Speeding-up the kernel k-means clustering method: A prototype based hybrid approach

T. Hitendra Sarma\textsuperscript{1,*}, P. Viswanath\textsuperscript{1}, B. Eswara Reddy\textsuperscript{2}

\textsuperscript{1}Department of Computer Science and Engineering, Rajeev Gandhi Memorial College of Eng. and Technology-Nandyal 518501, A.P., India.
\textsuperscript{2}Department of Computer Science and Engineering JNTUA College of Engineering, Anantapur-515002, A.P., India.
Email: hitendrasarma@ieee.org, \{viswanath.pulabaigari,eswarcsejntu\}@gmail.com

Abstract

Kernel k-means clustering method has been proved to be effective in identifying non-isotropic and linearly inseparable clusters in the input space. However, this method is not a suitable one for large data-sets because of its quadratic time complexity with respect to the size of the data-set. This paper presents a simple prototype based hybrid approach to speed-up the kernel k-means method for large data-sets. The proposed method works in two stages. First, the data-set is partitioned into a number of small group-lets by using the leaders clustering method. Each group-let is represented by a prototype called its leader. The conventional leaders clustering method is modified such that these group-lets are formed in the kernel induced feature space. The data-set is re-indexed according to these group-lets. Later, kernel k-means clustering method is applied over the set of leaders to derive a partition of the leaders set. Finally, each leader is replaced by its group to get a partition of the entire data-set. The time complexity of the proposed method

*Corresponding Author, Tel. No. +91-9493239032, Fax: +91-8514-275123
is $O(n + p^2)$, where $p$ is the size of the leaders set. Its space complexity is also $O(n + p^2)$. Experimental studies with data-sets of varying sizes shows that with a small loss of quality, the proposed algorithm can significantly reduce the computation time, particularly for large data-sets.

*Keywords:* Data mining, Unsupervised classification, kernel k-means clustering method, prototypes.

1. Introduction

Kernel k-means clustering method has been proved to be effective to identify clusters which are non-isotropic and linearly inseparable in the input space [1, 2]. It is an iterative method, wherein, first the data points are mapped from the input space to higher dimensional feature space through a non-linear transformation $\phi(\cdot)$ and then minimizes the clustering error, similar to that in k-means method [3, 4], but in the feature space. The distance between a data point and a cluster center in the feature space can be computed using a *kernel function* without knowing the explicit form of the transformation [5]. This is, because, the dot product between two data points $x$ and $y$ in the feature space, which is $\phi(x) \cdot \phi(y)$, can be computed as a function $k(x, y)$, where $k : \mathcal{D} \times \mathcal{D} \rightarrow \mathbb{R}$ is called the *kernel function*. Here $\mathcal{D}$ is the data-set and $\mathbb{R}$ is the set of real numbers. This is often known as the kernel trick and is valid for transformations that satisfies Mercer’s conditions [6]. Some standard kernel functions are given below.

- Linear kernel: $k(x_i, x_j) = x_i \cdot x_j$

- Polynomial kernel of degree $d$: $k(x_i, x_j) = (x_i \cdot x_j + 1)^d$, where $d$ is a positive integer.
- Radial (RBF) kernel: $k(x_i, x_j) = \exp\left(-\frac{{||x_i-x_j||}^2}{2\sigma^2}\right)$, where $\sigma \in \mathbb{R}$.

For two arbitrary data points $x_i$ and $x_j$, very often, in the iterative process of clustering, $k(x_i, x_j)$ is needed. So, a matrix called kernel matrix $K = [k_{ij}]_{n \times n}$ is found, where the $(i, j)^{th}$ entry is $k_{ij} = k(x_i, x_j)$. Here $n$ is the data-set size. The kernel matrix is precomputed and stored. So, the time and space requirements are $O(n^2)$. This is the drawback of kernel k-means clustering method and because of which it is not suitable when $n$ is large. Recently a new approach is proposed to overcome the problem with the space required in case of large data-sets. In this scheme the kernel matrix is divided into finite number of blocks such that size of each block is small enough to fit in the main memory. Later these blocks are processed individually in each iteration [7]. However, in each iteration, transferring each block from secondary memory to main memory is the additional computational burden. This method is discussed in detail, in subsequent sections.

