Mihir Shekhar
Data Science and Analytics Center
IIIT Hyderabad, India
Email: mihir.shekhar@research.iiit.ac.in

Lini Thomas
Data Science and Analytics Center
IIIT Hyderabad, India
Email: lini.thomas@iiit.ac.in

Kamalakar Karlapalem
Data Science and Analytics Center
IIIT Hyderabad, India
Email: kamal@iiit.ac.in

***Abstract*—High dimensional data is often challenging to cluster due to the curse of dimensionality leading to challenges in identifying clusters. The key challenge in high dimensional clustering is to develop a solution that identifies clusters which are as complete as they can be, while not merging well-separated clusters. We propose core points which represent local compact regions. The strongly connected component from the k-nearest neighbor graph of core points provides for a group of points that are strongly mutually connected. These mutually connected regions represent the core structure of the clusters. Our empirical analysis and experimental results present the rationale behind our solution and validate the goodness of the clusters against the state of the art high dimensional clustering algorithms. The novelty of our solution is to use the concept of reverse nearest neighbors to generate natural clusters in high dimensions.**

***Index Terms*—clustering**

I. Introduction

High dimensional clustering is prevalent for datasets relevant to various applications like multimedia, sound, image, text etc. A clustering algorithm performs an unsupervised learning by which it groups points such that the intra point similarity is maximised within a cluster and intra point similarity across clusters is minimised. The onus of getting clusters is based on the similarity among points which typically is defined by the distance measure (example: Euclidean, Cosine, Manhattan) between points. Below we discuss two important challenges for high dimensional clustering which we have addressed in this paper.

1. Difficulty in parameter Estimation: It is difficult to identify an optimal parameter value in clustering as the data is unlabelled. Furthermore, the presence of multiple parameters in the algorithm makes this problem complex. For example, setting number of clusters parameter in k-means [1] algorithm is a difficult problem for higher dimensional data.

Another issue in parameter selection is that confidence on the parameter values picked is low, when the clustering solution shows significant changes in the resultant clusters with small changes in the parameter values. For example: DBSCAN [2] has two parameters 1) the radius that defines the nearest neighborhood of a point 2) the number of points k that is expected to lie in the neighborhood of radius. Small changes in either parameter often yield in drastically different clustering solutions. Hence, stable clustering characteristics are important for an efficient parameter selection. For example: a parameter based stability criterion is used to determine clustering in RECORD [3]. However, the naive criterion of RECORD often results in generation of trivial clustering solutions for complex datasets. OPTICS [4] generates a reachability plot for the dataset. The reachability plot shows the nature of clustering obtained at different parameter values. Similarly, for complex datasets, reachability plot is often difficult to interpret as demonstrated in [5].

2. Curse of Dimensionality : The curse of dimensionality refers to two major issues- data sparseness and distance concentration. Data sparseness arises due to an increase in the containing volume (envelope) of all points with an increase in dimensions. Thus, more amount of data is required to derive statistically sound estimates, i.e., the shape, the size and the distribution of clusters.

Distance concentration refers to the phenomenon of diminishing difference between nearest and farthest neighbor of a point, as the number of dimensions increase. Thus, data sparsity along with the distance concentration makes the task of obtaining clusters difficult. Hence, any algorithm that use distance thresholds to determine clusters will fail because distance not only varies a lot, but is also a difficult parameter to set.

Certain class of clustering algorithms like subspace [6] and randomized projection clustering techniques [7] deal with the curse of dimensionality by identifying a relevant subset of dimensions. Such clustering algorithms localize the search for relevant subset of dimensions, where clusters could exist in multiple possibly overlapping subspaces. Our algorithm works on the complete space of dimension and thus is different from the above mentioned approaches.

In this paper, we present a solution for high dimensional clustering that uses the concept of reverse nearest neighbors and strongly connected components. The clustering algorithm is called strongly connected component clustering algorithm (SCCC). The SCCC has smooth clustering characteristics with respect to a change in the parameter values which means the clustering do not change drastically with little change in parameter value. We further discuss the reasons and empirical analysis of stability of the clustering solution. We also provide a preliminary parameter estimation technique for the SCCC algorithm, which can be efficiently fine tuned using greedy search method proposed in this paper.

In this work we :

1) formulate strongly connected component clustering algorithm (SCCC) which utilises the k-reverse nearest neighbors to detect outliers and clusters.
2) demonstrate how and why the SCCC algorithm is not sensitive to small changes in parameter values.
3) demonstrate the technique for parameter estimation.
4) perform extensive experimental evaluation comparing the SCCC algorithm with other state of art algorithms on datasets varying in size and dimensionality.

In Section I-A, we discuss the related work and motivate the need for this type of algorithm. Section I-B provides a background on reverse nearest neighbors. Section II presents the definitions and the algorithm. We present insights into the nature of seed points, core points, the strongly connected component structure (SCC) and parameter estimation details backed by empirical evidence in Section III. Section IV presents results of experiments conducted on real and synthetic datasets which validate the efficiency of the results. We conclude our work in Section V.

