Distance based fast hierarchical clustering method for large datasets

Bidyut Kr. Patra, Neminath Hubballi, Santosh Biswas and Sukumar Nandi

Department of Computer Science & Engineering,
Indian Institute of Technology Guwahati, Assam 781039, INDIA
{bidyut, neminath, santosh_biswas, sukumar} @iitg.ernet.in

Abstract. Average-link (AL) is a distance based hierarchical clustering method, which is not sensitive to the noisy patterns. However, like all hierarchical clustering methods AL also needs to scan the dataset many times. AL has time and space complexity of $O(n^2)$, where n is the size of the dataset. These prohibit the use of AL for large datasets. In this paper, we have proposed a distance based hierarchical clustering method termed *l*-AL which speeds up the classical AL method in any metric (vector or non-vector) space. In this scheme, first leaders clustering method is applied to the dataset to derive a set of leaders and subsequently AL clustering is applied to the leaders. To speed-up the leaders clustering method, reduction in distance computations is also proposed in this paper. Experimental results confirm that the *l*-AL method is considerably faster than the classical AL method yet keeping clustering results at par with the classical AL method.

Keyword: distance based clustering, leaders clustering, average-link, large datasets.

1 Introduction

Clustering technique is required in numerous fields of engineering namely Data Mining, Pattern Recognition, Statistical Data Analysis, Bio-informatics, etc. [1–3]. Given a set of data points (called patters), clustering involves grouping these patterns into different disjoint subsets termed as clusters based on some similarity measures. In other words, patterns in a cluster are more similar to each other than patterns in other clusters.

The clustering methods are mainly divided into two categories *viz.*, partitional clustering and hierarchical clustering, based on the way they produce the results. Partitional clustering methods create a single clustering (flat clustering) of the dataset. Partitional clustering can be classified into two classes based on the criteria used *viz.*, distance based and density based. Distance based methods optimize a global criteria based on the distance between the patterns. k-means, CLARA, CLARANS are examples of distance based clustering method. Density based methods optimize local criteria based on density information of the patterns. DBSCAN [4], DenClue are some well known density based clustering methods.

Hierarchical clustering methods create a sequence of nested clusterings of the dataset. Like partitional clustering, hierarchical clustering methods can also be classified in two classes *viz.*, density based (e.g., OPTICS [5], Chameleon) and distance based (e.g., single-link (SL) [6], complete-link (CL) [7], average-link (AL) [8]).

The above three distance based hierarchical clustering methods namely SL, CL and AL differ in the "distance measure" between a pair of clusters. In SL (CL), distance between a pair of clusters C_1 and C_2 (say), is the distance between two closest (farthest) patterns one from C_1 and the other from C_2. In other words, only a pair of patterns decides the distance and it is independent of number of patterns present in the clusters. Therefore, SL and CL clustering methods are sensitive to outliers or noisy patterns. To minimize the effect of noisy patterns, inter-cluster distance in AL technique is computed using all patterns present in both clusters. Distance between C_1 and C_2 in AL method is the average of distances between all pairs of patterns, one taken from C_1 and the other from C_2. It builds a dendogram where each level represents a clustering of the dataset. A suitable clustering is chosen from the dendogram based on the requirement. Selection of a clustering from the dendogram can be done by specifying (i) the minimum distance between any pair of clusters (ii) number of desired clusters.

For some applications like network intrusion detection system (NIDS) proportion of the data points is unequal (i.e., number of the data points of abnormal/attack type is very less compared to normal data points). These low proportional data points (abnormal/attack data points) look like outliers in the feature space. These abnormal data points are likely to get merged with the clusters of normal data points in SL and CL methods as they are sensitive to the noisy (outlier) points. However, AL method works well even in the presence of noisy (outlier) data points. So, AL clustering method is more suitable for these type of applications.

AL needs to scan the dataset many times and has time and space complexity of $O(n^2)$. These prohibit the use of AL for large datasets. In this paper, we have proposed a distance based hierarchical clustering method termed *l*-AL which speeds up the classical AL method. *l*-AL method is suitable for any metric space.

