International Journal On Engineering Technology and Sciences – IJETS™
ISSN (P): 2349-3968, ISSN (O): 2349-3976
Volume 1 Issue 6, October 2014

Optimal Clustering with Nearest Neighbor Relationships

Ms. N. Narmatha
II Year M.E(CSE)
Shree Venkateshwara Hi-Tech Engg College, Gobi
narmatha.nallasamy@gmail.com

Dr. T. Senthil Prakash
Professor & HOD
Shree Venkateshwara Hi-Tech Engg College, Gobi
jtyesp@yahoo.co.in

Ms.S. Kokila
II Year M.E(CSE)
Shree Venkateshwara Hi-Tech Engg College, Gobi
kokiyellowever@gmail.com

ABSTRACT- *The hubs are data points frequently occur in K-Nearest Neighbor (KNN) lists of other points. Hubness model uses the hubs relationship in data partitioning process. Hubness indicates the point centrality with in a high dimensional data cluster. Cluster prototypes are identified using the hub information. Hubness score is estimated using the frequency of items that are retrieved from the KNN queries. K Hubs algorithm is used to initialize the hubs for the clusters. Hubness proportional clustering (HPC) algorithm is used group the probabilistic data models. Hubness-proportional K-means (HPKM) algorithm integrates the hubness based centroid selection and partitioning process. The hubness based clustering scheme is improved with hubness score interval analysis mechanism. Kernel mapping scheme is enhanced with hubness relationship analysis. Shared neighbor clustering is integrated with the hubness mechanism. Hub based automatic cluster count estimation mechanism is integrated with the system.*

Keywords*: Hubness, K-Nearest Neighbor, Hubness proportional clustering, Kernel mapping.*

1 INTRODUCTION

Clustering is the task of assigning a set of objects into groups so that the objects in the same cluster are more similar to each other than to those in other clusters. Clustering is a main task of explorative data mining, and a common technique for statistical data analysis used in many fields, including machine learning, pattern recognition, image analysis, information retrieval, and bioinformatics. The notion of a cluster varies between algorithms and is one of the many decisions to take when choosing the appropriate algorithm for a particular problem. At first the terminology of a cluster seems obvious: a group of data objects. However, the clusters found by different algorithms vary significantly in their properties, and understanding these cluster models is key to understanding the differences between the various algorithms.

Connectivity models for example hierarchical clustering builds models based on distance connectivity. Centroid models for example the K-means algorithm represents each cluster by a single mean vector [1]. Distribution models clusters are modeled using statistic distributions, such as multivariate normal distributions used by the Expectation-maximization algorithm. Density models for example DBSCAN and OPTICS defines clusters as connected dense regions in the data space. Subspace models in Biclustering, clusters are modeled with both cluster members. Group models algorithms do not provide a refined model for their results and just provide the grouping information. A clustering is essentially a set of such clusters, usually containing all objects in the data set. Additionally, it may specify the relationship of the clusters to each other, a hierarchy of clusters embedded in each other. Clustering's can be roughly distinguished in hard clustering and soft clustering. Hard clustering object belongs to a cluster or not. Soft clustering object belongs to each cluster to a certain degree.

2 RELATED WORK

Even though hubness has not been given much attention in data clustering, hubness information is drawn from k nearest- neighbor lists, which have been used in the past to perform clustering in various ways. These lists may be used for computing density estimates, by observing the volume of space determined by the k-nearest neighbors. Density based clustering methods often rely on this kind of density estimation. The implicit assumption made by density-based algorithms is that clusters exist as high density regions separated from each other by low-density regions. In high-dimensional spaces this is often difficult to estimate, due to data being very sparse. There is also the issue of choosing the proper neighborhood size, since both small and large values of k can cause problems for density based approaches.

Enforcing k-nearest-neighbor consistency in algorithms such as K-means was also explored. The most typical usage of k-nearest-neighbor lists,

however, is to construct a k-NN graph [7] and reduce the problem to that of graph clustering. Consequences and applications of hubness have been more thoroughly investigated in other related fields: classification [10], [4], image feature representation [5], data reduction [6], collaborative filtering text retrieval and music retrieval [11]. In many of these studies it was hubs can offer valuable information that can be used to improve existing methods and devise new algorithms for the given task.