A. Related Work

Various approaches have been proposed for tackling the issue of high dimensional clustering. SNN [8] uses shared nearest neighbors to determine the distance between points instead of vanilla distance measure like euclidean or cosine. Houle [9] outlines that the shared nearest neighbors are more effective in reducing dimensionality curse. However, SNN requires three parameters which is difficult to tune. Density Peaks [10] approach is based on the idea that cluster centers are characterized by a higher density than their neighbors and by a relatively large distance from points with higher densities. However, its performance is usually affected by the cutoff distance, the density peaks, the selection of cluster centers and the allocation strategy of data points. HDBSCAN [11] extends DBSCAN [2] by extracting cluster hierarchy from the dataset with respect to a parameter $minpts$. Thus, different values of ϵ were used for generating different clusters. Further, the notion of stability of clusters is used to obtain a flat clustering from the hierarchy. Setting of $minpts$ parameter is a challenge in this algorithm.

Mean Shift [12] algorithm is a parameter free algorithm. It is a kernel density based algorithm which works by finding peaks of kernel density. However, the performance of kernel density estimators decrease with increase in dimension and becomes increasingly inefficient to run. RECORD [3] uses the number of clusters to measure the stability of the clustering solution. This naive assumption often results in trivial solutions.

The concept of reverse nearest neighbors has been used in clustering.This concept is recently studied by Radovanovic et. al [13]. Radonovic et. al [14] identified influential points using reverse nearest neighbor, also called as hubs. Hubs have more k-reverse nearest neighbors than a predefined threshold [1]. This is the first work that elaborates on the importance of hubs as useful candidates in clustering, classification, and semi-supervised learning. Radovanovic et. al [13] further studied the properties of the hub points and used them to initialize cluster centroid. These observations were used in proposing GKH [13], GHPC [13] and Global Hubness-Proportional K-Means (GHPKM) [13] clustering algorithms, as a variation of the K-means [15] algorithm using hubs as cluster prototypes or guiding points. As these algorithms are variations of K-Means [1], they require number of clusters as a parameter which is difficult to obtain beforehand. The resulting algorithm was found to be more robust to noise and providing better clustering solution over K-means++ [16].

In conclusion, it can be seen that the current literature provides solution either to the problem of high dimensional clustering or to that of the challenges in parameter estimation but not both. There is much need however for solving both of these problems simultaneously.

B. Background

In real life high dimensional data, there are always subsets of points which are mutually closer, hence increasing the possibility of introducing gaps between the group of points. Our aim is to identify local relatively compact regions which together form a single cluster inspite of sparse regions bridging them to form one cluster. These local compact regions can be considered to be the representative regions. It is challenge merging these compact regions because it requires parameters which can control the merging of compact disconnected sub-regions of a cluster while restricting the merging of different clusters. Further, the points that lie in the sparse regions between dense sub-regions within the clusters should be merged with the appropriate cluster while making sure that outliers and noise points are not included in the cluster. Achieving all of this within reasonable computation time in high dimensions is an added challenge while keeping the number of parameters low and ensuring that the clustering solution does not change with small changes in parameter values.

k-Reverse Nearest Neighbors: The concept of reverse nearest neighbors has been used in literature to identify the influence set in a database for querying [17]. The reverse neighbors has been used in data mining for various tasks including clustering [3], [13], classification [14], [18]–[20] and outlier detection [21] tasks. [22]–[24] use the concept of $kRNN$'s to identify the relatively dense regions within a cluster.

The number of reverse nearest neighbors of a point can vary from 0 to $n-1$ where n is the number of points in the dataset. The higher the number of reverse nearest neighbors of a point A, more is the number of points that consider the point A to be among their k-nearest neighbors.

The reverse nearest neighbors of a point have two interesting and subtle properties.

The number of $kRNN$s provide us with a rough understanding of the local density within the neighborhood of a point. The points having low $kRNN$ count can be associated with low density [13]. However, it can not be

[1] 2*standard deviation + expectation

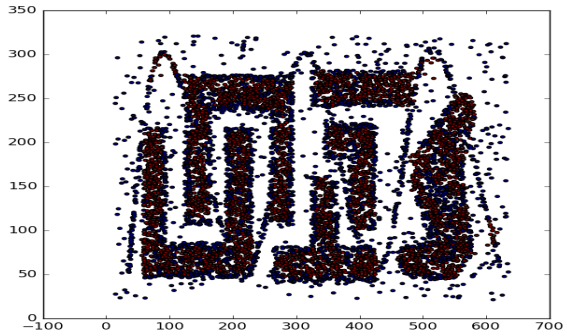


Fig. 1: Points with higher reverse nearest neighbor($\geq k$) are shown in maroon color in the figure. Blue points have lower than k reverse nearest neighbors. It can be observed that maroon points are most concentrated in high dense regions.

said with confidence that points of high $kRNN$ count are points of higher density. This phenomenon can be observed from the Figure 1. Often with development in well studied areas, such as clustering [25], the solutions provided become more difficult to understand with dependencies on multiple parameters/conditions. The strength of our solution lies in its simplicity where the use of powerful ideas put together can produce better or comparable results to the state of art of clustering solutions [2], [11], [13], [26].

