The Role of Hubness in Clustering High-Dimensional Data

Nenad Tomašev[1], Miloš Radovanović[2], Dunja Mladenić[1], and Mirjana Ivanović[2]

[1] Institute Jožef Stefan
Department of Knowledge Technologies
Jamova 39, 1000 Ljubljana, Slovenia
nenad.tomasev@ijs.si, dunja.mladenic@ijs.si
[2] University of Novi Sad
Department of Mathematics and Informatics
Trg D. Obradovića 4, 21000 Novi Sad, Serbia
radacha@dmi.uns.ac.rs, mira@dmi.uns.ac.rs

Abstract. High-dimensional data arise naturally in many domains, and have regularly presented a great challenge for traditional data-mining techniques, both in terms of effectiveness and efficiency. Clustering becomes difficult due to the increasing sparsity of such data, as well as the increasing difficulty in distinguishing distances between data points. In this paper we take a novel perspective on the problem of clustering high-dimensional data. Instead of attempting to avoid the curse of dimensionality by observing a lower-dimensional feature subspace, we embrace dimensionality by taking advantage of some inherently high-dimensional phenomena. More specifically, we show that hubness, i.e., the tendency of high-dimensional data to contain points (hubs) that frequently occur in k-nearest neighbor lists of other points, can be successfully exploited in clustering. We validate our hypothesis by proposing several hubness-based clustering algorithms and testing them on high-dimensional data. Experimental results demonstrate good performance of our algorithms in multiple settings, particularly in the presence of large quantities of noise.

1 Introduction

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 [1]. 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*, *density-based*, 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 is high-dimensional [2–5]. 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 give rise to problems with distance-based algorithms [6–8].

The difficulties in dealing with high-dimensional data are omnipresent and abundant. However, not all phenomena which 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 meanings in [9]. This, however, was not used for data clustering. Therefore, we focused first of all on exploring the potential value of using hub points in clustering by constructing hubness-based clustering algorithms and testing them in high-dimensional settings. The hubness phenomenon and its relation to clustering will be further addressed in Section 3.

The rest of the paper is structured as follows. In the next section we present related work, Section 3 discusses in general the phenomenon of hubness, while Section 4 describes the proposed algorithms that are exploiting hubness for data clustering. Section 5 presents the experiments we performed on both synthetic and real world data, and in Section 6 we give our final remarks.

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 [10–12]. 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 [13]. Enforcing k-nearest-neighbor consistency in algorithms such as K-means was also experimented with [14]. This approach proposed moving closed neighbor-sets between clusters in iterations instead of using single data points. However, the most typical usage of k-nearest-neighbor lists relates to constructing a k-NN graph, where nodes are connected by an edge if one of them is in the k-nearest-neighbor list of the other [15]. The problem is then reduced to graph clustering, with a number of approaches available.

3 The Hubness Phenomenon

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 $D \subset \mathbb{R}^d$ be a set of data points and let $N_k(x)$ denote the

number of k-*occurrences* of point x, 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 [16]. 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. Moreover, 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 appears in high-dimensional data as an inherent property of high dimensionality, and is not an artefact of finite samples nor a peculiarity of some specific data sets [16].