In the proposed scheme, we have used leaders clustering method to derive a set of leaders of the dataset. Later, AL is used to cluster the leaders. The final clusterings are obtained by just replacing the leaders by their corresponding followers. *l*-AL has lower time complexity because AL is used only on the leaders which are much smaller in number compared to the dataset. It may be noted that leader generation involves single scan of the dataset. Expansion of the clusters involves a constant complexity. Further, technique has been proposed to reduce the number of distance computations in leaders clustering method.

The contributions of this paper are:

- Technique has been proposed to reduce the number of distance calculations required in leaders clustering. Triangle inequality property of metric space has been used for this reduction.
- A distance based hierarchical clustering method termed *l*-AL is proposed which speeds up the classical AL method and scans the dataset once. *l*-AL uses the accelerated leader clustering technique to generate the leaders.
- *l*-AL does not use any vector space properties [1]. It utilizes only the distance information between the data points. Therefore, *l*-AL method is suitable for vector as well as non-vector metric space.

[1] vector addition and scalar multiplication

– Experimental results illustrate *l*-AL method to be faster than classical AL method yet maintaining clustering results at par with classical AL for various cut-off distances.

The rest of the paper is organized as follows. Section 2 describes a summary of related works. Section 3 describes the brief background of the proposed clustering method. Section 4 describes the proposed leader-average-link (*l*-AL) method and also a relationship between the AL method and the *l*-AL method is formally reported. Experimental results and conclusion are discussed in Section 5 and Section 6, respectively.

2 Related work

In this section, a brief review of related works is reported for distance based hierarchical clustering methods.

T. Zhang et al. in [9] introduced a clustering method called BIRCH for large datasets. The core concept of BIRCH is *Clustering Feature (CF)*. The *CF* utilizes the vector space (Euclidean space) properties to store the summary of k data points $\{\overrightarrow{X_i}\}_{i=1..k}$. The *CF* is defined as $CF = (k, \sum_{i=1}^{k} \overrightarrow{X_i}, \sum_{i=1}^{k} \overrightarrow{X_i}^2)$. One can easily compute average intra-cluster and inter-cluster distances from the CF values. However, in many applications, datasets [2] are from non-vector metric space. Therefore, BIRCH method cannot handle those applications.

Dash et al. in [10] proposed a fast hierarchical clustering method based on the partially overlapping partitioning (POP). First, dataset is partitioned into a number of overlapping cells and cells are progressively merged into clusters until the distance between any two closest cells is more than a pre-specified distance. Next, traditional hierarchical agglomerative clustering (centroid based) method is applied for obtaining the final clusterings. It uses the vector space properties to calculate centroid of a cluster.

Nanni et al. in [11] exploited the triangle inequality property of the distance metric to speed-up the hierarchical clustering methods (SL and CL).

Recently, Koga et al. [12] proposed a fast approximation algorithm for SL method. Unlike classical SL method it quickly finds close clusters in linear time using a probabilistic approach (*LSH* [13].) Koga et al. showed that their method runs in linear time under certain assumptions.

These methods successfully speedup the traditional clustering methods. However, these methods are not suitable either for large datasets (entire dataset in main memory of machine) or categorical datasets (non-vector space). The proposed *l*-AL method speeds up the exiting AL clustering method but needs to store only the leaders in the main memory of the machine (as AL is applied only on the leaders) and uses only the distance information. So, for large datasets *l*-AL method is more suitable instead of classical AL.

3 Background of the proposed method

As already discussed, the proposed *l*-AL builds on two clustering methods *viz.*, leaders clustering and average-link clustering method; they are discussed in this section. These

[2] datasets with categorical features

two clustering methods have their own advantages and disadvantages. The proposed clustering methods exploit the advantages of these two clustering methods.