II. Definition, Insight and Algorithm

Let $X = \{x_1, ..., x_i, ..., x_N\}$ be a dataset of points where each point $x_i = (x_{i1}, x_{i2}, ..., x_{ij}, ..., x_{id}) \in R^d$ is a feature vector, where d denotes the number of dimensions and x_{ij} denotes the j^{th} feature of the i^{th} data point in the dataset. The cardinality of a set A is denoted by $|A|$. The distance between two points x_i and x_j is denoted by $dist(x_i, x_j)$. $dist(x_i, x_j)$ can either be calculated using Euclidean, Cosine, Manhattan or any other metric. In this paper, the distance measure is assumed to be Euclidean unless otherwise stated. Let us denote $kNN(x_i)$ as the set of k nearest neighbors of x_i.

Definition 1: **k-reverse nearest neighbor set** of a point x_i, denoted by $kRNN(x_i)$ is defined as the points in the set $\{ x_j \in X | x_i \in kNN(x_j)\}$.

Reverse Nearest Neighbors can be used to identify locally compact neighborhood of points in the dataset irrespective of density distribution. A point x_i, having $|kRNN(x_i)| < k$ is considered to be at lower density as discussed in Section I-B.

Definition 2: We define **Seed Point** to be a point, x_i for which $|kRNN(x_i)| \geq k$. The set of Seed Points in the dataset is denoted by $S(X)$.

It can be understood that the seed points are considered nearest neighbors of greater than or equal to expected number of points (k). Further, k is the lowest threshold for which seed points are high density points with a high probablity. *Although, the definition of the seed point bears similarity to the definition of the hub point [13], the characteristic of the seed point and hub point is different. According to [13], the hub point refer to the point that has more than expectation+ 2*deviation number of reverse nearest neighbors. Hub points are fewer in number in comparison to seed points, and do not provide with the required number of points that are minimally needed to connect all points of a cluster. More importantly, the definition of hub entails using a global measure which is the expectation and variance of the $kRNN$'s of all points in the dataset. It hence, finds regions of global high density. Our aim is however to identify local regions of relatively high density compared to its adjacent surroundings. Such a local definition helps identify relative dense subregions well spread throughout the clusters.*

Definition 3: We define a core point to be a point, $x_j | \{x_j \in S(X)$ and $|kNN(x_j) \cap S(X)| \geq \tau\}$. The set of all core points in X is defined as $core(X)$.

τ is a thresholding parameter and the seed points having greater or equal to τ nearest neighbors is termed as core points.

In our work, we use number of reverse nearest neighbors ($|kRNN|$) to determine our seed points to find regions of relatively high density. However, for determining core points, we add an additional filtering to remove outlier and noise points. The initial seed points determined by points with a high $kRNN$ neighborhood form a compact region of influence. However, two seed points that are outliers could result in the merging of two clusters. To avoid such a phenomenon, we give importance to the overlapping region of influence of two seed points. For the compact regions of two seeds to overlap, the kNN of a seed can be expected to contain a seed. The higher the number of seeds present in the KNN neighborhood of a given seed, the more is the assurance of a strong mutual connectivity. For this purpose, we introduce a parameter τ. τ defines the number of seed points that need to be present in the neighborhood of a seed point P for the P to be called a core point. In this paper, we use τ=2. Our empirical analysis shows that most of the core points lie within a cluster. Our simple idea of using a threshold for seed point to qualify as core points has substantial improvement in results due to the removal of seed points that lie in the space between disparate clusters and those that lie in the outlier region.

Definition 4: **CoreGraph(X)** is a directed graph wherein the set $core(X)$ forms the vertices of the graph. A directed edge (u,v) exists from core point u to another core point v, if v belongs to the set $kNN(u)$.

Graph G is said to be strongly connected, if every vertex in G is reachable from every other vertex in G.

Definition 5: $SCC(X)$ is the set of strongly connected components obtained on running the strongly connected components algorithm [27] on $CoreGraph(X)$.

Including directed edges between two core points where one is the kNN of the other, gives us an initial understanding of connectivity between core points. However, it is the strongly connected component of the graph that can best capture mutual connectedness of the points. A strongly connected component (SCC) [27] of a directed graph partitions the graph into directed subgraphs within which each node is mutually reachable

from other nodes. If a connection has been established from a point A to another point B through intermediate KNN connected nodes, the fact that there also exists a connectivity from B to A, reinforces the compactness of the connectivity among points. Strongly connected component of a graph provides for this reinforced mechanism of connectivity. The higher the value of k, the more is the compactness of the group.

The points that do not belong to any of the SCC's computed might still lie close enough to a cluster. Further processing is done on the points that are not members of any of the computed SCC in order to include possible cluster points. The following steps are followed in order to identify such cluster points and determine outliers .

1) The points having atleast $\max\{1, k/d\}$ core point in its kNN is merged with the nearest cluster.
2) The points which have less than $\max\{1, k/d\}$ core points in its neighborhood are treated as an outlier.

The fraction, k/d is used to factor in the sparsity of data points that arise in the case of high dimensionality. Thus, this fraction is a good indicator of identifying whether a neighborhood is sparse or not. However, for $d > k$, this fraction will be less than 1 resulting in no outlier. Hence, we set the threshold to $\max\{1, k/d\}$, so that if a point does not have any core point in its neighborhood, it gets eliminated and is labelled as an outlier.

