Optimizing k-NN graph generation via ANNs Techniques in Chameleon2 Clustering

M. Sai Akshay Reddy, Rishi Parsai

BTP : Mid-Semester

Supervisor:
Prof. Kapil Ahuja

Mentor: Priyanshu Singh

Computer Science and Engineering
Indian Institute of Technology Indore

October 09, 2023

Table of Contents

What is Clustering?

- Clustering involves a process of exploration, where data points or objects are organized into clusters in a way that maximizes similarity within each cluster while minimizing similarity between different clusters.
- The main objective of clustering is to uncover underlying patterns, structures, or inherent groupings within a dataset without any prior information about these groupings.
- It is used in various fields such as Data Mining, Recommendation Systems, Biology and Genetics, Anomaly Detection, Social Network Analysis.

Why not Traditional Clustering Algorithms?

- Common clustering algorithms like K-means, DBSCAN, CURE, ROCK, etc are tailored for static models..
- Moreover, these algorithms often fail when datasets have clusters which have varied shapes, densities, and sizes.

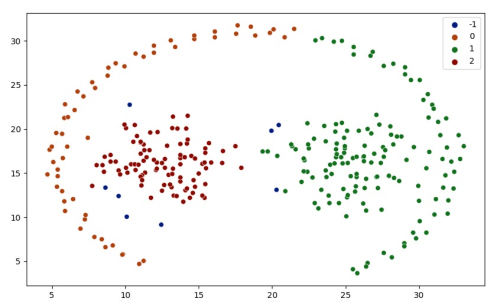


Figure: pathbased dataset : DBSCAN : 0.725

Table of Contents

1 Introduction

2 Previous Work

3 Approximate Nearest Neighbors Algorithm

4 Results

5 Future Work

Chameleon Clustering

- Ch. Clustering finds cluster similarity using **dynamic model**.
- Clusters are merged only if they satify the merging criteria, computed using RI and RC metrics. [3]

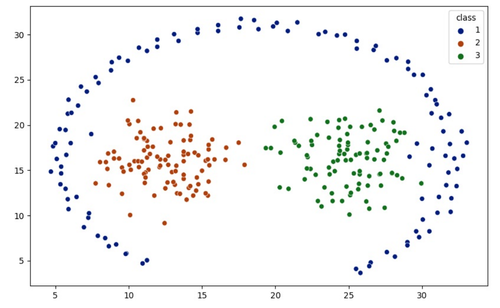


Figure: pathbased dataset, Ch2 : 0.887 , DBSCAN : 0.725

Continued...

- CHAMELEON is a two-phase algorithm .
  1. **K-NN Construction (Pre-Computation)**
  2. **Partitioning Phase (hMETIS Library)**
  3. **Merging Phase**

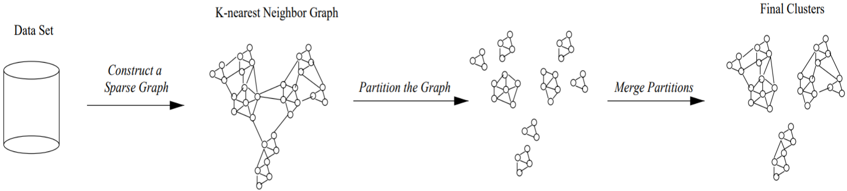
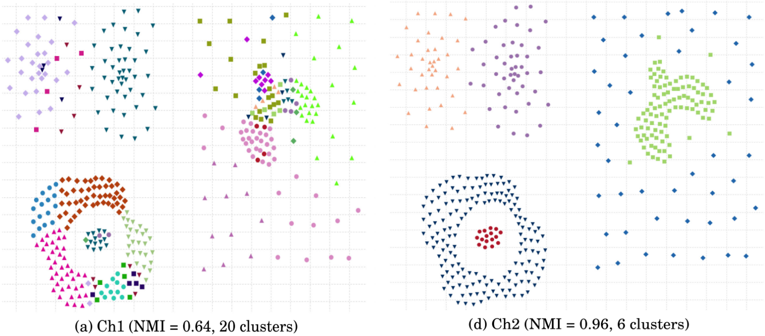


Figure: Two Phase Algorithm

Drawbacks of Chameleon Algorithm

- The main drawback of Chameleon 1 lies in its inability to manage small clusters unlike DBSCAN and CURE, Chameleon doesn't handle noise properly.