3.1 The leaders clustering method

The leaders clustering method [2] is a distance based partitional clustering method. It is a single scan and an incremental clustering method. Recently, leaders clustering method has been started using in preclustering phase of many data mining applications [14, 15]. For a given threshold distance τ, it produces a set of leaders $\mathcal{L}$ incrementally as follows. For each pattern x, if there is a leader $l \in \mathcal{L}$ such that $||x - l|| <= \tau$, then x is assigned to the cluster represented by l. In this case, we call x as a *follower* of the leader l. If there is no such leader, then x becomes a new leader. The time complexity of the leaders clustering is $O(mn)$, where $m = |\mathcal{L}|$. The space complexity is $O(m)$, if only leaders are stored; otherwise $O(n)$. However, Viswanth et al. in [14] shown that under certain assumptions about the datasets, there exists an upper-bound for m, which is independent of n . Therefore, time complexity of the leaders clustering method becomes linear. However, it can only find convex shaped clusters.

3.2 The average-link clustering method

The average-link [8, 16] is a distance based agglomerative hierarchical clustering method. In average-link, distance between two clusters C_1 and C_2 is the average of distances between all pairs in $C_1 \times C_2$. That is,

$$Distance(C_i, C_j) = \frac{1}{|C_i| * |C_j|} \sum_{x_i \in C_i} \sum_{x_j \in C_j} ||x_i - x_j||$$

The average-link method with inter-cluster distance (h) is depicted in Algorithm 1. The AL method is not sensitive to noisy patterns. The time and space complexity of the AL method are $O(n^2)$ [8, 16]. It scans the dataset many times. Therefore, AL method is not suitable for the large dataset.

4 The proposed clustering method

To overcome the deficiencies of the AL method, we propose a clustering method termed as *l*-AL, which is the combination of leaders and average-link. Further, technique to

Algorithm 1 AL($\mathcal{D}$,h)

Place each pattern $x \in \mathcal{D}$ in a separate cluster. This is the initial clustering $\pi_1 = \{C_1, C_2, \ldots, C_n\}$ of $\mathcal{D}$. Compute the inter-cluster distance matrix and set $i = 1$.
while There is a pair of clusters $C_x, C_y \in \pi_i$ such that *Distance*$(C_x, C_y) \leq h$ **do**
  Select two closest clusters C_l and C_m and merge into a single new cluster $C = C_l \cup C_m$.
  Next clustering is $\pi_{i+1} = \pi_i \cup \{C\} \setminus \{C_l, C_m\}$; $i = i + 1$
  Update the distances from C to all other clusters in the current clustering π_i.
end while
Output all clustering $\pi_1, \pi_2, \ldots, \pi_p$.

reduce the number of distance computations while applying the leaders clustering to the datasets is also proposed. In this section, we first discuss the technique to speed up the leaders clustering followed by the proposed *l*-AL scheme.

4.1 Accelerating leader clustering method

We use triangle inequality property to reduce the number of distance computations of the leaders clustering method. We term this approach as *Accelerated leader*. In recent years, triangle inequality property of the metric space has been used to reduce the distance computations in the clustering methods [11, 17, 18]. The triangle inequality property can be stated as follows.

$$\forall a, b, c \in \mathcal{D}, d(a, b) \leq d(b, c) + d(a, c) \tag{1}$$

where $\mathcal{D}$ is the set of data points, d is a distance function over the metric space $M = (\mathcal{D}, d)$.

Let l_1, l_2 be the two leaders and x be an arbitrary pattern of the dataset. Form equation (1),

$$d(x, l_2) \geq |d(l_1, l_2) - d(x, l_1)| \tag{2}$$

From equation(1) it may be noted that a lower bound on the distance between leader l_2 and pattern x (termed as $d^{lower}(x, l_2)$) can be obtained from $d(l_1, x)$ and $d(l_1, l_2)$ without calculating the exact distance between l_2 and x.

The *Accelerated leader* works as follows. The scheme requires a distance matrix for leaders. This distance matrix can be generated hand-in-hand during the generation of leaders (without any extra distance computation). Therefore, one can easily estimate $d^{lower}(x, l_2)$ only by computing distance $d(l_1, x)$.