The interplay between clustering and hubness was briefly examined, where it was observed that hubs may not cluster well using conventional prototype-based clustering algorithms, since they not only tend to be close to points belonging to the same cluster but also tend to be close to points assigned to other clusters. Hubs can be viewed as analogues of outliers have high inter- and intracluster distance, suggesting that hubs should also receive special attention. In this paper, we have adopted the approach of using hubs as cluster prototypes and/or guiding points during prototype search.

3 HIGH-DIMENSIONAL DATA CLUSTERING

Clustering in general is an unsupervised process of grouping elements together, so that elements assigned to the same cluster are more similar to each other than to the remaining data points. This goal is often difficult to achieve in practice. Over the years, various clustering algorithms have been proposed, which can be roughly divided into four groups: partitional, hierarchical, densitybased, and subspace algorithms. Algorithms from the fourth group search for clusters in some lower dimensional projection of the original data, and have been generally preferred when dealing with data that are high dimensional. The motivation for this preference lies in the observation that having more dimensions usually leads to the so-called curse of dimensionality, where the performance of many standard machine-learning algorithms becomes impaired. This is mostly due to two pervasive effects: the empty space phenomenon and concentration of distances. The former refers to the fact that all high-dimensional data sets tend to be sparse, because the number of points required to represent any distribution grows exponentially with the number of dimensions. This leads to bad density estimates for high-dimensional data, causing difficulties for density-based approaches. The latter is a somewhat counterintuitive property of high-dimensional data representations, where all distances between data points tend to become harder to distinguish as dimensionality increases, which can cause problems with distance-based algorithms [9].

The difficulties in dealing with high-dimensional data are omnipresent and abundant. However, not all phenomena that arise are necessarily detrimental to clustering techniques. We will show in this paper that hubness, which is the tendency of some data points in high-dimensional data sets to occur much more frequently in k-nearest neighbor lists of other points than the rest of the points from the set, can in fact be used for clustering. To our knowledge, this has not been previously attempted. In a limited sense, hubs in graphs have been used to represent typical word a meaning was not used for data clustering. A similar line of research has identified essential proteins as hubs in the reverse nearest neighbor topology of protein interaction networks. We have focused on exploring the potential value of using hub points in clustering by designing hubness-aware clustering algorithms and testing them in a high-dimensional context.

There are two main contributions of this paper. First, in experiments on synthetic data we show that hubness is a good measure of point centrality within a high-dimensional data cluster and that major hubs can be used effectively as cluster prototypes. In addition, we propose three new clustering algorithms and evaluate their performance in various high-dimensional clustering tasks. We compared the algorithms with baseline state-of-the-art prototypebased method kernel-based and density-based approaches. The evaluation shows that our algorithms frequently offer improvements in cluster quality and homogeneity. The comparison with kernel Kmeans reveals that kernel-based extensions of the initial approaches should also be considered in the future. Our current focus was mostly on properly selecting cluster prototypes, with the proposed methods tailored for detecting approximately hyperspherical clusters.

4 PROBLEM STATEMENT

The hubs are data points frequently occur in K-Nearest Neighbor (KNN) lists of other points. Hubness model uses the hubs relationship in data partitioning process. Hubness indicates the point centrality with in a high dimensional data cluster. Cluster prototypes are identified using the hub information. Hubness score is estimated using the frequency of items that are retrieved from the KNN queries. K Hubs algorithm is used to initialize the hubs for the clusters. Hubness proportional clustering (HPC) algorithm is used group the probabilistic data models. Hubness-proportional K-means (HPKM) algorithm integrates the hubness based centroid selection and partitioning process. The following problems are identified from the existing system.

