Improving Clustering Performance on High Dimensional Data using Kernel Hubness

R.Shenbakapriya
PG Student, Department of Information Technology, SNS College of Technology, Coimbatore-35

M.Kalimuthu
Associate Professor, Department of Information Technology, SNS College of Technology, Coimbatore-35

P.Sengottuvelan
Associate Professor, Department of Information Technology, Bannari Amman Institute of Technology, sathyamangalam-401

Abstract
Clustering high dimensional data becomes difficult due to the increasing sparsity of such data. One of the inherent properties of high dimensional data is hubness phenomenon, which is used for clustering such data. Hubness is the tendency of high-dimensional data to contain points (hubs) that occurs frequently in k-nearest neighbor lists of other data points. The k-nearest-neighbor lists are used to measure the hubness score of each data point. The simple hub based clustering algorithms detect only hyperspherical clusters in the high dimensional dataset. But the real time high dimensional dataset contains more number of arbitrary shaped clusters. To improve the performance of clustering, a new algorithm is proposed which is based on the combination of kernel mapping and hubness phenomenon. The proposed algorithm detects arbitrary shaped clusters in the dataset and also improves the performance of clustering by minimizing the intra-cluster distance and maximizing the inter-cluster distance which improves the cluster quality.

Keywords
High dimensional data, hubness Phenomenon, Kernel mapping, and K-nearest neighbor.

1. INTRODUCTION
Clustering is an unsupervised process of grouping elements together. A cluster is a collection of data objects that are similar to one another within the same cluster and are dissimilar to the objects in other clusters. There are different clustering techniques available in the literature, such as hierarchical, partitional, and density-based and subspace [1]. Clustering methods can be used for detecting the underlying structure of the data distribution.

Partitional clustering methods start with an initial partition of the observation and optimize these partitions according to utility function or distance function. Hierarchical clustering methods works by grouping data objects into a tree of clusters. It can be further classified as either agglomerative or divisive, depending on whether the hierarchical decomposition is formed in a bottom-up or top-down fashion. Density-based clustering methods regard clusters as dense regions of objects in the data space that are separated by regions of low density. Subspace clustering methods search for groups of clusters within different subspaces of the same data set. This paper mainly focused on partitional clustering. To overcome the problems in partitional clustering methods on high dimensional data, a new algorithm which is based on combination of kernel mappings [6] and hubness phenomenon [4] was proposed.

The rest of the paper is structured as follows. In the next section we present the related work on this research, Section 3 presents the discussion of Kernel Principal Component Analysis, while Section 4 discusses the hubness phenomenon, Section 5 describes the kernel hubness clustering. Section 6 presents the experiments we performed on the real world dataset.

2. RELATED WORK
Applications of hubness have been investigated in other fields: classification, data reduction, image feature representation , text retrieval , collaborative filtering and music retrieval [4]. The emergence of hubs had been noted first in analyzing music collections. The researchers exposed some songs which were similar to many other songs, i.e., frequent neighbors.

In high dimensional data, it is difficult to estimate the separation of low density regions and high density regions due to data being very space [3],[5]. It is necessary to chose the proper neighborhood size, because both small and large values of k can cause problems for density based approaches.

Kernel k-means maps data points from the input space to the high dimensional feature space through a non-linear transformation [7]. The kernel based clustering minimizes the clustering error in feature space.

Hubs can approximate local data centers which lead to improvement over centroid-based approach [4]. Hub-based algorithms are specifically designed for high dimensional data. Kernel methods are much more powerful than other methods of clustering, because they can handle non-hyperspherical clusters.

It is believed that the kernel k-means, which is used with the non-parametric histogram intersection kernel [6], is good for image clustering. In this paper we have proposed a new clustering algorithm which uses the concept of kernel and hubness phenomenon.

3. KERNEL PRINCIPAL COMPONENT ANALYSIS
Linear Principal Component Analysis (PCA) projects high dimensional data onto a lower dimensional subspace by seeking a linear combination of a set of projection vectors that can best describe the variance of data in a sum of squared-error sense. Kernel PCA extends the capability of linear PCA by capturing nonlinear structure in the data [6], since a linear PCA performance in the feature space corresponds to a nonlinear projection in the original data space.

The basic steps for Kernel Principal Component Analysis are summarized in Fig. 1.