Several improvements are proposed to cater the other drawbacks of this method such as prior knowledge about the number of clusters ($k$), and the local minima problem. Such improvements includes, The global kernel k-means method [8] and soft geodesic kernel k-means [9] methods. Further, semi-supervised clustering algorithms, such as semi-supervised kernel fuzzy c-means method [10], etc., were aimed to improve the clustering accuracy. Kernel based approaches, such as, kernel based c-means method [11], kernel based fuzzy c-means method, have been successfully used to deal with classification and clustering problems. Spectral clustering methods are also

\footnote{Some authors call the k-means clustering method as the c-means clustering method.}
used to identify non-linearly separable clusters in the input space. Some formal arguments show that, both kernel and spectral methods have some similarities [12, 13]. Some other related methods are reviewed in[14].

Often, in many situations it is enough to derive an approximate partition of the data-set, but in a lesser time. The scope of this paper is to present a simple prototype based hybrid approach to speed-up the kernel k-means clustering method which is suitable for large data-sets. The proposed method is called prototype based hybrid kernel k-means clustering method. Our method runs in two stages. In the first stage, the data-set is partitioned into a number of group-lets by employing a fast clustering method called the leaders clustering method [15]. Each group-let is represented by a prototype called its leader. The paper presents a significant modification to the conventional leaders clustering method, such that, the group-lets are found in the kernel space (not in the input space), but their leaders are represented by patterns in the input space. Some additional information such as the number of patterns existing in each group-let (called the count of that prototype) can be computed without any additional computational cost. In the second stage, kernel k-means clustering method is applied with the set of leaders to derive a partition of the set of leaders. Finally, each leader is replaced by its group to get a partition of the data-set. The proposed hybrid approach runs faster than the conventional kernel k-means method, but, since the method is an approximate one, the quality of the final clustering result is slightly degraded. The information, like the count of each prototype, is appropriately used to reduce such deviation. Moreover the proposed method is applicable for large data-sets.
The paper is organized as follows. Section 2 briefly reviews the conventional kernel k-means clustering method. Section 3 describes the existing block based approach for kernel k-means method, which is proposed for large data-sets. Section 4 outlines the leaders clustering method and its modified version proposed in this paper. The main contribution of this paper, prototype based hybrid kernel k-means clustering method, is described in Section 5. Experimental results are given in Section 6 and Section 7 gives some of the conclusions and future work.

2. kernel k-means clustering method

Let \( D = \{x_1, x_2, \ldots, x_n\} \) be the data-set of size \( n \), \( k \) be the number of clusters required, \( \mu^{(0)} \) be the initial seed points and \( \pi^{(0)} \) be the initial partition of the data-set. kernel k-means clustering method is an iterative method, takes \( k, \mu^{(0)} \) and an initial partition \( \pi^{(0)} \) as input and produces the final partition of the entire data-set, \( \Pi_D \) as the output.

The objective function is to minimize the criterion function

\[
J = \sum_{j=1}^{k} \sum_{\phi(x_i) \in C_j} ||\phi(x_i) - m_j||^2
\]

where \( m_j \) is the mean of the cluster \( C_j \) in the induced space is

\[
m_j = \frac{\sum_{\phi(x_i) \in C_j} \phi(x_i)}{|C_j|}
\]

Distance between two data points \( \phi(x_i) \) and \( \phi(x_j) \) in the induced space, that is
\[ ||\phi(x_i) - \phi(x_j)||^2 = \phi^2(x_i) - 2\phi(x_i) \cdot \phi(x_j) + \phi^2(x_j) \]
\[ = k(x_i, x_i) - 2k(x_i, x_j) + k(x_j, x_j). \]

Further, \[ ||\phi(x_i) - m_j||^2 \] can be calculated without knowing the transformation \[ \phi(\cdot) \] explicitly, as given below:

\[ ||\phi(x_i) - m_j||^2 = ||\phi(x_i) - \sum_{\phi(x_l) \in C_j} \frac{\phi(x_l)}{|C_j|}||^2 \]
\[ = \phi(x_i) \cdot \phi(x_i) - F(x_i, C_j) + G(C_j), \]

where
\[ F(x_i, C_j) = -\frac{2}{|C_j|} \sum_{\phi(x_l) \in C_j} \phi(x_l) \cdot \phi(x_i) \]
and
\[ G(C_j) = \frac{1}{|C_j|^2} \sum_{\phi(x_l) \in C_j} \sum_{\phi(x_s) \in C_j} \phi(x_l) \cdot \phi(x_s) \]

The iterative process of kernel k-means method is outlined in the Algorithm 1.