Let τ be the leader's threshold. Let $\mathcal{L} = \{l_1, l_2, \ldots, l_k\}$ be the set of leaders generated at an instant and all be marked as "unprocessed" leaders. The scheme starts with calculating the distance between a new pattern x and leader l_f (where l_f is the first generated leader among the set of "unprocessed" leaders). If $d(x, l_f) \leq \tau$, then x becomes the follower of leader l_f. If $d(x, l_f) > \tau$, we can avoid the distance computations from all leaders $l_i \in L - \{l_f\}$ where estimated lower bound $d^{lower}(x, l_i) > \tau$. Leaders l_i, l_f are marked as "processed" (pruned) leaders. If all leaders are pruned then x becomes a new leader and added to $\mathcal{L}$. If all leaders are not marked as "processed", we repeat same procedure of calculating distance between x with next unprocessed leader $l_u \in \mathcal{L}$ if $d(x, l_u) < d(x, l_f)$. If no (unprocessed) $l_u \in \mathcal{L}$ is found such that $d(x, l_u) > d(x, l_f)$, then there cannot be a leader l_j such that $d(x, l_j) \leq \tau$; so x becomes a new leader and added to $\mathcal{L}$. The whole procedure of *Accelerated leaders* is depicted in Algorithm 2.

4.2 The leader-average-link(*l*-AL) method

In this sub-section, we discuss the proposed *l*-AL scheme. The *l*-AL method works as follows. First, a set of leaders ($\mathcal{L}$) is obtained applying the Accelerated leaders clustering method to the dataset (as discussed in previous subsection). Next, these leaders

Algorithm 2 Accelerated leader($\mathcal{D}$, τ)

$\mathcal{L} \leftarrow \{l_1\}$; { Let $l_1 \in \mathcal{D}$ be the first scanned pattern}
for each $x \in \mathcal{D} \setminus l_1$ **do**
  $S \leftarrow \mathcal{L}$; $MIN = \infty$;
  while (x does not become a follower and S is not empty) **do**
    Pick a leader l_i and delete from S. { l_i is first generated leader in S.}
    if $d(x, l_i) \leq \tau$ **then**
      x becomes a follower of l_i; break;
    else if $d(x, l_i) < MIN$ **then**
      $MIN = d(x, l_i)$;
      for each leader $l_k \in S(l_k \neq l_i)$ **do**
        if $d^{lower}(x, l_k) > \tau$ **then**
          delete l_k from set S.
        end if
      end for
    end if
  end while
  if (x not be follower of any exsisting leaders in $\mathcal{L}$) **then**
    x becomes new leader and added to $\mathcal{L}$.
  end if
end for
Output $\mathcal{L}^\star = \{(l, \textit{followers}(l)) \mid l \in \mathcal{L}\}$.

are clustered using classical AL method with minimum inter-cluster distance h. Finally, each leader is replaced by its followers set to produce the final sequence of clusterings. The *l*-AL method is depicted in Algorithm 3.

The time and space complexity of the proposed method are analyzed as follows.

1. The step of obtaining set of all leaders $\mathcal{L}$ takes time of $O(mn)$, where m is the size of the leader set. The space complexity is $O(m)$. It scans the dataset once.
2. The time complexity of the AL($\mathcal{L}$, h) is $O(m^2)$. The space complexity is $O(m^2)$.

The overall running time of *l*-AL is $O(mn + m^2) = O(mn)$. Experimentally, we also show that *l*-AL is considerably faster than that of the classical AL method, since AL works with the whole dataset, whereas the *l*-AL works with set of leaders . The space complexity of the *l*-AL method is $O(m^2)$.

Algorithm 3 *l*-AL($\mathcal{D}$, τ, h)

Apply Accelerated leader($\mathcal{D}$, τ) as given in Algorithm 2. Let the set of leaders be $\mathcal{L}$.
Apply AL($\mathcal{L}, h$) as given in Algorithm 1. Let output be $\pi_1^{\mathcal{L}}, \pi_2^{\mathcal{L}}, \ldots, \pi_k^{\mathcal{L}}$ { A sequence of clusterings of leaderset}
Each leader in clustering $\pi_i^{\mathcal{L}}$ is replaced by its followers set. This gives a sequence of clustering of the dataset (say $\pi_1^{\mathcal{D}}, \pi_2^{\mathcal{D}}, \ldots, \pi_k^{\mathcal{D}}$).
Output $\pi_1^{\mathcal{D}}, \pi_2^{\mathcal{D}}, \ldots, \pi_k^{\mathcal{D}}$.