Algorithm 1: GENERATECORE

Input : X, k, τ
Output: $core(X)$

1 $S(X) \leftarrow \phi$;
2 $core(X) \leftarrow \phi$;
3 $CoreGraph(X) \leftarrow \phi$;
/* Calculate seed points by finding reverse nearest neighbor as per Definition 2. */
4 $S(X) = FindSeed(X, k)$
/* Identify core points from the seed points. */
5 **foreach** $(x_i \in S(X))$ **do**
6 **if** $(|kNN(x_i) \cap S(X)| \geq \tau)$ **then**
7 $core(X) := core(X) \cup x_i$;
8 **end**
9 **end**

Algorithm 1 computes the set of all core points. Algorithm 2 uses the core points generated to compute the final set of clusters. Algorithm 2 first generates core points using $GenerateCore$ in line 2. In line 3-9 of Algorithm 2, all the core points are connected to generate $CoreGraph(X)$. In line 10, the SCC of the Core Graph is obtained. From lines 11-19, noncore points and outliers are identified. Non core points are merged to the cluster containing nearest core point. The worst case time complexity of our algorithm is $O(n^2)$ and is dominated by run time of finding k-nearest neighbors. The memory requirement is $O(k.n)$, where k is the neighborhood parameter.

Algorithm 2: SCCC

Input : X, k, τ
Output: $C(X) = C_1(X) \cup C_2(X) \cup \ldots C_m(X)$, $C_{outlier}(X)$

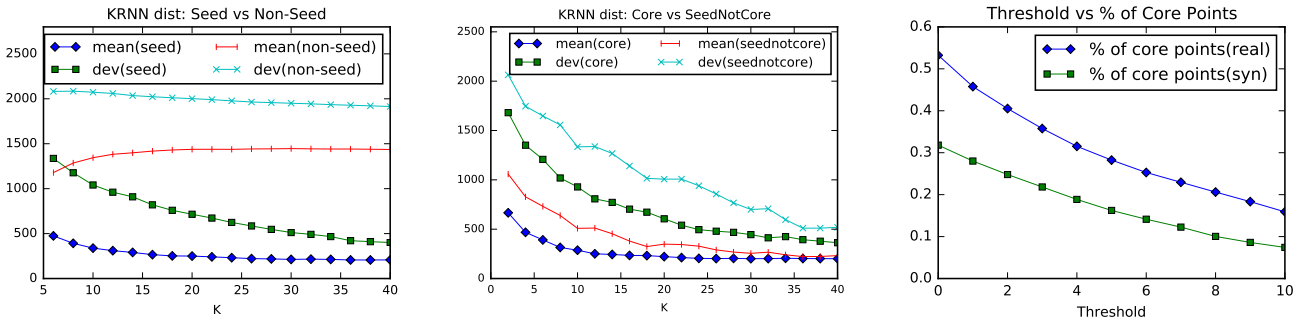
1 $C_{outlier} \leftarrow \phi$
2 $core(X) := GENERATECORE(X, k, \tau)$
/* Create CoreGraph from the core points. */
3 **foreach** $(x_i \in core(X))$ **do**
4 **foreach** $(x_j \in core(X)\ AND\ x_i \neq x_j\ )$ **do**
5 **if** $((x_j \in kNN(x_i)))$ **then**
6 $CoreGraph(X) := CoreGraph(X) \cup (x_i \rightarrow x_j)$
7 **end**
8 **end**
9 **end**
/* Identify strongly connected components of CoreGraph */
10 $C(X) := SCC(CoreGraph(X))$
11 **foreach** $(x_i \in (X - core(X)))$ **do**
12 **if** $(|(kNN(x_i) \cap core(X)| \geq \max\{1, k/d\})$ **then**
/* Merge the point to SCC belonging to nearest core point */
13 $C_{x_i}(X) := getNearestPartition(C(X), x_i)$
/* $getNearestPartition(C(X), x_i)$ generates nearest cluster partition to x_i */
14 $C_{x_i}(X) := C_{x_i} \cup x_i$
15 **end**
16 **else**
17 $C_{outlier}(X) := C_{Outlier}(X) \cup x_i$
18 **end**
19 **end**

III. EMPIRICAL ANALYSIS

In this section, we provide insights and empirical analysis of the set of seed points, core points, the SCC built and the other characteristics of the algorithm.

A. Seed Points

For empirical evaluation, we have created a synthetic dataset as a mixture of Gaussians with 10000 datapoints and 50 dimensions. We calculate the distance of the farthest $kRNN$ of all seed points and non seed points. We calculate for each seed point, the mean of the distances from the seed to each of its $KRNN$'s. The average of such distances for all seeds are then calculated for changing values of the parameter k. We repeat this experiment on the set of non seed points as well. The experimental results on the dataset is presented in Figure



(a) (b) (c)

Fig. 2

2a. We provide results for varying values of k until the value of k exceeds the value used in the other experiments. Note that the mean of the all $kRNN$ of non seeds is far larger that of the seed points emphasizing that the seed points are regions of relatively higher density. The seed points were seen to form around 40-50% of the dataset.