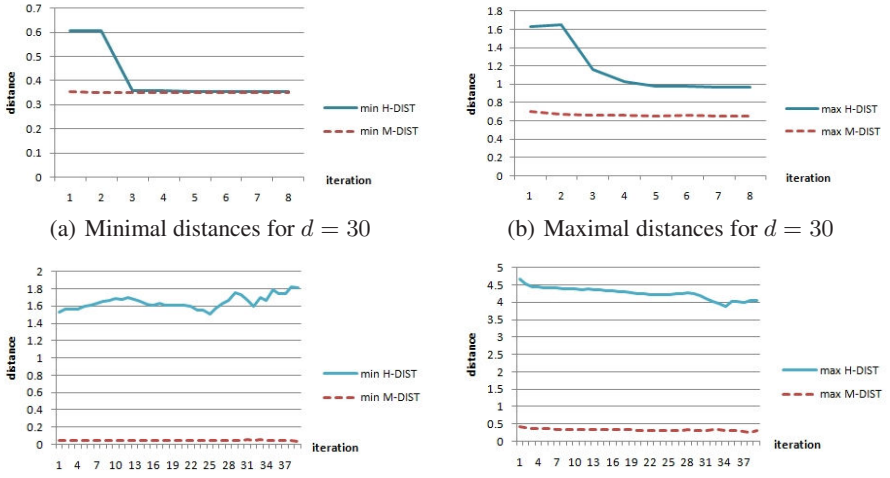
3.1 The Emergence of Hubs

Hubness is closely related to the aforementioned concentration of distances in high-dimensional spaces. If distances do concentrate for a given data set, then its points are lying approximately on a hypersphere centered at the data mean. Naturally, if data is drawn from several distributions, as is usually the case in clustering problems, this could be rephrased by saying that data are lying approximately on several hyperspheres centered at the corresponding distribution means. However, it has been shown that the variance of distances to the mean is still non-negligible, regardless of the concentration phenomenon – for any finite number of dimensions [7]. This implies that some of the points will still end up being closer to the data (or cluster) mean than other points. It is well known that points closer to the mean tend to, on average, be closer to all other points, for any observed dimensionality. However, in high-dimensional data, this tendency is amplified [16]. On average, points which are closer to all other points will naturally have a higher probability of being included in k-nearest-neighbor lists of other points in the data set, which gives rise to an increase in their hubness scores.

3.2 Relation of Hubs to Data Clusters

There has been some previous work on how well high-hubness elements cluster, as well as the general impact of hubness on clustering algorithms [16]. A correlation between low-hubness 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 intra-cluster dissimilarity. 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 also lead to a decrease in inter-cluster dissimilarity. However, this does not necessarily hold for an arbitrary cluster configuration.

It was already mentioned that points closer to cluster means tend to have higher hubness scores than the rest of the 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++ [17] multiple times on several randomly generated Gaussian mixtures for various fixed numbers of dimensions,



(a) Minimal distances for $d = 30$ (b) Maximal distances for $d = 30$

(c) Minimal distances for $d = 2$ (d) Maximal distances for $d = 2$

Fig. 1. Evolution of minimal and maximal distances from cluster centroids to hubs and medoids on synthetic data for neighborhood size 10 in case of 10 data clusters

observing the high-dimensional case. We measured in each iteration the distance from current cluster centroid to the medoid and to the hub, and scaled by the average intra-cluster 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. Figure 1 gives example plots of how these ratios evolve through iterations for the case of 10-cluster data, used neighborhood size 10, with 30 dimensions for the high-dimensional case, and 2 dimensions to illustrate low-dimensional behavior.

It can be noticed from the charts that, in the low-dimensional case, hubs in the clusters are far away from the centroids, even farther than average points. There is no correlation between data means and high-hubness instances in the low-dimensional scenario. On the other hand, for the high-dimensional case, we observe that the minimal distance from centroid to hub converges to minimal distance from centroid to medoid. This implies that some medoids are in fact cluster hubs. Maximal distances to hubs and medoids, however, do not match. There exist hubs which are not medoids, and vice versa. Also, we observe that maximal distance to hubs also drops with iterations, hinting that as the iterations progress, centroids are becoming closer and closer to data hubs. This brings us to the idea that will be explained in detail in the following section: *Why not use hubs to approximate data centers?* After all, we expect points with high hubness scores to be closer to centers of relatively dense regions in high-dimensional spaces than the rest of the data points, making them viable candidates for representative cluster elements. We are not limited to observing only the points with the highest hubness scores, we can also take advantage of hubness information for any given data point. More generally, in case of irregularly shaped clusters, hubs are expected to be found near the centers of compact sub-clusters, which is also beneficial.