In case of large data-sets storing the kernel matrix \( K \) in the main memory is an infeasible task. Recently, Rudnichy et.al., proposed a block based approach to address this problem [7]. In this block based approach, the kernel matrix \( K \) is computed before starting the iterative process and it is stored in the secondary memory. That is the size of \( K \) can be theoretically extended
Algorithm 1 kernel k-means clustering method ($D, k, \mu^{(0)}, \pi^{(0)}$)

1. For each cluster $C_j$, find $|C_j|$ and $G(C_j)$.
2. Compute $F(x_i, C_j)$ for each $x_i$ and for each cluster $C_j$.
3. Find $||\phi(x_i) - m_j||^2$ using equation (4) and assign $x_i$ to its nearest center.
4. Update $m_j$, for $j = 1$ to $k$, using equation (2).
5. Repeat step 1 through step 4 till convergence.

Output:
The final Partition $\Pi_D = \{C_1, C_2, \ldots, C_k\}$.

to as large as the entire disk. Later, $K$ is split into blocks, where size of each block is determined according to the I/O capability and the available main memory size. In each iteration, each block is moved as a whole form secondary memory to the main memory and processed. So the number of I/O operations is equal to the number of blocks, but it is a costly operation when there are more number of blocks. Hence this approach is also not a good choice for large data-sets.

4. kernel based leaders clustering method

Leaders clustering method is single scan clustering method which produces a partition of the data-set in linear time [15], but the result is more unstable since it is highly dependent on scanning order of the data-set. However, recently in many hybrid methods, the leaders method is used to produce an intermediate partition of the data-set which is further reused to derive a strong and more stable clustering result [16, 17, 18, 19, 20]. In this paper
we present a modified version of the conventional leaders clustering method, called, kernel based leaders clustering method. The important modifications proposed for the conventional leaders clustering method are as follows:

- The group-lets are found in the kernel space, not in the input space.
- Each group-let is represented by a prototype, called its leader, which is a pattern in the input space.
- Number of patterns in each group-let, called its count, is also stored.
- Finally the data-set is re-indexed according to these group-lets, such that all the patterns in each group-let can be retrieved easily.

The number of group-lets depends on the threshold $t$. The patterns in the same group-let are called the followers of the leader of that group-let. For a given threshold distance $t$, the kernel based leaders method maintains a set of leaders $\mathcal{L}$ and the number of followers of each leader $l$, which is $\text{count}(l)$. $\mathcal{L}$ is initially empty and is incrementally built. For each data point $x$ in the data-set $\mathcal{D}$, if there is a leader $l \in \mathcal{L}$, such that distance between $\phi(x)$ and $\phi(l)$ is less than or equal to $t$, then $x$ is assigned to the group-let represented by $l$ and $\text{count}(l)$ is incremented by 1. Otherwise $x$ itself becomes a new leader and is added to $\mathcal{L}$, and $\text{count}(l)$ becomes 1. The algorithm outputs the set of leaders $\mathcal{L}$, the number of followers of each leader i.e., $\text{count}(l)$ and the set of followers of each leader $l$ i.e., $\text{followers}(l)$. The kernel based leaders clustering method is given in Algorithm 2.
Algorithm 2 kernel based leaders clustering method($\mathcal{D}$, $t$)

\[ \mathcal{L} = \emptyset; \]

for each $x \in \mathcal{D}$ do

Find a leader $l \in \mathcal{L}$ such that $||\phi(l) - \phi(x)|| \leq t$ /* where $||\phi(l) - \phi(x)||$ can be computed using the equation 3 */

if there is no such $l$ or when $\mathcal{L} = \emptyset$ then

\[ \mathcal{L} = \mathcal{L} \cup \{x\}; \]
\[ count(x) = 1; \]
\[ followers(x) = \{x\}; \]

else

\[ count(l) = count(l) + 1; \]
\[ followers(l) = followers(l) \cup \{x\}; \]

end if

end for

Output:

\[ \mathcal{L}^* = \{< l, count(l), followers(l) > | l \text{ is a leader } \}. \]