(a) Ch1 (NMI = 0.64, 20 clusters)

(d) Ch2 (NMI = 0.96, 6 clusters)

Chameleon 2 Algorithm

- Chameleon 2 (Ch2) is an improved version of Chameleon Algorithm (Ch1)
- Symmetrical K-NN Graph is constructed and this eliminates many between cluster connections and lead to better results.
- Flood Fill Algorithm is applied to refine the partition done recursive bisection or hMETIS.
- Similarity Measures are improved compared to Chameleon 1. [1]

Table of Contents

Approximate Nearest Neighbour Search : ANNS

- The Exact KNN algorithm takes $O(n)$ for a single query point, and for n points it boils down to $O(n^2)$.
- ANNS algorithms sacrifice some degree of accuracy for the sake of efficiency, these algorithms become very useful for large datasets.
- ANN search algorithms build a data structure called index from the dataset, and employ different algorithms to it to attain sub-linear and logarithmic time complexities.
- Here, we have integrated two ANN techniques with our clustering algorithm, namely FLANN and HNSW.

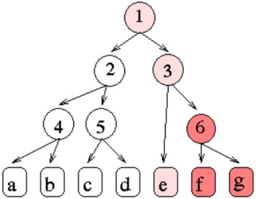
Fast Library for Approximate Nearest Neighbour: FLANN

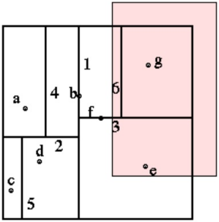
Introduction

- This Library offers three different algorithms for ANN search, they are: *Randomized k-d trees*, *Hierarchical k-means tree* and *Standard linear scan*.
- We have incorporated *Randomized k-d trees* method because of its *higher accuracy* and *easier fine tuning*.
- For lower dimensions it has been shown that the time complexity of k-d trees method for finding NNs for n points comes out to be $O(n \log n)$. [2]

k-d tree

- k-d (or k-dimensional) tree is a binary tree where each node is a k-dimensional point, and every non-leaf node represents a hyperplane that divides space into two half spaces.
- The process begins by selecting a dimension to split the data which is done on various criteria such as finding the dimension with the maximum spread or variance in the data.
- This process is repeated recursively for each subtree until a stopping criterion is met





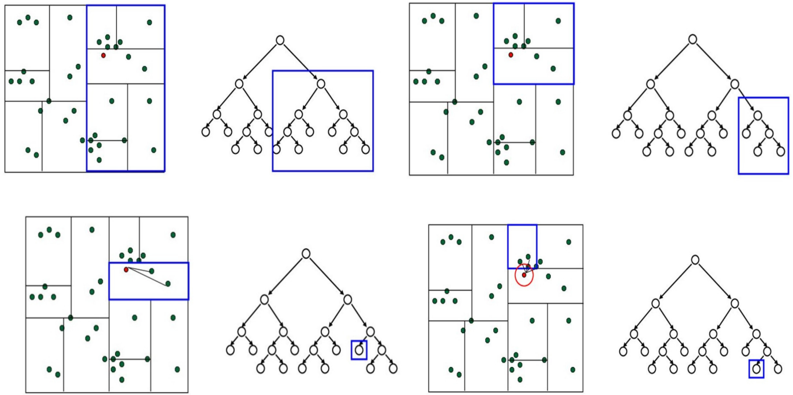
Randomized k-d tree

- Following are the key differences between Randomized and the Traditional k-d tree.

1. Perpendicular hyperplane is centered at the mean of the dimension values of all input data point .
2. The splitting dimension is chosen at random from top-5 dimensions that have the largest sample variance.
3. Multiple randomized k-d trees are built as the index.

Finding Nearest Neighbour in k-d tree

- Explore the branch of tree closest to the query point.
- On reaching the leaf node, distance between the points in region and query point is calculated.



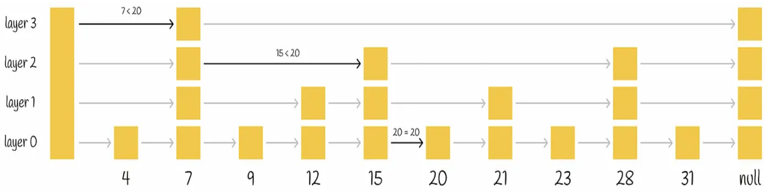
Hierarchical Navigable Small World: HNSW

Introduction

- HNSW is a proximity graph based method.
- HNSW is an efficient in high-dimensional space since its index size is independent of *d*, but other traditional ANN search methods become inefficient due to *curse of dimensionality*. [4]
- The two fundamental techniques that contributed most heavily to HNSW: are **probability skip list**, and **navigable small world graphs**.