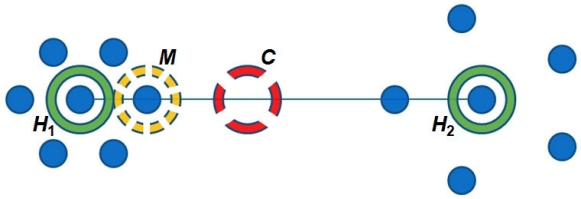


Fig. 2. Illustrative example. The red dashed circle marks the centroid (C), yellow dotted circle the medoid (M), and green circles denote two elements of highest hubness (H_1, H_2), for neighborhood size 3

4 Hub-Based Clustering

If hubness is viewed as a kind of local centrality measure, it may be possible to use hubness for clustering in various ways. In order to test this hypothesis, we opted for an approach that allows observations about the quality of resulting clustering configurations to be related directly to the property of hubness, instead of being a consequence of some other attribute of the clustering algorithm. Since it is expected of hubs to be located near the centers of compact sub-clusters in high-dimensional data, a natural way to test the feasibility of using them to approximate these centers is to compare the hub-based approach with some centroid-based technique. For that reason, the considered algorithms are made to resemble K-means, by being iterative approaches for defining clusters around separated high-hubness data elements.

As Fig. 1 showed, centroids and medoids in K-means iterations tend to converge to locations close to high-hubness points. This implies that using hubs instead of either of these could actually speed up the convergence of the algorithms leading it straight to the promising regions in the data space. To illustrate this point, consider the simple example shown in Fig. 2, which mimics in two dimensions what normally happens in multidimensional data, and suggests that not only might taking hubs as centers in following iterations provide quicker convergence, but that it also might prove helpful in finding the best end configuration. Centroids depend on all current cluster elements, while hubs depend mostly on their neighboring elements and therefore carry *local* centrality information. We will consider two types of hubness below, namely *global* hubness and *local* hubness. We define local hubness as a restriction of global hubness on any given cluster, considered in the context of the current algorithm iteration. Hence, the local hubness score represents the number of k-occurrences of a point in k-nearest-neighbor lists of elements from within the same cluster.[3]

The fact that hubs emerge close to centers of dense subregions might suggest some sort of a relationship between hubness and the density estimate at the observed data point. There are, however, some important differences. First of all, hubness does not depend on scale. Let D_1 and D_2 be two separate sets of points. If the local distance matrices defined on each of them separately are proportional, we might think of D_1 and D_2 as two copies of the same abstract data model appearing at different scales. Even

[3] Henceforth, we will use the capitalized K to represent the desired number of clusters and small k for neighborhood size.

though the density estimate might be significantly different, depending on the defining volumes which are affected by scale, there will be a perfect match in hubness scores of the corresponding points. However, there is a more subtle difference. Let $D_k(x)$ be the set of points where x is among the k nearest neighbors. Hence, the hubness score of x is then given by $N_k(x) = |D_k(x)|$. For each $x_i \in D_k(x)$, whether point x is among the k nearest neighbors of x_i depends on two things: $distance(x, x_i)$, and the density estimate at point x_i, not the density estimate at point x. Consequently, a hub might be a k-neighbor for points where density is high, as well as for points where density is low. Therefore, there is no direct correspondence between the magnitude of hubness and point density. Naturally, since hubs tend to be close to many points, it would be expected that density estimates at hub points are not low, but they do not necessarily correspond to the points of highest density among the data. Also, in order to calculate the exact volume of the neighborhood around a given point, one needs to have a suitable data representation. For hubness, one only needs the distance matrix.

Computational complexity of hubness-based algorithms is mostly determined by the cost of computing hubness scores. Computing the entire distance matrix may not be feasible for some very large datasets. However, it was demonstrated in [18] that it is possible to construct a k-NN graph (from which hubness scores can be read) in $\Theta(ndt)$, where the user-defined value $t > 1$ expresses the desired quality of graph construction. It was shown that good quality may be achieved with small values of t.

4.1 Deterministic Approach

A simple way to employ hubs for clustering is to use them as one would normally use centroids. Also, it allows us to make a direct comparison with the K-means algorithm. The algorithm, referred to as *K-hubs*, is given in Algorithm 1.

Algorithm 1 K-hubs

```
initializeClusterCenters();
Cluster[] clusters = formClusters();
repeat
    for all Cluster c ∈ clusters do
        DataPoint h = findClusterHub(c);
        setClusterCenter(c, h);
    end for
    clusters = formClusters();
until noReassignments
return clusters
```

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 described in the following section. However, even though K-hubs exhibited low stability, it converges to the stable configurations very quickly, in no more than four iterations on all the data sets used for testing, most of which contained around 10000 data instances.

