The Distance-Based Representative Skyline Calculation Using Unsupervised Extreme Learning Machines

Mei Bai, Junchang Xin, Guoren Wang and Xite Wang

Abstract *A representative skyline* contains k skyline points that can represent its full skyline, which is very useful in the multiple criteria decision making problems. In this paper, we focus on the distance-based representative skyline (k-DRS) query which can describe the tradeoffs among different dimensions offered by the full skyline. Since k-DRS is a NP-hard problem in d-dimensional ($d \geq 3$) space, it is impossible to calculate the exact k-DRS in d-dimensional space. By in-depth analyzing the properties of the k-DRS, we propose a new perspective to solve this problem and a k distance-based representative skyline algorithm based on US-ELM (DRSELM) is presented. In DRSELM, first we apply US-ELM to divide the full skyline set into k clusters. Second, in each cluster, we design a method to select a point as the representative point. Experimental results show that our DRSELM significantly outperforms its competitors in terms of both accuracy and efficiency.

Keywords Skyline · k representative skyline · k-DRS · US-ELM

1 Introduction

Given a large dataset, it is impracticable for a user to browse all the points in the dataset. Hence, obtaining a succinct representative subset of the dataset is crucial. A well-established approach to representing a dataset is with the *skyline* operator [1].

M. Bai (✉) · J. Xin · G. Wang · X. Wang
College of Information Science & Engineering, Northeastern University,
Shenyang, People's Republic of China
e-mail: baimei861221@163.com

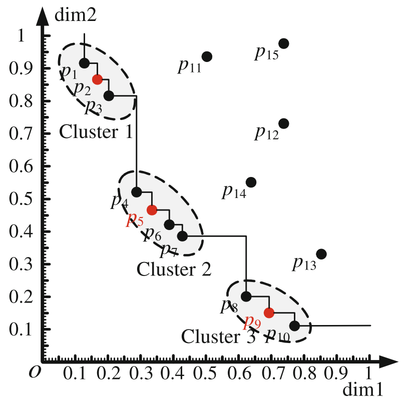
J. Xin
e-mail: xinjunchang@ise.neu.edu.cn

G. Wang
e-mail: wanggr@mail.neu.edu.cn

X. Wang
e-mail: wangxite@research.neu.edu.cn

Fig. 1 Example of distance-based representative skyline

p_1<0.13, 0.92>
p_2<0.16, 0.87>
p_3<0.20, 0.82>
p_4<0.27, 0.52>
p_5<0.32, 0.46>
p_6<0.37, 0.42>
p_7<0.42, 0.37>
p_8<0.63, 0.20>
p_9<0.70, 0.15>
p_{10}<0.77, 0.10>



The skyline consists of the points which are not *dominated* by any other point. Given two points p_1 and p_2, if the values of p_1 are as good as or better than those of p_2 in any dimension, and better in at least one dimension. With loss of generality, we assume that a smaller value indicates a better performance in all dimensions. However, when the skyline size is large, the full skyline is helpless to the user. Detecting a subset of the full skyline set with fixed size (such as k points) is necessary. As investigated in [2], Tao et al. proposed a *distance-based representative skyline* (k-DRS for short) which can best describe the tradeoffs among different dimensions offered by the full skyline. They applied a *distance* metric to measure the "representativeness" of the chosen set. Given a subset $\mathcal{K}$ with k skyline points from the full skyline set $\mathcal{S}$, $Er(\mathcal{K}, \mathcal{S}) = \max_{p \in \mathcal{S}-\mathcal{K}} \min_{p' \in \mathcal{K}} \parallel p, p' \parallel$, where $\parallel p, p' \parallel$ is the Euclidean distance between p and p'. The k-DRS is the set $\mathcal{K}$ with the minimum value $Er(\mathcal{K}, \mathcal{S})$. As illustrated in Fig. 1, given $k = 3$, the 3-DRS is $\{p_2, p_5, p_9\}$ with the corresponding value $Er(\mathcal{K}, \mathcal{S}) = \parallel p_5, p_7 \parallel = 0.134$. Obviously, when the full skyline are divided into k clusters, k-DRS aims to select k skyline points from these k clusters and these k skyline points should come from different clusters.

In this paper, we deeply analyze the properties of k-DRS query, and solve the problem using the extreme learning machine (ELM for short) [3–5]. Compared with support vector machines (SVMs) [6, 7], ELM shows better predicting accuracy than that of SVMs [4, 8–10]. Moreover, various extensions have been made to the basic ELMs to make it more efficient and more suitable for special problems. such as ELMs for online sequential data [10–12], ELMs for distributed environments [13], and ELMs for semi-labeled data and unlabeled data [14]. As proposed in [14], US-ELM can be applied to unsupervised data which has more widely applications. Meanwhile, the experiments show that US-ELM gives favorable performance compared to the state-of-the-art clustering algorithms [15–18]. Therefore, we apply US-ELM to cluster the full skyline points, and then select the appropriate the skyline points from every cluster as the k-DRS.