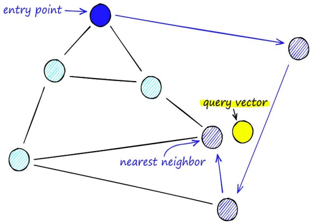
Probability Skip List

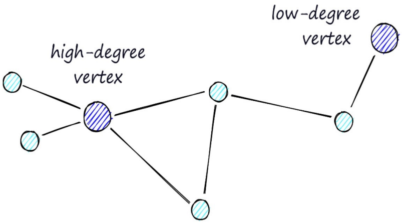
- Skip list is a data structure that allows inserting and searching elements within a sorted list for O(logn) on average.
- A skip list is constructed by several layers of linked lists, with lowest layer as the original linked list having all elements.
- When moving to higher levels, the number of skipped elements increases, thus decreasing the number of connections.



Navigable Small World Graphs

- NSW is a graph structure with polylogarithmic $T = O(log^k n)$ search complexity which uses greedy routing. [5]
- Routing refers to the process of starting the search process from low-degree vertices and ending with high-degree vertices.
- Since low-degree nodes have few connections, the algorithm can rapidly move between them to easily reach the region where the NN is likely to be located.





Coming back to HNSW

- HNSW adds hierarchy to NSW and produces a graph where links are separated across different layers.
- At the top layer, we have the longest links, and at the bottom layer, we have the shortest.
- Process starts from the top layer and proceeds to one level below every time the local NN is greedily found among the layer nodes. The local NN at bottom layer is the answer.

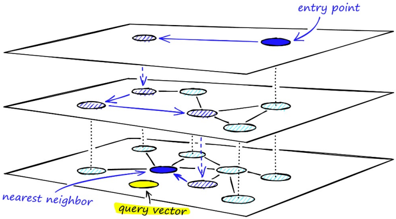
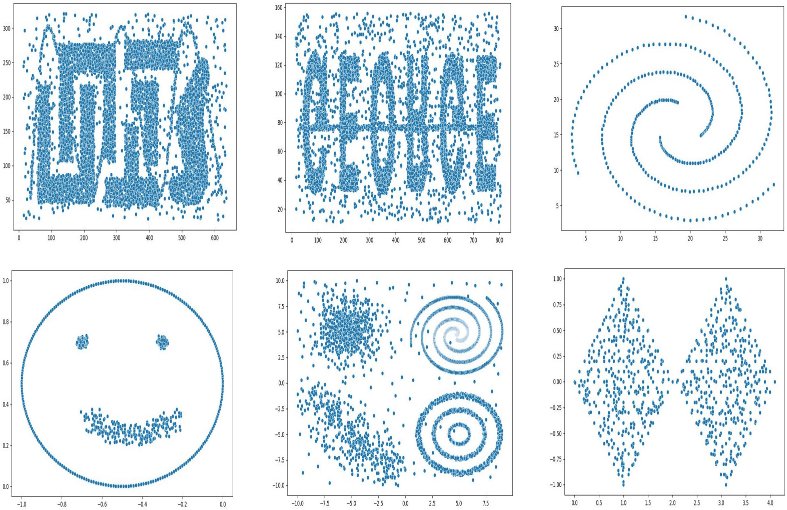