5. Prototype based hybrid kernel k-means clustering method

This section describes the proposed hybrid kernel k-means clustering method. The method runs in two stages. First, a set of leaders $\mathcal{L}$ is found using the modified leaders clustering method as given in section 4. Along with the set of leaders $\mathcal{L}$, $count(l)$ and $followers(l)$ for each leader $l$ are also stored. Later, kernel k-means clustering method is applied over the set of leaders $\mathcal{L}$ to derive a partition of the set of leaders $\Pi_\mathcal{L}$. In each iteration of kernel k-means method, each leader $l_i$ is assigned to the cluster $C_j$ such
that \(||\phi(l_i) - m_j||^2\) is minimized. Here we assume that all the patterns in a

group-let are very near to their leader (almost at the same point where their
leader exist) in the induced space. Hence \(||\phi(l_i) - m_j||^2\) can be computed as

follows.

\[
||\phi(l_i) - m_j||^2 = ||\phi(l_i) - \frac{\sum_{\phi(l_r) \in C_j} \phi(l_r)}{\sum_{\phi(l_r) \in C_j} \text{count}(l_r)}||^2
\]

\[
= \phi(l_i) \cdot \phi(l_i) - F(l_i, C_j) + G(C_j),
\]

where

\[
F(l_i, C_j) = \frac{2}{\sum_{\phi(l_r) \in C_j} \text{count}(l_r)} \sum_{\phi(l_r) \in C_j} \{\text{count}(l_r) \times k(l_i, l_r)\}
\]

and

\[
G(C_j) = \frac{1}{|\sum_{\phi(l_r) \in C_j} \text{count}(l_r)|^2} \{H_1 + H_2\}
\]

where

\[
H_1 = \sum_{\phi(l_r) \in C_j} \{\text{count}(l_r)^2 \times k(l_r, l_r)\}
\]

and

\[
H_2 = \sum_{\phi(l_r) \in C_j} \sum_{\phi(l_s) \in C_j} \{\text{count}(l_s) \times k(l_r, l_s)\} \text{ for } l \neq s
\]

Finally, at the end of the iterative process, each leader is replaced by all
its followers to get a partition of the entire data-set \(\Pi_D\).

Let the size of the leaders set be \(p\). In our method, the running time
taken for each iteration is significantly small, as it is working with only a
few prototypes but not with the entire data-set. The time complexity of
conventional kernel k-means method is \(O(n^2)\). The time taken to generate
the leaders in the induced space is \(O(n)\). The overall time complexity of the
proposed hybrid method is \(O(n + p^2)\). Often, \(p << n\), hence the proposed
The method is a faster one when compared to conventional kernel k-means method. The space complexity of the method is \(O(n + p^2)\). The proposed method is given in Algorithm 3.

The time and space complexities of our method depend on the number of leaders which in turn depends on the threshold \(t\). Increasing the \(t\) value can decrease the number of leaders \(p\), but, the deviation in the quality of the clustering result also might be increased. Experimental it is studied how \(t\) affects the running time of the proposed method and also the clustering quality using the proposed method.

\begin{algorithm}
\caption{Prototype based hybrid kernel k-means method(\(D, t\))}
\begin{enumerate}
\item Generate set of leaders \(L\) by using the leaders clustering method given in Algorithm 2 and store \(L^*\).
\item Compute the initial partition \(\pi^{(0)}\) of the leaders set \(L\) using the given initial seed points \(\mu^{(0)}\).
\item Apply kernel k-means clustering method \((L, k, \mu^{(0)}, \pi^{(0)})\), where, in each iteration, the nearest cluster for a leader is found using equation 7. Let \(\Pi_L\) be the output.
\item Replace each leader \(l\), by its group-let \((i.e., \text{with all its followers})\) to get the partition of the entire data-set \(\Pi_D\).
\item Output \(\Pi_D\).
\end{enumerate}
\end{algorithm}