4.3 Relationship between AL and *l*-AL methods

As discussed in previous sub-section *l*-AL clusters dataset at a computational cost significantly lower than classical AL. It may be noted that *l*-AL may overestimate or underestimate the distance between a pair of clusters with compared to the classical AL method (termed as distance error). This may lead to deviation in clustering results obtained by *l*-AL compared to AL. In this subsection a theoretical upper bound of the distance error is established.

Let $l_1, l_2 \in \mathcal{L}$ be two leaders obtained using the threshold τ. Let $F(l_1) \subseteq \mathcal{D}$ be the set of followers of leader l_1 including l_1. Similarly, $F(l_2)$ is the set of followers of l_2.

Lemma 1. *If the leaders threshold is τ, then l-AL may introduce an error of average value $Er(l_1, l_2) < 2\tau$, while measuring the distance between a pair of leaders (l_1, l_2).*

Proof: Let $||l_1 - l_2|| = T > 2\tau$. We have three cases.

1. We assume that all followers of l_1 are more than T distance away from the followers of l_2, except the leaders themselves. (This case is illustrated in Fig. $1(a)$). Formally, $||x_i - x_j|| > T$ where $x_i \in F(l_1) \setminus \{l_1\}$ and $x_j \in F(l_2) \setminus \{l_2\}$. Therefore, distance between a pair of followers (x_i, x_j) can be at most $T + 2\tau$. So, for all followers (of this case) *l*-AL underestimates the distance and approximates to T (as $||l_1 - l_2|| = T$). Therefore, error incurred by a pair of such followers is at most 2τ. The average error $Er(l_1, l_2)$ introduced by the *l*-AL method for a pair of leader can be computed as follows.

$$\begin{aligned} Er(l_1, l_2) &= \frac{(m_1 - 1)(m_2 - 1)2\tau + (m_1 - 1)\tau + (m_2 - 1)\tau}{m_1 m_2} \\ &< \frac{m_1 m_2 * 2\tau}{m_1 m_2} = 2\tau \end{aligned} \tag{3}$$

   where $m_1 = |F(l_1)|$ and $m_2 = |F(l_2)|$.
   The first term of the numerator of equation (3) appears due to errors introduced by the followers of l_1 ($m_1 - 1$ in number) and l_2 ($m_2 - 1$ in number). Second (third) term captures errors introduced by $l_2(l_1)$ and followers of $l_1(l_2)$.
2. We assume that $||x_i - x_j|| < T$ such that $x_i \in F(l_1) \setminus \{l_1\}$ and $x_j \in F(l_2) \setminus \{l_2\}$, distance between x_i, x_j cannot be less than $T - 2\tau$. Similar to case 1 we obtain the average error $Er(l_1, l_2) < 2\tau$. (Fig. $1(b)$). Here, *l*-AL overestimates the distance.

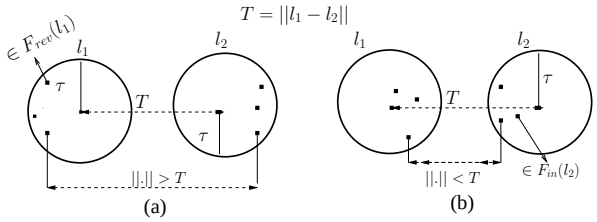

Fig. 1. (a) *l*-AL underestimates the distance (b) *l*-AL overestimates the distance

3. If distance between any pair of followers is $||x_i - x_j|| = (T - 2\tau, T + 2\tau)$, the average error is less than 2τ.

From all three cases, we obtain that average error $Er(l_1, l_2)$ is less than 2τ □

The distance error computation between two leaders can easily be extended for a pair of clusters, as follows.