B. Core Points

Figure 2b show the similar experiments as in Figure 2a, where we compare the mean of $kRNN$ count of core points against the mean of $kRNN$ count of the seed points which are not core points. The means are calculated as in Figure 2a. It can be observed that mean of core point is lower than mean of non core seed points and there is a decrease in standard deviation also. It is evident from the experiment that the core points have regions of higher density concentration than non core seed points. The discarded set of points(seed not core) tend to merge clusters as the mean of their $kRNN$ count is low.

Figure 2c shows the relation between number of core points and threshold τ. We change the number of seed points required to be present in the kNN neighborhood of a seed for it to qualify as a core point. We use the term 'seed threshold τ' to denote the number of seeds required to be present in the kNN of seed for it to qualify as a core point. From Figure 2c, it can be observed that the number of core points is inversely proportional to τ. It was observed experimentally that setting $\tau = 2$ removed most of the confusing seed points while still retaining most of the informative seed points as part of cluster.

C. Strongly Connected Component (SCC)

We compared clustering of SCC (core points) with clustering obtained after merging other points with SCC. It was observed that as expected core points were better clustered as compared to other points. It was observed that the ARI of the core points after SCC is built is 0.9 for the synthetic dataset. After the final merging of the core points with other points, the ARI reduced to 0.7845. Similarly, the average cluster entropy (ACE) was initially 0 for the SCC points which increase to 0.422 after the non core points were merged.

D. Parameter Estimation

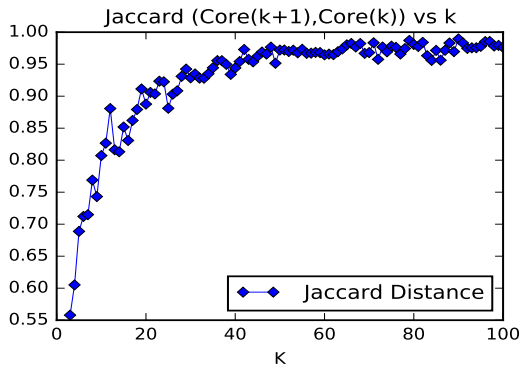
Figure 3 shows the plot of Jaccard similarity of consecutive core points with change in the parameter k for the synthetic dataset described earlier and segmentation dataset. Segmentation is a real life dataset with dimension 23 and 2100 data points and was obtained from UCI Machine Learning Repository [28].

Initially the variation in the set of core points is large with an incremental increase in k. However, the set of core points does not vary substantially beyond a certain k. This can be observed for both real and synthetic datasets. It is obvious that a change in the set of core points will directly reflect on the clustering solution where a significant shift in core points will show a significant change in the solution and a small change will only a reflect a minor difference in the solution. Thus, changes in clustering solution is smooth with respect to parameter k.

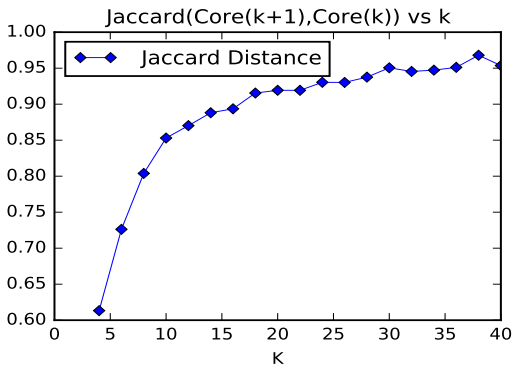
We set the initial value of the parameter k to the value of where the curve stabilises initially. Further, greedy search approach can be used to change the parameters using a binary search. By greedy search, we mean the user can take huge strides for parameter estimation. The user has a handle on what to expect as he increases or decreases the value of k as small changes need not be a worry. The parameter k can be iteratively doubled, if the user is interested in merging existing clusters. Similarly, the parameters can be iteratively halved, if the user is interested in separating existing clusters.

For the synthetic dataset, we obtained $k = 21$ which yielded 3 clusters according to our expectation. Hence, $k = 13$ was used. Further, when we used above technique for segmentation dataset, the value of initial k obtained was 53 and number of clusters obtained was 3. To have more separation between the clusters, we halved the parameter k and obtained 4 clusters at $k = 26$. Similarly, we further halved the parameter and obtained 6 clusters. Further, halving produced undesirably large number of clusters and hence, we settled at $k = 13$.

Thresholding parameter τ causes small changes in clustering. Based on experiments conducted, we set $\tau = 2$ for all of the datasets.



(a) Segmentation dataset



(b) Synthetic dataset

Fig. 3: We plot the Jaccard Similarity of distribution of core point generated at k and k+1. It is observed it becomes stable at some point. This initial stable region coressponds to initial value of k. For segmentation dataset value of k is around 53 and for synthetic dataset it is 21.