6. Experimental Study

This section describes the experiments performed with some benchmark data-sets as well as with some synthetic data-sets. The data-sets employed
are: Banana, Rings, Concentric Circles, Desert, Iris, Pendigits, Optical Character recognition (OCR), Letter Image Recognition (LIR) data-set, Shuttle data-set and Gaussian data-sets. Iris and Pendigits data-sets are available at the UCI machine learning repository [21]. Letter Image Recognition (LIR) and Shuttle data-sets are also used in [18]. Banana, Rings, Concentric Circles, Desert and Gaussian data-sets are artificially generated. OCR data-set is also used in[22, 23, 18].

The synthetic data-sets used in our experiments includes various types of clusters. See Figure 1. The Gaussian data-set is generated from a tri-modal gaussian distribution 
\[ p(x) = \frac{1}{3}N(\mu_1, \Sigma_1) + \frac{1}{3}N(\mu_2, \Sigma_2) + \frac{1}{3}N(\mu_3, \Sigma_3). \]
For dimensionality 2, \( \mu_1 = (0, 0), \mu_2 = (5, 5), \mu_3 = (-5, 5). \) The covariance matrix is taken as the identity matrix of size \( d \times d, \) where \( d \) is the dimensionality of the data. All the data-sets have only numeric valued features. The properties of the data-sets are given in Table 1.

To calculate the value of the threshold \( (t) \), used to generate the set of leaders in Algorithm 2, a random sample of 5% is selected from the given data-set and average of all distinct pair wise distances(from that sample) in the induced space is computed. Finally, 1% to 100% of such average distance is taken as the the range for \( t. \) The RBF kernel is used in both hybrid kernel k-means and conventional kernel k-means methods because of its superiority over other kernel functions.

Experiments are conducted on a PC with an intel P4 processor (3.2 Ghz) with 512 MB RAM. The proposed method is compared with the conventional kernel k-means method in the case where the kernel matrix is large enough to be stored in main memory and also the proposed method is compared
Figure 1: Two-dimensional artificial data-sets used in the experiments, (a) Banana data-set, (b) Rings data-set, (c) Desert data-set, and (d) Concentric Circles data-set.
Table 1: Details of data-sets used

<table>
<thead>
<tr>
<th>Data-set</th>
<th>Number of data points</th>
<th>dimensionality</th>
<th>Number of Clusters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Banana</td>
<td>200</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Rings</td>
<td>250</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Concentric Circles</td>
<td>230</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Desert</td>
<td>350</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Iris</td>
<td>150</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Pendigits</td>
<td>10992</td>
<td>16</td>
<td>10</td>
</tr>
<tr>
<td>OCR data-set</td>
<td>10003</td>
<td>192</td>
<td>10</td>
</tr>
<tr>
<td>LIR data-set</td>
<td>20000</td>
<td>16</td>
<td>26</td>
</tr>
<tr>
<td>Shuttle data-set</td>
<td>58000</td>
<td>9</td>
<td>7</td>
</tr>
<tr>
<td>Gaussian data-set</td>
<td>60000</td>
<td>2</td>
<td>3</td>
</tr>
</tbody>
</table>
with block based kernel k-means method in case of large data-sets. The comparison is done both in terms of Clustering Accuracy (CA) and the overall Running Time (RT). CA of a clustering method \( M \) is the percentage of similarity between the two partitions, one is the actual partition of the data-set and the other is the partition obtained using the clustering method \( M \), where the similarity is found using Rand-Index \([24, 25]\).