Theorem 1. *If the leaders threshold is τ, then l-AL may introduce an average error $Er(C_1, C_2) < 2\tau$ in measuring the distance between a pair of clusters (C_1, C_2) .*

Proof: From Lemma 1, we know that average error between a pair of leaders $Er(l_1, l_2) < 2\tau$. Let the upper bound on the average error $Er(l_1, l_2)$ be $2\tau - \epsilon$, where $0 < \epsilon << \tau$. Then the average error between a pair of clusters (C_1, C_2) is as follows.

$$Er(C_1, C_2) = \frac{(2\tau - \epsilon) * m_1^l m_2^l}{m_1^l m_2^l} = 2\tau - \epsilon < 2\tau,$$

where m_1^l and m_2^l are the numbers of leaders of the clusters C_1 and C_2, respectively. □

For large datasets, numbers of leaders are considerably less compared to the size of the data. The numbers of followers per leader are considerably large. As a result, there is high probability that followers of leader are distributed evenly. This leads to error in distance computation between leaders by *l*-AL method is marginal, which is also reflected in our experimental results. So, Corollary 1 can be deduced.

Corollary 1 *If the followers of leaders are distributed uniformly, the average distance error for those leaders is* 0. □

Table 1. Datasets Used

Dataset	# Pattern	# Features
Circle (Synthetic)	28000	2
Gaussian(Synthetic)	4078	2
Pendigits	7494	16
Letter	20000	16
Shuttle	58000	9

Table 2. Performance of Accelerated leaders for Circle dataset

Threshold (τ)	Method	# Computations (in Million)
0.1	Leaders	90.13
	Accelerated leader	**28.66**
0.2	Leaders	20.03
	Accelerated leader	**2.37**
0.3	Leaders	11.33
	Accelerated leader	**0.96**
0.4	Leaders	5.97
	Accelerated leader	**0.49**
0.5	Leaders	3.85
	Accelerated leader	**0.35**
0.6	Leaders	2.81
	Accelerated leader	**0.30**

5 Experimental Results

In this section, we discuss the experimental evaluation of our proposed clustering method. We evaluated the *l*-AL method and *Accelerated leader* separately. We conducted the experiments with synthetic and real world datasets (Table 1) (*http://archive.ics.uci.edu/ml*) after removing the class labels.

We implemented leaders clustering and *Accelerated leader* using C language and executed on Intel Core 2 Duo with $2GB$ RAM IBM PC. These two methods are tested with Circle and Shuttle datasets. The detailed results are shown in Table 2 and Fig 2. For the Circle dataset, with leader's threshold $\tau = 0.1$, our proposed *Accelerated leader* computes 60 millions less distance calculations to achieve same results as that of the classical leaders method (Table 2). For the other values of τ, proposed *Accelerated leader* performs significantly less computations compared to that of classical leaders method (Table 2).

To show the performance of the proposed leaders clustering speeding-up technique with variable dataset size, experiments are conducted on Shuttle dataset with leaders threshold $\tau = 0.001$. This is reported in Fig 2. It may be noted that with the increase of the data size, number of distance calculations in *Accelerated leader* reduces significantly compared to classical leaders.

Performance of *l-AL* method To show the performance of the *l*-AL method, we implemented AL and *l*-AL methods using C language and executed on Intel Xeon Processor ($3.6GHz$) with 8GB RAM IBM Workstation. We computed the Rand Index (RI)([19]) between the final clustering results of the *l*-AL and the AL method. We conducted experiments with synthetic (Gaussian) (Fig. 3) as well as real world large datasets. The detailed results are provided in Table 3, Table 4 and Table 5.

The Gaussian is a 2 dimensional data with four clusters. Three clusters are drawn from the normal distribution with means $((0\ 0)^T, (0\ 8)^T, (7\ 7)^T)$ and covarience matrix $I_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. Each of these three clusters has 1000 patterns. Fourth cluster is drawn

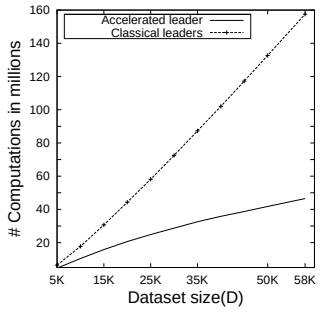

Fig. 2. Number of distance computations Vs dataset size for Shuttle data ($\tau = 0.001$)

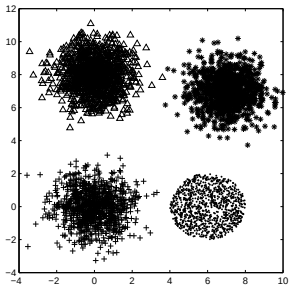

Fig. 3. Gaussian (Synthetic) Dataset.