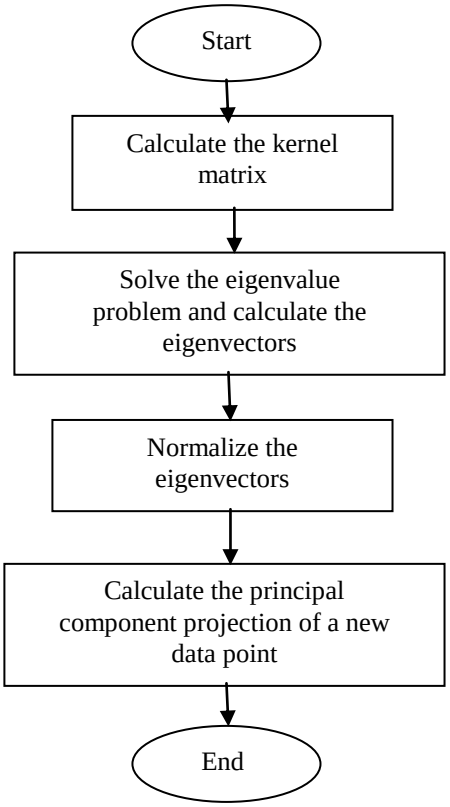

Fig 1: Kernel Principal Component Analysis

The nonlinearity is introduced by mapping the input data from the input space to a high-dimensional feature space.

3.1 Properties of Kernel PCA

1. Extends usual principal component analysis (PCA) to a high dimensional feature space using the "kernel trick".
2. Can extract up to n (number of samples) nonlinear principal components.
3. Kernel PCA can give a good reencoding of the data when it lies along a non-linear manifold.

4. HUBNESS PHENOMENON

Hubness is an act of high dimensional data to contain points that frequently occur in k-nearest neighbor lists of other points. Let $S \subset R^d$ be a set of high dimensional data points and let $N_k(y)$ denote the number of k-occurrences of point $y \in S$, i.e., the number of times y occurs in k-nearest neighbor lists of other points from S. Hubness is an inherent property of high dimensional data which is related to distance concentration phenomenon [4]. The number of k-occurrences of point $y \in S$ is referred as hubness score in rest of the text. The frequently occurring data points in k-neighbor sets are referred as hubs and very rarely occurring points are referred as anti-hubs.

4.1 Appearance of Hubs

The concentration of distances enables to view unimodal high dimensional data lying on a hypersphere centered at the data distribution mean. The variance of distances to the mean remains non-negligible for any countable number of dimensions, which indicates that some of the points still end up being closer to the data mean than other points [2]. The points closer to the mean tend to be closer to all other points in the dataset, for any dimensionality that observed. In high dimensional data, this act is made stronger. Such points will have a higher probability of being included in k-nearest neighbor sets of other points in the dataset, which increases their ability, and they emerge as neighbor-hubs.

Hubs can also exist in multimodal data, situated in the nearness of cluster centers [2]. The degree of hubness depends on the intrinsic data dimensionality, i.e., the number of variables needed to represent all pairwise distances in the data. Hubness phenomenon is related to high dimensional data regardless of the distance or similarity measure. The existence of hubs can be verified using Euclidean and Manhattan distances.

4.2 Relation of hub to centroid and medoid

In low dimensional data hubs in the clusters are far-off from the centroids, even out of average points. There is no relationship between cluster means and frequent neighbors in the low dimensional environment [2]. This fact may changes with the increase in dimensionality. The minimal distance from centroid to hub converges to minimal distance from centroid to medoid. This concept implies that some medoids are actually cluster hubs. As medoids the centroids are also closer to data hubs. This relationship brings us to get an idea that the points with high hubness scores are closer to centers of clustered sub regions of high dimensional space than other data points in the dataset. Hence these points can act as cluster representatives [4].

5. KERNEL BASED HUBNESS CLUSTERING

Hubness is viewed as a local centrality measure and is possible to use it for clustering high dimensional data in various ways. There are two types of hubness, namely global hubness and local hubness [2]. Local hubness can be defined as a restriction of global hubness on any given cluster of the current algorithm iteration. Local hubness score represents the number of k-occurrences of a point in k-nearest neighbor lists of elements within the same cluster. Global hubness represents the number of k-occurrences of a point in k-nearest neighbor lists of all elements of the dataset. This global hubness is used for determining the number of clusters automatically.

The high dimensional data contains more number of attributes, in which some attributes are more important for representing the data points. In order to identify the important attributes in the dataset, the Kernel Principal Component Analysis is used. The kernel principal components are used for defining the kernel function. By using the kernel function[6] , i.e., an appropriate non-linear mapping from the original input space to a higher dimensional feature space, clusters that are non-linearly separable in input space can be extracted. Kernel hubness clustering algorithm is described in Algorithm 1.