The clustering accuracy and the overall running time of the proposed method depends on the value of \( t \). Experiments are performed to analyze the effect of \( t \) on CA and RT for various data-sets specified in table 1. Figure 2 and figure 3 presents the clustering accuracy (CA) and the total running time (RT) with respect to \( t \) for Pendigits, LIR, Shuttle and Gaussian data-sets. Figure 2(a) and 2(b) shows the clustering accuracy (CA) and the total running time (RT) with respect to \( t \) for Pendigits data-set, whereas 2(c) and 2(d) shows the clustering accuracy (CA) and the total running time (RT) with respect to \( t \) for LIR data-set. Similarly, figure 3(a) and 3(b) shows the clustering accuracy (CA) and the total running time (RT) with respect to \( t \) for Shuttle data-set, whereas 3(c) and 3(d) shows the clustering accuracy (CA) and the total running time (RT) with respect to \( t \) for Gaussian data-set. For the remaining data-sets also the similar types of graphs are obtained.

The values of clustering accuracy and running time achieved by kernel k-means method and the proposed hybrid kernel k-means method (for the best threshold \( t \) in the specified range), for each one of the data-sets are calculated and the results are presented in table 2. For shuttle and Gaussian data-sets, the conventional kernel k-means method is infeasible, hence, the block based approach is used in this case. The results for these data-sets are
Table 2: Clustering Accuracy (CA) and Running Time (RT) using kernel k-means method and the proposed hybrid kernel k-means methods

<table>
<thead>
<tr>
<th>Data-set</th>
<th>CA (%) using kernel k-means method</th>
<th>Running Time of the kernel k-means method (In seconds)</th>
<th>CA (%) using the Proposed method</th>
<th>Running Time of the proposed method (In seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Banana</td>
<td>90.56</td>
<td>0.022</td>
<td>89.22</td>
<td>0.007</td>
</tr>
<tr>
<td>Rings</td>
<td>100</td>
<td>0.054</td>
<td>89.25</td>
<td>0.003</td>
</tr>
<tr>
<td>Concentric Circles</td>
<td>74.4</td>
<td>0.029</td>
<td>73.6</td>
<td>0.003</td>
</tr>
<tr>
<td>Desert</td>
<td>74.88</td>
<td>0.096</td>
<td>72.39</td>
<td>0.042</td>
</tr>
<tr>
<td>Iris</td>
<td>89.76</td>
<td>0.019</td>
<td>88.59</td>
<td>0.004</td>
</tr>
<tr>
<td>Pendigits</td>
<td>92.78</td>
<td>418.078</td>
<td>89.56</td>
<td>20.123</td>
</tr>
<tr>
<td>OCR data-set</td>
<td>86.38</td>
<td>841.500</td>
<td>82.00</td>
<td>34.777</td>
</tr>
<tr>
<td>LIR data-set</td>
<td>92.92</td>
<td>3261.337</td>
<td>89.96</td>
<td>172.901</td>
</tr>
</tbody>
</table>

given separately in the table 3.

These experimental results shows that the proposed prototype based hybrid kernel k-means method takes less time when compared to conventional k-means clustering method as well as the block based approach proposed for large data-sets.

The percentage of reduction in clustering accuracy and the percentage of reduction in running time are calculated as follows.
Table 3: Clustering Accuracy (CA) and the Running Time (RT) using block based kernel k-means method and the proposed hybrid kernel k-means methods

<table>
<thead>
<tr>
<th>Data-set</th>
<th>CA (%) using block based kernel k-means method</th>
<th>Running Time of block based kernel k-means method (In minutes)</th>
<th>CA (%) using the Proposed method</th>
<th>Running Time of the proposed method (In minutes)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Shuttle data-set</td>
<td>93.12</td>
<td>263</td>
<td>90.14</td>
<td>38</td>
</tr>
<tr>
<td>Gaussian data-set</td>
<td>98.32</td>
<td>334</td>
<td>92.33</td>
<td>32</td>
</tr>
</tbody>
</table>
Table 4: Percentage of reduction in clustering accuracy and Percentage of reduction in running time for all the data-sets used