Table 3. Results for Gaussian Dataset

Threshold (τ)	Cut-off (h)	Method	Time (Sec.)	Rand Index (RI)
0.25	4.0	***l*-AL**	**0.04**	**0.999**
	4.0	AL	14.96	–
	4.5	***l*-AL**	**0.04**	**0.999**
	4.5	AL	14.96	—-
	5.0	***l*-AL**	**0.04**	**1.000**
	5.0	AL	14.96	—
0.50	6.5	***l*-AL**	**0.01**	**0.870**
	6.5	AL	14.96	—
	7.0	***l*-AL**	**0.01**	**1.000**
	7.0	AL	14.96	—
	8.0	***l*-AL**	**0.01**	**1.000**
	8.0	AL	14.96	—

from a uniform random distribution (Fig. 3). For this dataset, leaders thresholds τ were chosen as $0.25, 0.50$. The clustering results of the proposed *l*-AL method are same as the classical AL method with cut-off distances (h) $5.0, 7.0, 8.0$ and results are very close to AL method for the cut-off distances (h) $4.0, 4.5, 6.5$ (Table 3). The execution time of the proposed method is less than 0.3% of that of the AL method.

To show the effectiveness of the proposed method in the real world large datasets, we experimented with Pendigits, Letter and Shuttle datasets (Table 1). For Pendigits dataset, clustering results of *l*-AL method is very close ($RI = 0.899, 0.897, 0.911, 0.904, 0.935, 0.933, 0.913, 0.909$) to that of the classical AL method with different τ $(30, 40)$ and different h $(145, 150, 155, 160)$ (Table 4). The *l*-AL consumes less than 0.5% of CPU time of that of the AL method.

For Letter dataset, with $\tau = 4$ and different cut-off distances ($h = 10, 12, 15$) *l*-AL method produces clustering results ($RI = 0.811, 0.835, 0.977$) close to that of the

Table 4. Results for Standard Datasets

Dataset	Threshold (τ)	Cut-off (h)	Method	Time (Sec.)	Rand Index (RI)
Pendigits	30	145	***l*-AL**	**1.13**	**0.899**
		145	AL	201.55	–
		150	***l*-AL**	**1.13**	**0.897**
		150	AL	201.55	—-
		155	***l*-AL**	**1.13**	**0.911**
		155	AL	201.55	—
		160	***l*-AL**	**1.13**	**0.904**
		160	AL	201.55	—
	40	145	***l*-AL**	**0.31**	**0.935**
		145	AL	201.55	—
		150	***l*-AL**	**0.31**	**0.933**
		150	AL	201.55	—
		155	***l*-AL**	**0.31**	**0.913**
		155	AL	201.55	—
		160	***l*-AL**	**0.31**	**0.909**
		160	AL	201.55	—

Table 5. Results for Large Real Datasets

Dataset	Threshold (τ)	Cut-off (h)	Method	Time (Sec.)	Rand Index (RI)
Letter	4	10	***l*-AL**	**3.28**	**0.811**
		10	AL	1464.10	—
		12	***l*-AL**	**3.28**	**0.835**
		12	AL	1464.10	—
		15	***l*-AL**	**3.28**	**0.977**
		15	AL	1464.10	—
Shuttle	0.001	0.8	***l*-AL**	**55.55**	**0.999**
		0.8	AL	7140.54	—
		0.9	***l*-AL**	**55.55**	**0.999**
		0.9	AL	7140.54	—
		1.0	***l*-AL**	**55.55**	**0.999**
		1.0	AL	7140.54	—
		1.2	***l*-AL**	**55.55**	**1.000**
		1.2	AL	7140.54	—

classical AL method (Table 5). However, *l*-AL is more than 400 times faster than that of the classical AL method.