4.2 Probabilistic Approach

Even though points with highest hubness 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 [19]. 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 *hubness-proportional clustering* (HPC).

Algorithm 2 HPC

```
initializeClusterCenters();
Cluster[] clusters = formClusters();
float t = t_0; {initialize temperature}
repeat
    float θ = getProbFromSchedule(t);
    for all Cluster c ∈ clusters do
        float choice = randomFloat(0,1);
        if choice < θ then
            DataPoint h = findClusterHub(c);
            setClusterCenter(c, h);
        else
            for all DataPoint x ∈ c do
                setChoosingProbability(x, N_k^2(x));
            end for
            normalizeProbabilities();
            DataPoint h = chooseHubProbabilistically(c);
            setClusterCenter(c, h);
        end if
    end for
    clusters = formClusters();
    t = updateTemperature(t);
until noReassignments
return clusters
```

The reason why hubness-proportional clustering is reasonable in the context of high dimensionality lies in the skewness of the distribution of k-occurrences. Namely, there exist many more data points having a low hubness score, 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. It is possible to take as the output the best solution according to some predefined criterion like minimum squared error, rather than simply taking the last produced cluster configuration. This may in some situations produce even better clustering results. We were mostly focused on finding the best stable hub

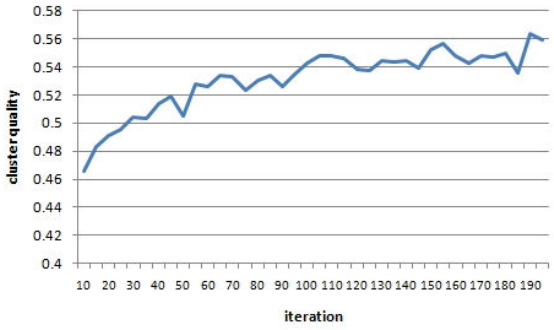


Fig. 3. Estimated quality of clustering for various durations of probabilistic search in HPC

configuration, thus we only used the last produced configuration to estimate the results for tests presented in the rest of the paper. To justify the use of the proposed stochastic scheme, we executed a series of initial tests for a synthetic mixture of Gaussians, for dimensionality $d = 50$, $n = 10000$ 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 re-initializing the seeds. The results are displayed in Fig. 3. The silhouette index [20] 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.

5 Experiments and Evaluation

We tested our approach on various high-dimensional synthetic and real-world data sets. We will use the following abbreviations in the forthcoming discussion: KM (K-Means), GKH (Global K-Hubs), LKH (Local K-Hubs), GHPC (Global Hubness-Proportional Clustering) and LHPC (Local Hubness-Proportional Clustering), *local* and *global* referring to the type of hubness score that was used (see Section 4). For all algorithms, including KM, we used the D^2 initialization procedure described in [17]. Hubness could also be used for cluster initialization, an option which we have not fully explored yet. For determining $N_k(x)$ we used $k = 10$ by default in our experiments involving synthetic data, since the generated sets were large enough to insure that such a value of k would not overly smooth out hubness. There is no known way of selecting the best k for finding neighbor-sets, with the problem also depending on the particular application. To check how the choice of k reflects on hubness-based clustering, we ran a series of tests on a fixed synthetic data set for a range of k values. The results suggest that the algorithms proposed in this paper are not very sensitive to changes of k, with no observable monotonicity in scores, meaning that the clustering quality does not rise with rising k, or vice versa. (We omit these charts due to space considerations.)

In the following sections, as a baseline we will use K-means++, since it is suitable for determining the feasibility of using hubness to estimate local centrality of points.