Algorithm 1 KHC

Input: Kernel matrix K, number of clusters k, initial clusters C_1, C_2, $C_3 \ldots, C_K$.
Output: Final clusters C_1, C_2, $C_3 \ldots, C_K$.

1: **for all** points x_n n=1,2,...,N **do**
2: **for all** clusters C_i i=1 to k **do**
3: Calculate distance between hub and other points using kernel function
4: **end for**

5: find optimal distance
6: **end for**
7: **for all** clusters C_i i=1 to k **do**
8: Update cluster C_i
9: **end for**
10: **if** converged **then**
11: **return** final clusters C_1, C_2, C_3…,C_K.
12: **else**
13: **goto** step1
14: **end if**

The number of clusters is determined automatically using hubness score of all points. The initial clusters are formed using kernel principal components and the hubness scores. The number of clusters and the initial clusters are used as input parameters for kernel hubness clustering algorithm.

6. EXPERIMENTS

Real world data is much more complex and difficult to cluster; therefore the experiments are made on these datasets. For experiment UCI iris dataset was taken. The proposed algorithm is executed on this dataset and the cluster quality is measured. The clustering quality in these experiments is measured by using two quality indices, namely silhouette index and isolation index. These indices measure a percentage of k-neighbor points that are clustered together.

The cluster visualization for basic hub based clustering algorithms and proposed algorithm is shown below.