For Shuttle dataset, we executed AL and *l*-AL methods and results are reported in Table 5. It is noted that clustering results $(RI = 0.999, 1.000)$ are at par or same with the AL method at $\tau = 0.001$ and $h = 0.8, 0.9, 1.0, 1.2$.

6 Conclusions

In this paper, we proposed a clustering method *l*-AL for the large dataset in any metric space. In this method, we first apply leaders clustering to derive a set of prototypes of the dataset and subsequently the classical AL method is applied to the prototypes. The technique to reduce the number of distance computations in the leaders method is also proposed. The clustering results produced by the *l*-AL method are at par with that of the AL method. The *l*-AL method takes significantly less time compared to that of the AL method. Like AL, *l*-AL is immune to clustering of data with noise. As *l*-AL is faster, it can be used in application like network intrusion detection system where data size is very large and spurious patterns are very less.

References

1. Dunham, M.H.: Data Mining: Introductory and Advanced Topics. Prentice Hall, New Delhi, India (2003)
2. Hartigan, J.A.: Clustering Algorithms. John Wiley & Sons, Inc., New York, NY, USA (1975)
3. Jain, A.K., Murty, M.N., FLYNN, P.J.: Data Clustering: A Review. ACM Computing Surveys **31**(3) (1999) 264–323
4. Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: Proceedings of 2nd ACM SIGKDD. (1996) 226–231
5. Ankerst, M., Breunig, M.M., Kriegel, H.P., Sander, J.: Optics: Ordering points to identify the clustering structure. In: Proceedings ACM SIGMOD. (1999) 49–60
6. Sneath, A., P.H., Sokal: Numerical Taxonomy. Freeman, London, UK (1973)
7. King, B.: Step-Wise Clustering Procedures. Journal of the American Statistical Association **62**(317) (1967) 86–101
8. Murtagh, F.: Complexities of hierarchic clustering algorithms: state of the art. Computational Statistics Quarterly **1** (1984) 101–113
9. Zhang, T., Ramakrishnan, R., Livny, M.: BIRCH: An Efficient Data Clustering Method for Very Large Databases. In: Proceedings of the 1996 ACM SIGMOD. (1996) 103–114
10. Dash, M., Liu, H., Scheuermann, P., Tan, K.L.: Fast hierarchical clustering and its validation. Data Knowl. Eng. **44**(1) (2003) 109–138
11. Nanni, M.: Speeding-up hierarchical agglomerative clustering in presence of expensive metrics. In: Proceedings of PAKDD Conference. (2005) 378–387
12. Koga, H., Ishibashi, T., Watanabe, T.: Fast agglomerative hierarchical clustering algorithm using Locality-Sensitive Hashing. Knowledge and Information Systems **12**(1) (2007) 25–53
13. Indyk, P., Motwani, R.: Approximate nearest neighbors:towards removing the curse of dimensionality. In: Proceedings of 30th ACM Symposium on theory of Computing. (1998) 604–613

14. Viswanath, P., Babu, V.: Rough-dbscan:a fast hybrid density based clustering method for latge data sets. Pattern Recognition Letters **30**(16) (2009) 1477–1488
15. Patra, B.K., Nandi, S.: A Fast Single Link Clustering Method Based on Tolerance Rough Set Model. In: Proceedings of RSFDGrC 2009. (2009) 414–422
16. Olson, C.F.: Parallel algorithms for hierarchical clustering. Parallel Computing **21** (1995) 1313–1325
17. Elkan, C.: Using the triangle inequality to accelerate k-means. In: ICML. (2003) 147–153
18. Nassar, S., Sander, J., Cheng, C.: Incremental and effective data summarization for dynamic hierarchical clustering. In: Proceedings of SIGMOD Conference. (2004) 467–478
19. Rand, W.M.: Objective Criteria for Evaluation of Clustering Methods. Journal of American Statistical Association, **66**(336) (1971) 846–850