Table 1. Averaged results of algorithm runs on high-dimensional mixtures of Gaussians. Standard deviations taken over dataset averages are given on the side

		LKH	GKH	LHPC	GHPC	KM
$K = 5$	Silhouette	0.47 ± 0.03	0.52 ± 0.02	$\mathbf{0.62} \pm 0.02$	$\mathbf{0.63} \pm 0.02$	0.58 ± 0.02
	Entropy	0.31 ± 0.04	0.16 ± 0.01	0.078 ± 0.02	$\mathbf{0.05} \pm 0.01$	0.10 ± 0.01
	Perf.	0.35 ± 0.05	0.41 ± 0.06	$\mathbf{0.78} \pm 0.08$	$\mathbf{0.78} \pm 0.06$	0.57 ± 0.05
$K = 10$	Silhouette	0.39 ± 0.03	0.49 ± 0.01	0.53 ± 0.02	$\mathbf{0.59} \pm 0.01$	0.54 ± 0.01
	Entropy	0.51 ± 0.06	0.21 ± 0.01	0.21 ± 0.03	$\mathbf{0.07} \pm 0.01$	0.12 ± 0.01
	Perf.	0.07 ± 0.03	0.07 ± 0.03	0.32 ± 0.06	$\mathbf{0.42} \pm 0.07$	0.14 ± 0.02

5.1 Synthetic Data: Gaussian Mixtures

For comparing the resulting clustering quality, we used mainly the silhouette index as an unsupervised measure of configuration validity, and average cluster entropy as a supervised measure of clustering homogeneity. Since most of the generated data sets are "solvable," i.e., consist of non-overlapping Gaussian distributions, we also report the normalized frequency with which the algorithms were able to find these perfect configurations. We ran two lines of experiments, one using 5 Gaussian generators, the other using 10. For each of these, we generated data of ten different high dimensionalities, more specifically for $10, 20, 30, \ldots, 100$. In each case, 10 different Gaussian mixtures were randomly generated, resulting in 200 different generic sets, 100 of them containing 5 data clusters, the other containing 10. On each of the data sets, KM and all of the hub-based algorithms have been executed 30 times and the averages were calculated.

Table 1 shows the final summary of all these runs. (Henceforth, we use boldface to denote measurements that are significantly better than others, in the sense of having no overlap of surrounding one-standard deviation intervals.) Global hubness is definitely to be preferred, especially in the presence of more clusters, which further restricts neighbor sets in the case of local hubness scores. Probabilistic approaches significantly outperform the deterministic ones, even though GKH and LKH also sometimes converge to the best configurations, but much less frequently. More importantly, the best overall algorithm in these tests was GHPC, which outperformed KM on all basis, having lower average entropy, a higher silhouette index, and a much higher frequency of finding the perfect configuration. This suggests that GHPC is a good option for clustering high-dimensional Gaussian mixtures. Regarding the number of dimensions when the actual improvements begin to show, in our lower-dimensional test runs, GHPC was better already on 6-dimensional mixtures. Since we concluded that using global hubness leads to better results, we only consider GKH and GHPC in the rest of the experiments.

5.2 Clustering in the Presence of High Noise Levels

Real-world data often contains noisy or erroneous values due to the nature of the data-collecting process. It is natural to assume that hub-based algorithms will be more robust with respect to noise, since the hubness-proportional search is driven mostly by the highest-hubness elements, not the outliers. In the case of KM, all of the instances within the current cluster directly determine the location of the centroid in the next iteration.

Table 2. Estimated cluster quality at various noise levels

	GKH		GHPC		KM	
	Silhouette	Entropy	Silhouette	Entropy	Silhouette	Entropy
Avg. total	0.78	0.35	**0.83**	**0.28**	0.70	0.62
Avg. noise 10–50%	0.77	0.37	**0.82**	**0.29**	0.68	0.66
Avg. noise 30–50%	0.73	0.42	**0.80**	**0.32**	0.66	0.71

When the noise level is low, some sort of outlier removal technique may be applied. In setups involving high levels of noise this is not the case. We generated a data set of 10000 instances as a mixture of 5 clearly separated Gaussians, farther away from each other than in the previously described experiments. To this data we incrementally added noise, 250 instances at a time, drawn from a uniform distribution on a hypercube containing all the data points. In other words, clusters were immersed in uniform noise. The highest level of noise for which we tested was the case when there was an equal number of actual data instances in original clusters and noisy instances. At each noise level, KM, GKH and GHPC were run 50 times each. To reduce the influence of noise on hubness estimates, $k = 20$ was used. The silhouette index and average entropy were computed only on the non-noisy restriction of the data, i.e., the original Gaussian clusters. A brief summary of total averages is given in Table 2. The hub-based algorithms show substantial improvements in higher noise levels, which is a useful property. The difference in entropy was quite convincing, 0.62 for KM and only 0.28 for GHPC on average over all the runs. Even though KM had a smaller square error calculated on the combined noisy data set, hub-based approaches were better at finding the underlying structure of the original data.