- The system performs the clustering using predefined cluster count
- The system fetches hyper spherical clusters only
- Inter cluster distance is not optimized
- Clustering accuracy is low

5 HUB-BASED CLUSTERING

Hubness is an aspect of the curse of dimensionality pertaining to nearest neighbors which has only recently come to attention, unlike the much discussed distance concentration phenomenon. Let D $\subset IR^d$ be a set of data points and let $N_k(x)$ denote the number of k-occurrences of point x $\in$ D, i.e., the number of times x occurs in k-nearest neighbor lists of other points from D. As the dimensionality of data increases, the distribution of k-occurrences becomes considerably skewed [3]. As a consequence, some data points, which we will refer to as hubs, are included in many more k-nearest-neighbor lists than other points. In the rest of the text, we will refer to the number of k-occurrences of point x $\in$ D as its hubness score. It has been shown that hubness, as a phenomenon, appears in high-dimensional data as an inherent property of high dimensionality, and is neither an artifact of finite samples nor a peculiarity of some specific data sets. Naturally, the exact degree of hubness may still vary and is not uniquely determined by dimensionality.

A Emergence of Hubs

Hubs also exist in clustered data, tending to be situated in the proximity of cluster centers. In addition, the degree of hubness does not depend on the embedding dimensionality, but rather on the intrinsic data dimensionality, which is viewed as the minimal number of variables needed to account for all pairwise distances in the data.

Generally, the hubness phenomenon is relevant to (intrinsically) high-dimensional data regardless of the distance or similarity measure employed. Its existence was verified for euclidean (l2) and Manhattan (l1) distances, lp distances with p > 2, fractional distances (l_p with rational p $\in$ (0; 1)), Bray-Curtis, normalized euclidean, and Canberra distances, cosine similarity, and the dynamic time warping distance for time series [12], [8]. In this paper, unless otherwise stated, we will assume the euclidean distance. The methods depend mostly on neighborhood relations that are derived from the distance matrix and are independent of the particular choice of distance measure. Hubs are points x having Nk(x) more than two standard deviations higher than the expected value k. In most experiments that follow, we will only concern ourselves with one major hub in each cluster, i.e., the point with the highest hubness score.

B Hubs and Data Clusters

There has been previous work on how well high-hubness elements cluster, as well as the general impact of hubness on clustering algorithms. A correlation between lowhubness elements and outliers was also observed. A low-hubness score indicates that a point is on average far from the rest of the points and hence probably an outlier. In high-dimensional spaces, however, low-hubness elements are expected to occur by the very nature of these spaces and data distributions. These data points will lead to an average increase in intracluster distance. It was also shown for several clustering algorithms that hubs do not cluster well compared to the rest of the points. This is due to the fact that some hubs are actually close to points in different clusters. Hence, they lead to a decrease in intercluster distance. This has been observed on real data sets clustered using state-of-the-art prototypebased methods and was identified as a possible area for performance improvement.

Points closer to cluster means tend to have higher hubness scores than other points. A natural question which arises is: Are hubs medoids? When observing the problem from the perspective of partitioning clustering approaches, of which K-means is the most commonly used representative, a similar question might also be posed: Are hubs the closest points to data centroids in clustering iterations? To answer this question, we ran K-means++ multiple times on several randomly generated 10,000-point Gaussian mixtures for various fixed numbers of dimensions (2, 5, 10, 20, 30, 50, 100), observing the high-dimensional case. We measured in each iteration the distance from current cluster centroid to the medoid and to the strongest hub, and scaled by the average intracluster distance. This was measured for every cluster in all the iterations, and for each iteration the minimal and maximal distance from any of the centroids to the corresponding hub and medoid were computed.

C Deterministic Approach

A simple way to employ hubs for clustering is to use them as one would normally use centroids. In addition, this allows us to make a direct comparison with the K-means method. The algorithm, referred to as K-hubs, is given in

```
1. initializeClusterCenters();
2. Cluster[ ] clusters = formClusters();
3. repeat
4.     for all Cluster c ∈ clusters do
5.         DataPoint h = findClusterHub(c);
```

```
6.        setClusterCenter(c, h);
7.      end for
8.      clusters = formClusters();
9.    until noReassignments
10. return clusters
```

Algorithm 1.K-hubs.

After initial evaluation on synthetic data, it became clear that even though the algorithm manages to find good and even best configurations often, it is quite sensitive to initialization. To increase the probability of finding the global optimum, we resorted to the stochastic approach. Even though K-hubs exhibited low stability, it converges to cluster configurations very quickly, in no more than four iterations on all the data sets used for testing, most of which contained around 10,000 data instances.