IV. Experiments and Results

A. Dataset

We have selected various real life and standard datasets of varying dimensions and sizes to demonstrate the efficiency of our approach. We pick datasets from varied domains like image, text and genes and provide results against algorithms that are most challenging to our work. We have divided the dataset into two groups 1) Low and Moderate dimension :- The datasets having 0 to 50 dimensions. This includes iris [28], glass [28], seed [28] and loan https://www.lendingclub.com/info/download-data.action. Further, it includes cellCycle384 and cellcycle 237 [29], image segmentation [28], wine [28] and wdbc [28]. The datasets with greater dimension than fifty are termed as high dimensional datasets. BBC and BBCSport [30] are text datasets. BBC consists of 2225 documents from the BBC news website corresponding to stories in five topical areas from 2004-2005. It has five class labels. Similarly, BBCSport consists of 737 documents from the BBCSport website corresponding to sports news articles in five topical areas from 2004-2005 and it consists of five class labels. We obtained the preprocessed dataset from [2]. MNIST [31] is a handwritten image data of digits having classes 0 to 9. We have created two subclasses of the MNIST dataset - the MNIST dataset containing data of labels corresponding to labels 0 and 1. This is the most simple setting for identifying distinct clusters in the dataset and is termed as MNIST01. Further, labels corresponding to 0,4,6 and 9 is used to create a subset as described in [32] called MNIST0469. Malicious Executable(AntiVirus) dataset [28] contains the most commonly occurring features in malacious and non-malacious program files. Christensen [33], Su [34] and Yeoh [35] are biological datasets and Digits is another image dataset for $0-9$ digits. [28].

[2]http://mlg.ucd.ie/datasets/bbc.html

B. Qualitative Evaluation

We have used two different metrics to measure clustering quality- Adjusted Rand Index (ARI) [36] and Average Cluster Entropy [37]. We have compared our results with HDBSCAN [11], GHPKM [13] and RECORD algorithm. HDBSCAN is able to identify variable density clusters and is an extension of popular DBSCAN [2] algorithm. Further, it has been shown that it performs well in higher dimensions. Hence, it was chosen to compare our approach with density based algorithms. We have used R implementation of HDBSCAN available at [3]. The parameter $minpts$ was varied from 0 to 50 and the best clustering was chosen. Our implementation with datasets is available at this link[4]. GHPKM is a reverse nearest neighbor based partitional algorithm, which uses hubs [14]. We have used the Hubminer tool [5]. We set the number of clusters equal to number of unique cluster labels in gold. The neighborhood parameter was varied from 2 to 30 and the best solution was chosen for comparison.
RECORD [3] is another SCC based algorithm which utilizes reverse nearest neighbors in their clustering solution, is the most similar algorithm to ours in terms of approach. We obtained the code of RECORD from the authors. We varied the parameters from 0 to 100 and used best clustering for comparison. We set our parameters with initial clustering parameter obtained as discussed in Section III-D and then performing grid search. Except for the datasets BBC and BBCSports, for which we have used cosine distance, euclidean distance was used. We have not shown results of HDBSCAN over BBC and BBCSports dataset, as the implementation used by us does not support cosine metric.

Table I shows the comparison of the performance of the algorithms: GHPKM, RECORD and HDBSCAN on the real life datasets. It can be observed that SCCC performs better than GHPKM on all the datasets. The reason can be attributed

[3]goo.gl/znnv7h
[4]goo.gl/Uns5mh
[5]https://github.com/datapoet/hubminer

TABLE I: Comparison of accuracy of different clustering algorithms using Adjusted Rand Index (ARI) and Average Cluster Entropy (ACE)

				SCCC		GHPKM		HDBSCAN		RECORD	
dataset	dim	size	label	ARI	ACE	ARI	ACE	ARI	ACE	ARI	ACE
Moderate Dimension (dim $\preccurlyeq$ 50)											
iris	4	150	3	**0.868**	**0.228**	0.334	1.57	0.568	0.667	0.568	0.667
glass	8	214	7	0.271	1.49	0.263	2.16	0.251	1.532	**0.295**	**1.24**
seed	7	210	3	**0.646**	**0.558**	0.209	0.996	0.30	0.708	0.0005	1.53
loan	10	11468	2	0.254	0.675	0.255	0.98	**0.256**	**0.670**	0.2118	0.679
segmentation	23	2100	7	**0.418**	**0.793**	0.1413	2.86	0.338	1.365	0.302	1.141
wine	34	178	3	**0.399**	**0.878**	0.341	1.557	0.278	0.942	0.387	1.058
cellCycle237	17	237	4	**0.535**	**0.965**	0.4133	1.577	0.48	0.983	0.205	1.139
cellCycle384	17	384	5	**0.485**	**0.99**	0.232	2.215	0.371	1.49	0.208	0.166
wdbc	30	569	2	0.528	0.520	0.53	0.95	**0.555**	**0.381**	0.436	0.911
soybean	35	307	18	**0.353**	**0.891**	0.09	3.8	0.301	1.69	0.286	1.905
High Dimension (dim $\succ$ 50)											
bbc	9635	2235	5	**0.409**	**0.707**	0.20	2.31	-	-	0.043	0.144
MNIST01	784	12665	2	**0.993**	**0.007**	0.209	0.997	0.624	0.064	0.721	0.254
MNIST0649	784	23632	4	**0.685**	**0.552**	0.06	1.99	0.206	1.336	0.007	1.94
antivirus	531	373	2	0.817	0.157	0.69	0.69	**0.906**	**0.047**	0.708	0.698
christensen	1413	217	3	**0.841**	**0.029**	0.429	1.323	0.482	0.178	0.25	0.188
bbcsport	4613	737	5	**0.369**	1.180	0.233	2.3	-	-	0.074	**0.290**
digits	64	1797	10	**0.811**	**0.384**	0.10	3.319	0.593	0.978	0.7081	0.416
su	5565	102	4	**0.947**	**0.119**	0.254	1.96	0.32	0.380	0.323	1.177
yeoh	12625	249	5	**0.825**	**0.487**	0.224	2.337	0.146	1.89	0.391	1.153