5.3 Experiments on Real-World Data

The two-part Miss-America data set (cs.joensuu.fi/sipu/datasets/) was used for evaluation. Each part consists of 6480 instances having 16 dimensions. Results were compared for various predefined numbers of clusters in algorithm calls. Each algorithm was tested 50 times for each number of clusters. Neighborhood size was set to 5. The silhouette index was again used to measure quality. For all the experiments on real-world data we used only the silhouette index because categories in real world data sets often violate the cluster assumption, so any conclusions based on label entropy would be less reliable. The results for both parts of the data set are given in Table 3. GHPC clearly outperformed both other algorithms, showing highest improvements for smaller numbers of clusters. Observe that for $K = 2$ it achieved a double of KM's silhouette index, 0.42 compared to 0.21 on Part I and 0.36 compared to 0.18 on Part II.

Tests were also run on several UCI datasets (archive.ics.uci.edu/ml/datasets.html). Values of all the individual features in the data sets were normalized prior to testing. The results, shown in Table 4, are mostly comparable between the algorithms. Value of k was set to 20. The datasets were simple, composed only of few clusters, so the results being similar is not surprising. Note, however, that GHPC did as well as KM on *Iris* dataset, which is only 4-dimensional. This suggests that hubness-based algorithms might also be successfully applied in some lower-dimensional cases.

Table 3. Cluster configuration quality measured by the silhouette index, on the Miss-America data set, parts I and II, for various cluster numbers

	K	2	4	6	8	10	12	14	16
Part I	GKH	0.40	0.20	0.10	0.09	0.06	0.06	0.06	0.06
	GHPC	**0.42**	**0.31**	**0.31**	**0.21**	**0.21**	**0.15**	**0.14**	**0.12**
	KM	0.21	0.13	0.12	0.08	0.08	0.08	0.07	0.07
Part II	GKH	0.31	0.13	0.07	0.06	0.05	0.05	0.05	0.05
	GHPC	**0.36**	**0.23**	0.12	0.09	0.09	0.08	0.09	0.07
	KM	0.18	0.12	0.10	0.08	0.08	0.08	0.08	0.07

Table 4. Cluster configuration quality measured by the silhouette index, on some UCI datasets

dataset	size	d	K	GKH-Sil.	GHPC-Sil.	KM-Sil.
wdbc	569	30	2	0.43	0.43	0.43
spambase	4601	57	2	0.28	**0.44**	0.39
arcene	100	1000	2	0.35	0.36	0.34
ovarian	253	15154	2	0.22	0.22	0.21
iris	158	4	3	0.56	0.58	0.58

6 Conclusions and Future Work

Using hubness for data clustering has not previously been attempted. We have shown that using hubs to approximate local data centers is not only a feasible option, but also frequently leads to improvement over the centroid-based approach. In our experiments GHPC (Global Hubness-Proportional Clustering) had an overall best performance in various test settings, on both synthetic and real-world data, as well as in the presence of high levels of artificially introduced noise. Global hubness estimates are generally to be preferred to the local ones if used in the proposed framework. Hub-based algorithms are designed specifically for high-dimensional data. This is an unusual property, since the performance of most standard clustering algorithms deteriorates with an increase of dimensionality. Hubness, on the other hand, is an inherent property of high-dimensional data, and this is precisely where GHPC may offer greatest improvement.

The proposed algorithms represent only one possible approach to using hubness for improving high-dimensional data clustering. Next, we will explore related agglomerative approaches. However, even the described algorithms offer space for improvements, since some questions are left unanswered: What is the best choice of k? Is it possible to automatically determine the appropriate number of clusters by carefully inspecting the hubs? In cases such as one depicted in Fig. 2 that would probably be possible. What is the best annealing schedule in GHPC? Is it possible to use several different values of k in LHPC to avoid over-smoothing the hubness estimates for small clusters over iterations and make local hubness more useful? Is there a better way to initialize hubness-based algorithms? We plan to address all these details in our future work.