<table>
<thead>
<tr>
<th>Data-set</th>
<th>Percentage of reduction in Clustering Accuracy</th>
<th>Percentage of reduction in Running Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Banana</td>
<td>1.48</td>
<td>68.18</td>
</tr>
<tr>
<td>Rings</td>
<td>10.5</td>
<td>94.44</td>
</tr>
<tr>
<td>Concentric Circles</td>
<td>1.08</td>
<td>89.65</td>
</tr>
<tr>
<td>Desert</td>
<td>3.33</td>
<td>56.25</td>
</tr>
<tr>
<td>Iris</td>
<td>1.3</td>
<td>78.94</td>
</tr>
<tr>
<td>Pendigits</td>
<td>3.47</td>
<td>95.18</td>
</tr>
<tr>
<td>OCR data-set</td>
<td>5.1</td>
<td>95.87</td>
</tr>
<tr>
<td>LIR data-set</td>
<td>3.19</td>
<td>94.69</td>
</tr>
<tr>
<td>Shuttle data-set</td>
<td>3.2</td>
<td>85.55</td>
</tr>
<tr>
<td>Gaussian data-set</td>
<td>6.1</td>
<td>90.41</td>
</tr>
</tbody>
</table>
Percentage of reduction in clustering accuracy is

\[
\frac{CA^{(1)} - CA^{(2)}}{CA^{(1)}} \times 100
\]  

(12)

where \(CA^{(1)}\) is the clustering accuracy obtained using k-means or kernel k-mean clustering method and \(CA^{(2)}\) is the clustering accuracy using the proposed prototype based hybrid method. Similarly

Percentage of reduction in Running Time is

\[
\frac{RT^{(1)} - RT^{(2)}}{RT^{(1)}} \times 100
\]  

(13)

where \(RT^{(1)}\) is the running time for k-means or kernel k-mean clustering method and \(RT^{(2)}\) is the running time for the proposed prototype based hybrid method. For \(t\) approximately equal to 5% to 8% of average random pair wise distance, the percentage of reduction in clustering accuracy and the percentage of reduction in running time for each data-set is given in tables 4.

On the whole, for \(t\) approximately equal to 5% to 8% of average random pair wise distance, the percentage of reduction in clustering accuracy is 3% to 5%, whereas, the percentage of reduction in running time is 82% to 90%. Hence in this experimental study, it is observed that with a small loss in the clustering quality the proposed prototype based hybrid kernel k-means method runs in a faster pace.

7. Conclusions and Future Work

The paper presented a new hybrid approach to speed up the kernel k-means clustering method. In this hybrid approach, first, the data-set is partitioned into small group-lets in the induced kernel space. These group-lets
Figure 2: Clustering Accuracy (CA) and Running Time (RT) for Pendigits and LIR datasets with respect to the threshold $t$. (a) and (b) Clustering Accuracy and Running Time (in seconds) with respect to $t$ for Pendigits data-set. (c) and (d) Clustering Accuracy and Running Time (in seconds) with respect to $t$ for LIR data-set.
Figure 3: Clustering Accuracy (CA) and Running Time (RT) for Shuttle and Gaussian data-sets with respect to the threshold $t$. (a) and (b) Clustering Accuracy and Running Time (in minutes) with respect to $t$ for Shuttle data-set. (c) and (d) Clustering Accuracy and Running Time (in minutes) with respect to $t$ for Gaussian data-set.
are found using kernel based leaders clustering method, which is a modified version of the conventional leaders clustering method, such that the grouplets are found in the kernel induced feature space. Leader of each grouplet is represented by its corresponding pattern in the input space. Later, these representative prototypes are provided as input for kernel k-means method which produces a partition of the set of prototypes. Finally, each prototype is replaced by its grouplet to get a partition of the entire data-set. The proposed method runs in a faster pace. However, there may be small reduction in the quality of the clustering result. The percentage of reduction in the clustering quality can be reduced by selecting an appropriate value for the threshold $t$, which is an input parameter for the kernel based leaders clustering method. This paper experimentally studied how the threshold $t$ affects the clustering quality and the running time of the proposed method. Based on our experimental study, for $t$ approximately equal to 5% to 8% of average random pair wise distance, in a random sample which is 5% of the data-set, the percentage of reduction in the clustering accuracy is 3% to 5%, whereas, the percentage of reduction in running time is 82% to 90%. Hence, with a small loss in the clustering quality the proposed method considerably reduces the running time. Future scope of this work is to study theoretically, how to select an appropriate value of the threshold $t$ for a given data-set.

References