$^*m \Rightarrow minpts$

TABLE II: Sensitivity to parameters

	D10N20			D30N20			D50N20			D100N20		
k	ARI	ACE	Sil	ARI	ACE	Sil	ARI	ACE	Sil	ARI	ACE	Sil
5	0.9923	0	0.736	0.9776	0.0358	0.74	0.8085	0.339	0.721	0.805	0.3481	0.721
10	0.9991	0.0013	0.718	0.9655	0.0774	0.74	0.785	0.4197	0.717	0.7847	0.4213	0.720
15	0.9998	0.0011	0.699	0.9281	0.1526	0.735	0.785	0.424	0.720	0.7841	0.4241	0.712
20	0.9998	0.0011	0.726	0.8820	0.2313	0.730	0.785	0.4242	0.776	0.7840	0.424	0.77

to partitional nature of GHPKM algorithm, which does not allow the identification of various shapes and sizes of clusters. HDBSCAN performs marginally better on antivirus, wdbc and loan dataset. It can can be noticed that our results are significantly better than HDBSCAN for high dimensional datasets. For example: MNIST01 and digits have a difference of almost 0.3 in the ARI and 0.6 in average cluster entropy. Further, it can be observed that SCCC performs better than RECORD in all datasets except glass, where the results of RECORD is marginally better.

C. Sensitivity to Parameters

We have generated several synthetic dataset of size 10000, with dimensions 10, 30, 50 and 100 as a mixture of Gaussian. The number of Gaussian was set to 3 and their deviation was chosen from $\{$0-10$\}$. We have further added 20 % uniform noise to these dataset from the space [-200,200]. These datasets are named D30N20, D10N20, D100N20 and D50N20 where suffix of D represents the dimension and suffix of N represents percentage of noise added.

Table II compares the variation in the clustering performance with a change in k. It can be observed from the Table II, that there is not a significant difference in clustering for various parameters. The experiment was repeated for real life datasets where the initial value of the parameter k was set to the optimal. The results of the clustering solution when k is varied by 20% above and below the optimal value. Minimum variation in clustering was obtained for MNIST01 dataset and was in the range $[\pm 0.02, \pm 0.002]$ for ARI and ACE respectively. Maximum variation was observed for Digits dataset and was in the range $[\pm 0.1, \pm 0.4]$ for ARI and ACE respectively. Mean variation was found to be in the range $[\pm 0.04, \pm 0.26]$ for ARI and ACE. It can be observed that there is larger difference in variation in clustering quality as compared to synthetic datasets (considering the absolute range the parameters were varied). It can be attributed to more complex distribution of points in real life setting. However, variation in clustering was smooth across complete parameter range. This slow change in results with change in k makes it easier to pick a suitable value of k.

V. Conclusion

High dimensional data clustering is a challenging problem due to curse of dimensionality and difficulty in parameter estimation. Our solution identifies core points as locally compact regions defined by a constraint on the number of reverse nearest neighbors. The core points act as representative points of the clusters. The strongly connected component of a directed kNN graph ensures that the cluster represented has point which are mutually connected (or similar). Thus, the strongly connected component form basic clusters, and other points are incorporated based on nearest neighbors of basic cluster points. Our empirical analysis shows the rationale for using reverse nearest neighbors for defining core points, and strongly connected components to determine basic clusters. The experimental results show that our clustering solution performs as good as and many times better than state of the art algorithms. As part of future work, we will study the importance of core points from the context of classification problem to analyse the notion of a class.

References

[1] J. Burkardt, "K-means clustering," *Virginia Tech, Advanced Research Computing, Interdisciplinary Center for Applied Mathematics*, 2009.

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

[3] S. Vadapalli, S. R. Valluri, and K. Karlapalem, "A simple yet effective data clustering algorithm," in *ICDM*, 2006, pp. 1108–1112.

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

[5] 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.

[6] L. Parsons, E. Haque, and H. Liu, "Subspace clustering for high dimensional data: a review," *Acm Sigkdd Explorations Newsletter*, vol. 6, no. 1, pp. 90–105, 2004.

[7] H.-P. Kriegel, P. Kröger, and A. Zimek, "Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 3, no. 1, p. 1, 2009.

[8] L. Ertoz, M. Steinbach, and V. Kumar, "A new shared nearest neighbor clustering algorithm and its applications," in *SDM*, 2002, pp. 105–115.