Acknowledgments. This work was supported by the Slovenian Research Agency program Knowledge Technologies P2-0103, and the Serbian Ministry of Science and Technological Development project no. OI174023.

References

1. Han, J., Kamber, M.: Data Mining: Concepts and Techniques. 2nd edn. Morgan Kaufmann Publishers (2006)
2. Aggarwal, C.C., Yu, P.S.: Finding generalized projected clusters in high dimensional spaces. In: Proc. 26th ACM SIGMOD Int. Conf. on Management of Data. (2000) 70–81
3. Kailing, K., Kriegel, H.P., Kröger, P., Wanka, S.: Ranking interesting subspaces for clustering high dimensional data. In: Proc. 7th European Conf. on Principles and Practice of Knowledge Discovery in Databases (PKDD). (2003) 241–252
4. Kailing, K., Kriegel, H.P., Kröger, P.: Density-connected subspace clustering for high-dimensional data. In: Proc. 4th SIAM Int. Conf. on Data Mining (SDM). (2004) 246–257
5. Kriegel, H.P., Kröger, P., Renz, M., Wurst, S.: A generic framework for efficient subspace clustering of high-dimensional data. In: Proc. 5th IEEE Int. Conf. on Data Mining (ICDM). (2005) 250–257
6. Aggarwal, C.C., Hinneburg, A., Keim, D.A.: On the surprising behavior of distance metrics in high dimensional spaces. In: Proc. 8th Int. Conf. on Database Theory (ICDT). (2001) 420–434
7. François, D., Wertz, V., Verleysen, M.: The concentration of fractional distances. IEEE Transactions on Knowledge and Data Engineering **19**(7) (2007) 873–886
8. Durrant, R.J., Kabán, A.: When is 'nearest neighbour' meaningful: A converse theorem and implications. Journal of Complexity **25**(4) (2009) 385–397
9. Agirre, E., Martínez, D., de Lacalle, O.L., Soroa, A.: Two graph-based algorithms for state-of-the-art WSD. In: Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP). (2006) 585–593
10. Tran, T.N., Wehrens, R., Buydens, L.M.C.: Knn density-based clustering for high dimensional multispectral images. In: Proc. 2nd GRSS/ISPRS Joint Workshop on Remote Sensing and Data Fusion over Urban Areas Workshop. (2003) 147–151
11. Biçici, E., Yuret, D.: Locally scaled density based clustering. In: Proc. 8th Int. Conf. on Adaptive and Natural Computing Algorithms (ICANNGA), Part I. (2007) 739–748
12. Zhang, C., Zhang, X., Zhang, M.Q., Li, Y.: Neighbor number, valley seeking and clustering. Pattern Recognition Letters **28**(2) (2007) 173–180
13. Hader, S., Hamprecht, F.A.: Efficient density clustering using basin spanning trees. In: Proc. 26th Annual Conf. of the Gesellschaft für Klassifikation. (2003) 39–48
14. Ding, C., He, X.: K-nearest-neighbor consistency in data clustering: Incorporating local information into global optimization. In: Proc. ACM Symposium on Applied Computing (SAC). (2004) 584–589
15. Chang, C.T., Lai, J.Z.C., Jeng, M.D.: Fast agglomerative clustering using information of k-nearest neighbors. Pattern Recognition **43**(12) (2010) 3958–3968
16. Radovanović, M., Nanopoulos, A., Ivanović, M.: Hubs in space: Popular nearest neighbors in high-dimensional data. Journal of Machine Learning Research **11** (2010) 2487–2531
17. Arthur, D., Vassilvitskii, S.: k-means++: The advantages of careful seeding. In: Proc. 18th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA). (2007) 1027–1035
18. Chen, J., Fang, H., Saad, Y.: Fast approximate kNN graph construction for high dimensional data via recursive Lanczos bisection. Journal of Machine Learning Research **10** (2009) 1989–2012
19. Corne, D., Dorigo, M., Glover, F.: New Ideas in Optimization. McGraw-Hill (1999)
20. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison Wesley (2005)