D Probabilistic Approach

Even though points with highest hubness scores are without doubt the prime candidates for cluster centers, there is no need to disregard the information about hubness scores of other points in the data. In the algorithm described below, we implemented a squared hubness-proportional stochastic scheme based on the widely used simulated annealing approach to optimization. The temperature factor was introduced to the algorithm, so that it may start as being entirely probabilistic and eventually end by executing deterministic K-hubs iterations. We will refer to this algorithm, specified by Algorithm 2, as hubnessproportional clustering (HPC).

```
1. initializeClusterCenters();
2. Cluster[] clusters = formClusters();
3. float t = t0; initialize temperature
4. repeat
5.   float θ getProbFromSchedule(t);
6.   for all Cluster c ∈ clusters do
7.     if randomFloat(0,1) < θ then
8.       DataPoint h = findClusterHub(c);
9.       setClusterCenter(c, h);
10.else
11.       for all DataPoint x ∈ c do
12.            etChoosingProbability(x, N_k^2 (x));
13.       end for
14.       normalizeProbabilities();
15.      DataPoint h = chooseHubProbabilistically(c);
16.       setClusterCenter(c, h);
17.    end if
18. end for
19.  clusters =formClusters();
20.   t = updateTemperature(t);
21. until noReassignments
22. return clusters
```

Algorithm 2. HPC.

The reason why hubness-proportional clustering is feasible in the context of high dimensionality lies in the skewness of the distribution of k-occurrences. Namely, there exist many data points having low hubness scores, making them bad candidates for cluster centers. Such points will have a low probability of being selected. To further emphasize this, we use the square of the actual hubness score instead of making the probabilities directly proportional to $N_k(x)$.

The HPC algorithm defines a search through the data space based on hubness as a kind of a local centrality estimate. To justify the use of the proposed stochastic scheme, we executed a series of initial tests on a synthetic mixture of Gaussians, for dimensionality d =50, n = 10; 000 instances, and K =25 clusters in the data. Neighborhood size was set to k = 10 and for each preset number of probabilistic iterations in the annealing schedule, the clustering was run 50 times, each time reinitializing the seeds. The silhouette index was used to estimate the clustering quality. Due to the significant skewness of the squared hubness scores, adding more probabilistic iterations helps in achieving better clustering, up to a certain plateau that is eventually reached. The same shape of the curve also appears in the case of not taking the last, but the error minimizing configuration.

E A Hybrid Approach

The algorithms do not require knowledge of data/object representation, so all that is required is a distance/similarity measure defined for each pair of data objects. However, if the representation is also available such that it is possible to meaningfully calculate centroids, there also exists a third alternative: use point hubness scores to guide the search, but choose a centroid-based cluster configuration in the end. We will refer to this algorithm as hubness-proportional K-means (HPKM). It is nearly identical to HPC, the only difference being in the deterministic phase of the iteration, as the configuration cools down during the annealing procedure: instead of reverting to K-hubs, the deterministic phase executes K-means updates.

```
1. initializeClusterCenters();
2. Cluster[] clusters = formClusters();
3. float t = t0; {initialize temperature}
4. repeat
5.   float θ = getProbFromSchedule(t);
6.   for all Cluster c ∈ clusters do
7.     if randomFloat(0, 1) < θ then
8.       DataPoint h = findClusterCentroid(c);
9.       setClusterCenter(c, h);
```

```
10. else
11.   for all DataPoint x ∈ c do
12.   setChoosingProbability(x, N_k^2 (x));
13.   end for
14.   normalizeProbabilities();
15.   DataPoint h = chooseHubProbabilistically(c);
16.   setClusterCenter(c, h);
17.   end if
18. end for
19.  clusters = formClusters();
20.  t = updateTemperature(t);
21. until noReassignments
22. return clusters
```

Algorithm 3. HPKM.

There are, indeed, cases when HPKM might be preferable to the pure hubness-based approach of K-hubs and HPC. Even though our initial experiments suggest that the major hubs lie close to local cluster means in high dimensional data, there is no guarantee that this would hold for every cluster in every possible data set. It is reasonable to expect there to be distributions which lead to such local data structure where the major hub is not among the most central points. Also, an ideal cluster configuration on a given real-world data set is sometimes impossible to achieve by using points as centers, since centers may need to be located in the empty space between the points.