[9] M. Houle, H.-P. Kriegel, P. Kröger, E. Schubert, and A. Zimek, "Can shared-neighbor distances defeat the curse of dimensionality?" in *Scientific and Statistical Database Management*. Springer, 2010, pp. 482–500.

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

[11] 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.

[12] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE transactions on pattern analysis and machine intelligence*, vol. 17, no. 8, pp. 790–799, 1995.

[13] N. Tomaev, M. Radovanovi, D. Mladeni, and M. Ivanovi, "The role of hubness in clustering high-dimensional data," in *Advances in Knowledge Discovery and Data Mining*. Springer Berlin Heidelberg, 2011.

[14] M. Radovanović, A. Nanopoulos, and M. Ivanović, "Hubs in space: Popular nearest neighbors in high-dimensional data," *Journal of Machine Learning Research*, vol. 11, no. Sep, pp. 2487–2531, 2010.

[15] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means: spectral clustering and normalized cuts," in *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2004, pp. 551–556.

[16] D. Arthur and S. Vassilvitskii, "k-means++: The advantages of careful seeding," in *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035.

[17] F. Korn and S. Muthukrishnan, "Influence sets based on reverse nearest neighbor queries," in *ACM Sigmod Record*, vol. 29, no. 2. ACM, 2000, pp. 201–212.

[18] N. Tomašev, M. Radovanovic, D. Mladenic, and M. Ivanovic, "A probabilistic approach to nearest-neighbor classification: Naive hubness bayesian knn," in *Proc. 20th ACM Int. Conf. on Information and Knowledge Management (CIKM)*, 2011, pp. 2173–2176.

[19] M. Radovanović, A. Nanopoulos, and M. Ivanović, "Time-series classification in many intrinsic dimensions," in *Proceedings of the 2010 SIAM International Conference on Data Mining*. SIAM, 2010, pp. 677–688.

[20] N. Tomasev and D. Mladenic, "Nearest neighbor voting in high-dimensional data: Learning from past occurrences," in *Data Mining Workshops (ICDMW), 2011 IEEE 11th International Conference on*. IEEE, 2011, pp. 1215–1218.

[21] M. Radovanović, A. Nanopoulos, and M. Ivanović, "Reverse nearest neighbors in unsupervised distance-based outlier detection," *IEEE transactions on knowledge and data engineering*, vol. 27, no. 5, pp. 1369–1382, 2015.

[22] T. N. Tran, R. Wehrens, and L. M. Buydens, "Knn density-based clustering for high dimensional multispectral images," in *Remote Sensing and Data Fusion over Urban Areas, 2003. 2nd GRSS/ISPRS Joint Workshop on*. IEEE, 2003, pp. 147–151.

[23] E. Biçici and D. Yuret, "Locally scaled density based clustering," in *International Conference on Adaptive and Natural Computing Algorithms*. Springer, 2007, pp. 739–748.

[24] C. Zhang, X. Zhang, M. Q. Zhang, and Y. Li, "Neighbor number, valley seeking and clustering," *Pattern Recognition Letters*, vol. 28, no. 2, pp. 173–180, 2007.

[25] A. K. Jain and R. C. Dubes, *Algorithms for clustering data*. Prentice-Hall, Inc., 1988.

[26] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM computing surveys (CSUR)*, vol. 31, no. 3, pp. 264–323, 1999.

[27] H. N. Gabow, "Path-based depth-first search for strong and biconnected components," *Information Processing Letters*, vol. 74, no. 3-4, pp. 107–114, 2000.

[28] A. Asuncion and D. Newman, "Uci machine learning repository," 2007.

[29] C. R. et al, "A genome-wide transcriptional analysis of the mitotic cell cycle," *Molecular cell*, vol. 2, no. 1, pp. 65–73, 1998.

[30] D. Greene and P. Cunningham, "Practical solutions to the problem of diagonal dominance in kernel document clustering," in *Proc. 23rd International Conference on Machine learning (ICML'06)*. ACM Press, 2006, pp. 377–384.

[31] Y. LeCun and C. Cortes, "The mnist database of handwritten digits," 1998.

[32] P. Raman and S. Venkatasubramanian, "Power to the points: Validating data memberships in clusterings," in *Data Mining (ICDM), 2013 IEEE 13th International Conference on*. IEEE, 2013, pp. 617–626.

[33] B. C. C. et al., "Aging and Environmental Exposures Alter Tissue-Specific DNA Methylation Dependent upon CpG Island Context," *PLOS Genetics*, vol. 5, no. 8, p. e1000602, Aug. 2009.

[34] A. I. S. et al, "Large-scale analysis of the human and mouse transcriptomes." *Proceedings of the National Academy of Sciences of the United States of America*, vol. 99, no. 7, pp. 4465–4470, Apr. 2002.

[35] Y. et al, "Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling." *Cancer Cell*, vol. 1, no. 2, pp. 133–143, Mar. 2002.

[36] D. Steinley, "Properties of the hubert-arable adjusted rand index." *Psychological methods*, vol. 9, no. 3, p. 386, 2004.

[37] H. Kim and H. Park, "Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis," *Bioinformatics*, vol. 23, no. 12, pp. 1495–1502, 2007.