Table of Contents

Representation of some Benchmark datasets



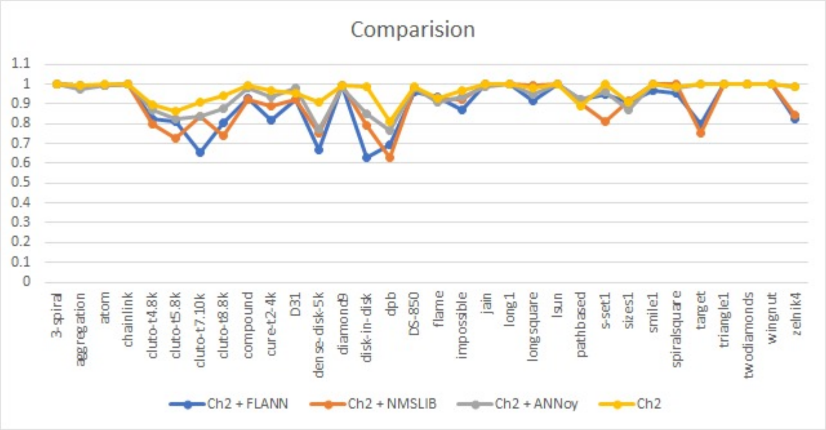
Results: Over 32 Ch2 Benchmark Datasets

Dataset	Ch2 + FLANN	Ch2 + HNSW	Ch2 + ANNoy	Ch2
3-spiral	1	1	1	1
jain	1	1	0.987	1
long1	1	1	1	1
lsun	1	1	1	1
smile1	0.967	1	0.997	1
target	0.795	0.751	1	1
triangle1	1	1	0.997	1
twodiamonds	1	1	1	1
wingnut	1	1	1	1
atom	0.994	0.991	0.993	1
chainlink	1	1	1	1
s-set1	0.95	0.813	0.96	0.997
diamond9	0.996	0.991	0.982	0.993
aggregation	0.996	0.993	0.974	0.992
compound	0.927	0.922	0.978	0.992
disk-in-disk	0.631	0.789	0.85	0.99

Results: Continued

Dataset	Ch2 + FLANN	Ch2 + HNSW	Ch2 + ANNoy	Ch2
spiralsquare	0.953	0.998	0.983	0.987
zelnik4	0.827	0.845	0.986	0.987
DS-850	0.963	0.979	0.983	0.984
longsquare	0.916	0.993	0.95	0.981
impossible	0.872	0.922	0.938	0.969
cure-t2-4k	0.817	0.889	0.938	0.967
D31	0.921	0.922	0.978	0.957
cluto-t8.8k	0.804	0.737	0.875	0.944
flame	0.935	0.927	0.907	0.927
cluto-t7.10k	0.652	0.838	0.84	0.912
sizes1	0.911	0.915	0.87	0.909
dense-disk-5k	0.67	0.75	0.775	0.908
cluto-t4.8k	0.825	0.797	0.871	0.893
pathbased	0.919	0.905	0.924	0.887
cluto-t5.8k	0.812	0.726	0.824	0.864
dpb	0.692	0.626	0.767	0.81
AVG.	0.898	0.901	0.941	0.964
SD.	0.114	0.108	0.07	0.049

Graphical Comparison of NMI Values



Graphical Comparison of Runime

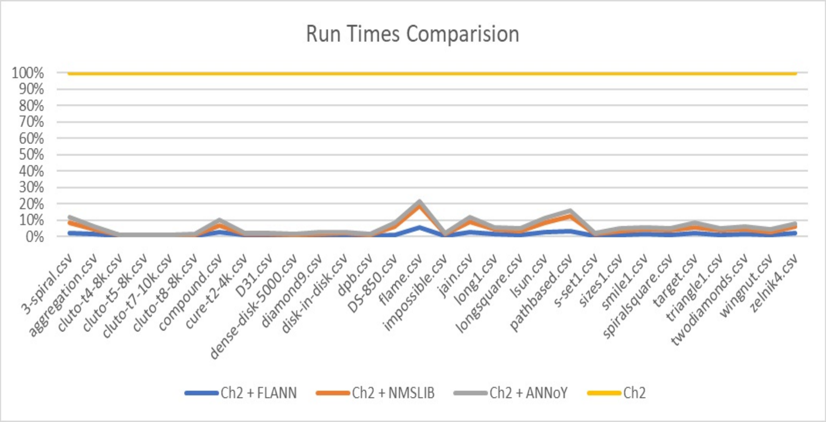


Table of Contents

Future Work

- We will adapt systematic process to fine tune the parameters and try to register the best results possible.
- To validate the efficacy of our methodologies even further, we will employ statistical evaluations, specifically utilizing the paired-test and z-test.
- Finally, if time permits we aim to refine or re-define and deploy a novel merging phase in our research.

References I

[1] Barton, T., Bruna, T., and Kordik, P. Chameleon 2: an improved graph-based clustering algorithm. *ACM Transactions on Knowledge Discovery from Data (TKDD) 13*, 1 (2019), 1–27.

[2] Friedman, J. H., Bentley, J. L., and Finkel, R. A. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software (TOMS) 3*, 3 (1977), 209–226.

[3] Karypis, G., Han, E.-H., and Kumar, V. Chameleon: Hierarchical clustering using dynamic modeling. *computer 32*, 8 (1999), 68–75.

References II

[4] Li, W., Zhang, Y., Sun, Y., Wang, W., Li, M., Zhang, W., and Lin, X. Approximate nearest neighbor search on high dimensional data—experiments, analyses, and improvement. *IEEE Transactions on Knowledge and Data Engineering 32*, 8 (2019), 1475–1488.

[5] Malkov, Y., Ponomarenko, A., Logvinov, A., and Krylov, V. Approximate nearest neighbor algorithm based on navigable small world graphs. *Information Systems 45* (2014), 61–68.

Thank You