6 OPTIMAL CLUSTERING WITH NEAREST NEIGHBOR RELATIONSHIPS

The hubness based clustering scheme is improved with hubness score interval analysis mechanism. Kernel mapping scheme is enhanced with hubness relationship analysis. Shared neighbor clustering is integrated with the hubness mechanism. Hub based automatic cluster count estimation mechanism is integrated with the system. The clustering scheme is designed to partition the records with predefined and optimized cluster count values. Cluster centroid intervals are analyzed in the initialization process. Hubness measure is applied in the shared neighbor clustering and kernel mapping models. The system is divided into six major modules. They are data preprocess, KNN Search, hubness measure estimation, HPKM clustering process, clustering with kernel mapping scheme and shared neighbor clustering process.

Data preprocess module is designed to perform noise elimination process. Nearest neighbor data values are identified using the KNN search module. Hubness score values are estimated using the KNN search results. Hubness-proportional K-means (HPKM) Clustering process is applied to partition the data. Data clustering is performed with the Kernel mapping mechanisms. Shared neighbor clustering is performed with hubness relationships.

The nearest neighbor based clustering process is defined in algorithm 1.. Diagnosis data, cluster type, cluster count and K value are used as input for the algorithm. KNN query results and clustering results are produced as output. Data cleaning process is performed on the diagnosis data (D). The KNN query is applied on the cleaned data (CD) values. Hubness measure is estimated using the KNN query results. The system performs the clustering process using Hubness-proportional K-means (HPKM), Kernel Mapping based Clustering (KMC) and Shared Neighbor Clustering (SNC) techniques.

```
Begin
        Input D, ctype, ccount and k
        Output cresults
        CD = clean (D) // Data cleaning process
        R = KNN query (CD) // KNN query process
        estimate hubness score
        estimate the centroid
        if ctype = 1 call HPKM // hubness-proportional K-means
        if ctype = 2 call KMC // Kernel Mapping based Clustering
        if ctype = 3 call SNC // Shared Neighbor Clustering
 end
```

Algorithm 4. Nearest Neighbor Relationships based Clustering Process Algorithm

A Data Preprocess

Data cleaning is performed in the data preprocess. Data values are parsed and updated into the database. Redundant data values are removed from the data set. Missing values are assigned with suitable values.

B KNN Search

Nearest neighbor transactions are identified in the KNN search process. K Nearest Neighbor (KNN) search algorithm is used for the nearest neighbor identification process. KNN search process is applied for all the transactions. Similarity measures are used in the KNN search process.

C Hubness Measure Estimation

Hubness measure estimation is applied for all the transactions. KNN search results are used in the hubness measure estimation process. Hubness measure is assigned for each transaction data. Hubness measure is used for the centroid selection process.

D HPKM Clustering Process

Hubness-proportional K-means (HPKM) algorithm is used for the data partitioning process.

Clustering process is performed in two ways. Predefined cluster count based data partitioning uses the cluster count collected from the user. Optimal cluster count estimation mechanism is used to identify the feasible cluster count automatically.

E Clustering With Kernel Mapping Scheme

The Kernel mapping scheme is used for the data clustering process. Hubness measure is used in the Kernel mapping scheme. Kernel mapping clustering is performed with predefined and optimal cluster count values. Hubness measure interval is used for the cluster count estimation process.

F Shared Neighbor Clustering Process

The clustering process is performed with shared neighbor clustering mechanism. Shared neighbor clustering process is improved with hubness measures. Distance measures are used to estimate the shared neighbor data values. Optimal cluster count based clustering model improves the clustering accuracy levels.

7 CONCLUSION

Clustering techniques are used to partition the transactions based on the relationships. Hubness measures are used to select the cluster centrality data points. Hubness-proportional K-means (HPKM) algorithm performs the clustering process with predefined cluster count. Kernel mapping model and shared neighbor clustering algorithm are enhanced with hubness mechanism. The system improves the cluster accuracy levels for high dimensional data environment. Optimal inter and intra cluster distances are considered in the centroid estimation process. The system improves the accuracy in quantitative noisy environment. Automated cluster count estimation mechanism is adopted for all clustering technique. The system supports high scalability in the clustering process.