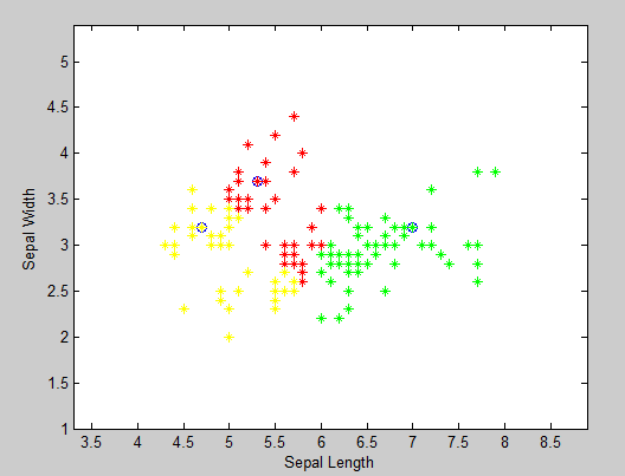

Fig 2: The visualization of clusters on iris data using HPC

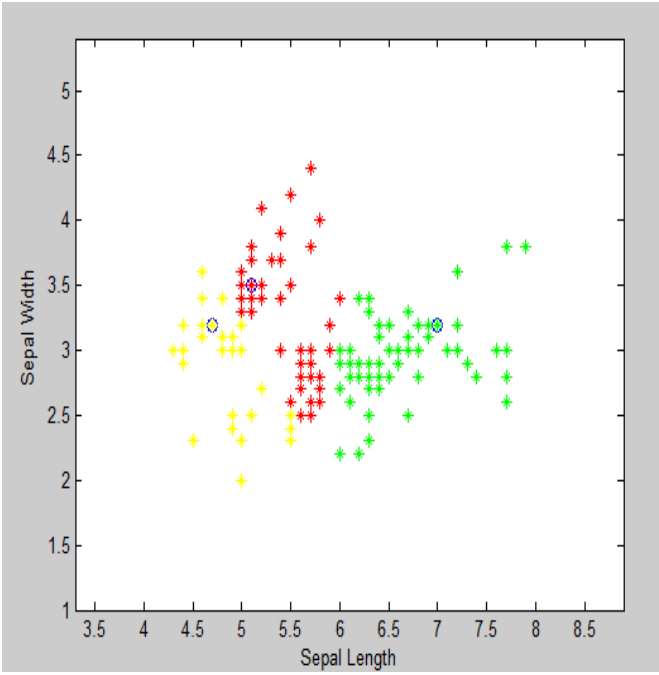

Fig 3: The visualization of clusters on iris data using HPKM

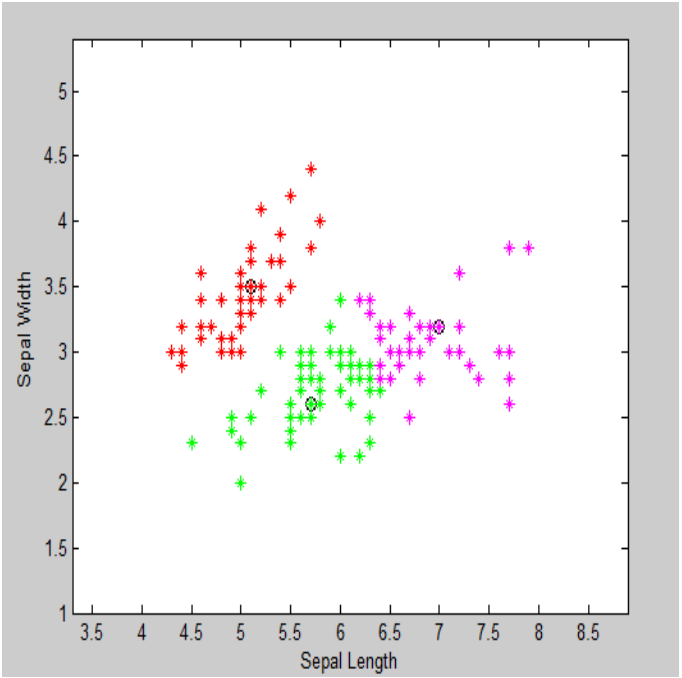

Fig 4: The visualization of clusters on iris data using KHC

7. CONCLUSION AND FUTURE WORK

The hubness phenomenon and kernel mapping has been used for clustering high dimensional data. Hubs are used to approximate local cluster prototypes is not only a feasible option, but also frequently leads to improvement over the centroid-based approach. Kernel hubness clustering algorithm is designed specifically for high dimensional data. This algorithm is expected to offer improvement by providing higher inter-cluster distance and lower intra-cluster distance. Since kernel mapping is applied, the algorithm detects arbitrary shaped clusters in the dataset. The hubs automatically determine the number of clusters to be formed. Hence users need not to specify the number of clusters as manually. In future the performance of clustering is improved by considering the time and iteration factors.

8. REFERENCES

[1] J. Han and M. Kamber (2006), "Data Mining: Concepts and Techniques," 2nd ed. Morgan Kaufmann Publishers.

[2] Miloˇs Radovanovi´c, Alexandros Nanopoulos, and Mirjana Ivanovi´c (2010), "Hubs in Space: Popular Nearest Neighbors in High-Dimensional Data," Journal of Machine Learning Research, pp. 2487-2531.

[3] N. Tomaˇsev and D. Mladeni´c (2012), "Nearest neighbor voting in high dimensional data: Learning from past occurrences," Computer Science and Information Systems, vol. 9, no. 2, pp. 691–712.

[4] N. Tomasev, M. Radovanovic, D. Mladenic, M. Ivanovic (2013), "The Role of Hubness in Clustering High-Dimensional data," IEEE Transactions on Knowledge and Data Engineering, vol:pp, issue:99, ISSN:1041-4347.

[5] N. Tomasev, R. Brehar, D. Mladenic, and S. Nedevschi (2011), "The influence of hubness on nearest-neighbor methods in object recognition," in Proc. 7th IEEE Int. Conf. on Intelligent Computer Communication and Processing (ICCP), pp. 367–374

[6] Grigorios F. Tzortzis and Aristidis C. Likas,*(2009),* "The Global Kernel K-Means Algorithm for Clustering in Feature Space" IEEE Transactions on Neural Networks, Vol. 20. No. 7,PP:1181-1194.

[7] I. S. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means: spectral clustering and normalized cuts," in Proc. 10th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, 2004, pp. 551–556.

[8] C.-T. Chang, J. Z. C. Lai, and M. D. Jeng (2010), "Fast agglomerative clustering using information of k-nearest neighbors," Pattern Recognition, vol. 43, no. 12, pp. 3958–3968.

[9] R. Xu, D. Wunsch (2005), "Survey of clustering algorithms," IEEE Transactions on Neural Networks 16 (3) pp. 645–678.

[10] Nanopoulos A., M. Radovanovi´c, and M. Ivanovi´c (2009), "How does high dimensionality affect collaborative filtering?" in Proc. 3rd ACM Conf. on Recommender Systems (RecSys), pp. 293–296.

[11] A.K. Jain, M.N. Murty, P.J. Flynn (1999), "Data clustering: a review," ACM Computing Surveys 31 (3) pp. 264–323.

[12] E. Plaka and L. E. Kavraki (2007), "Distributed computation of the Knn graph for large high dimensional point sets," Journal of Parallel and DistributeComputing, 67(3): 346-359.