REFERENCES

[1] Qinpei Zhao and Pasi Fränti, "Centroid Ratio for a Pairwise Random Swap Clustering Algorithm" IEEE Transactions On Knowledge And Data Engineering, May 2014

[2] Sicheng Xiong and Xiaoli Fern, "Active Learning of Constraints for Semi-Supervised Clustering" IEEE Transactions On Knowledge And Data Engineering, January 2014

[3] M. Ivanovic, "Hubs in Space: Popular Nearest Neighbors in High-Dimensional Data," J. Machine Learning Research, 2010.

[4] N. Tomasev and D. Mladenic, "Nearest Neighbor Voting in High Dimensional Data: Learning from Past Occurrences," Computer Science and Information Systems, vol. 9, no. 2, 2012.

[5] N. Tomasev and S. Nedevschi, "The Influence of Hubness on Nearest-Neighbor Methods in Object Recognition," Proc. IEEE Seventh Int'l Conf. Intelligent Computer Comm. and Processing, 2011.

[6] K. Buza, A. Nanopoulos and L. Schmidt-Thieme, "INSIGHT: Efficient and Effective Instance Selection for Time-Series Classification," Proc. 15th Pacific-Asia Conf. Knowledge Discovery and Data Mining (PAKDD), Part II, pp. 149-160, 2011.

[7] C.-T. Chang, J.Z.C. Lai, and M.D. Jeng, "Fast Agglomerative Clustering Using Information of k-Nearest Neighbors," Pattern Recognition, 2010.

[8] M. Radovanovic and Ivanovic, "On the Existence of Obstinate Results in Vector Space Models," Proc. 33rd Ann. Int'l ACM SIGIR Conf. Research and Development in Information Retrieval, 2010.

[9] A. Kaban, "Non-Parametric Detection of Meaningless Distances in High Dimensional Data," Statistics and Computing, vol. 22, no. 2, 2012.

[10] N. Tomasev and M. Ivanovic, "Hubness-Based Fuzzy Measures for High-Dimensional k-Nearest Neighbor Classification," Proc. Seventh Int'l Conf. Machine Learning and Data Mining, pp. 16-30, 2011.

[11] D. Schnitzer, A. Flexer and G. Widmer, "Local and Global Scaling Reduce Hubs in Space," J. Machine Learning Research, vol. 13, 2012.

[12] M. Radovanovic, A. Nanopoulos and M. Ivanovic, "Time-Series Classification in Many Intrinsic Dimensions," Proc. 10th SIAM Int'l Conf. Data Mining (SDM), pp. 677-688, 2010.

AUTHORS BIOGRAPHY

Ms.N.Narmatha pursuing M.E(CSE) degree from Shree Venkateshwara Hi-tech Engineering college, Erode, India in 2014 and B.Tech(IT) degree from Nandha Engineering college, Erode, India in 2011.she has attended national conference on MLP based intrusion detection using neural network and one workshop. Her research interests include data mining, software testing.

Dr.T.Senthil Prakash received the Ph.D. degree from the PRIST University, Thanjavur, India in 2013 and M.E(CSE) degree from Vinayaka Mission's University, Salem , India in 2007 and M.Phil.,MCA.,B.Sc(CS) degrees from Bharathiyar University, Coimbatore India, in 2000,2003 and 2006 respectively, all in Computer Science and Engineering. He is a Member in ISTE New Delhi, India, IAENG, Hong Kong..IACSIT, Singapore SDIWC, USA.

He has the experience in Teaching of 10+Years and in Industry 2 Years. Now He is currently working as a Professor and Head of the Department of Computer Science and Engineering in Shree Venkateshwara Hi-Tech Engineering College, Gobi, Tamil Nadu, and India. His research interests include Data Mining, Data Bases, Artificial Intelligence, Software Engineering etc.,He has published several papers in 17 International Journals, 43 International and National Conferences.

Stud.Ms. Ms.S.Kokila pursuing M.E(CSE) degree in Shree Venkateshwara Hi-Tech Engineering College, Erode, India in 2014 and B.E(CSE) degree from Velalar college of Engineering and Technology, Erode, India in 2012.She has published 1 National conferences, 4 workshops. She is a Member of Computer Society of India(CSI). Her research interests include Data mining, Databases.