As mentioned in Lemma 4 in [2], k-DRS is NP-hard when the dimensionality $d \geq 3$. Hence, it is impossible to calculate the exact k-DRS in d-dimensional space

($d \geq 3$). To solve the challenging issue, we attempt to solve k-DRS problem from another perspective.

The key contributions are summarized as follow. Through in-depth analysis of k-DRS properties, we propose a k distance-based representative skyline algorithm based on US-ELM methods (DRSELM for short). The calculation of the k-DRS is divided into two stages. Step 1: the full skyline set is divided into k clusters by using the US-ELM algorithm. Step 2: for each cluster, an appropriate skyline point is added to the k-DRS set. The chosen k skyline points are the k-DRS result. Then we test our algorithm on a variety of data sets, and comparisons with other related algorithms [2]. The results show that our algorithm is competitive in terms of both accuracy and efficiency.

The rest of paper is organized as follows. In Sect. 2, we give a brief overview of clustering data using US-ELM algorithm. In Sect. 3, we present our k distance-based representative skyline algorithm based on US-ELM. Experimental results and related work are given in Sects. 4 and 5, respectively. Section 6 concludes the paper.

2 Preliminaries

In this paper, we process k-DRS query using ELM to cluster the skyline points. Here, we introduce how to extend ELMs to cluster the data.

2.1 Brief Introduction to ELM

ELM is an algorithm for neural network, and is a single-hidden layer feed forward network. ELM aims to learn a decision rule or an approximation function based on a training set with N samples, $\{\boldsymbol{X}, \boldsymbol{Y}\} = \{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{N}$, where $\boldsymbol{x}_i \in \mathbb{R}^{n_i}$ and $\boldsymbol{y}_i \in \mathbb{R}^{n_o}$, n_i and n_o are the dimensions of input and output, respectively.

As described in Fig. 2, the training of ELMs contains two phases. Step 1: a pair of parameters $\{\boldsymbol{a}_j, b_j\}$ are randomly generated for the jth hidden layer node, where $\boldsymbol{a}_j$ is a n_i-dimensional vector and b_j is a random value. For an input vector $\boldsymbol{x}_i$, its output on the jth hidden node can be obtained by the following mapping function (we use

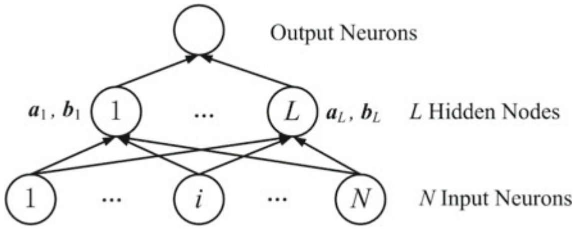

Output Neurons
$\boldsymbol{a}_1, \boldsymbol{b}_1$
1
...
L
$\boldsymbol{a}_L, \boldsymbol{b}_L$
L Hidden Nodes
1
...
i
...
N
N Input Neurons

Fig. 2 ELM framework

the Sigmoid function in this paper).

$$g(\boldsymbol{x}_i, \boldsymbol{a}_j, b_j) = \frac{1}{1 + exp(-(\boldsymbol{a}_j^T \times \boldsymbol{x}_i + b_j))} \tag{1}$$

Hence, the output on the hidden layer nodes can be written as

$$\boldsymbol{H} = \begin{bmatrix} g(\boldsymbol{x}_1, \boldsymbol{a}_1, b_1) & \dots & g(\boldsymbol{x}_1, \boldsymbol{a}_L, b_L) \\ \vdots & & \vdots \\ g(\boldsymbol{x}_N, \boldsymbol{a}_1, b_1) & \dots & g(\boldsymbol{x}_N, \boldsymbol{a}_L, b_L) \end{bmatrix}_{N \times L} \tag{2}$$

Step 2: On the *j*th hidden node, an adjustment factor $\boldsymbol{\beta}_j$ is generated, where $\boldsymbol{\beta}_j$ is a n_o-dimensional vector. The output on the output neuron is $\boldsymbol{Y}$ which is the output of the N samples. Then we can obtain the following equation

$$\boldsymbol{H} \cdot \boldsymbol{\beta} = \boldsymbol{Y} \tag{3}$$

where

$$\boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta}_1 \\ \vdots \\ \boldsymbol{\beta}_L \end{bmatrix}_{L \times n_o} \quad and \; \boldsymbol{Y} = \begin{bmatrix} \boldsymbol{y}_1 \\ \vdots \\ \boldsymbol{y}_N \end{bmatrix}_{N \times n_o} \tag{4}$$

According to Eq. 3, the values of $\boldsymbol{H}$ and $\boldsymbol{Y}$ haven been known, we can compute the values of $\boldsymbol{\beta}$ by the equation $\boldsymbol{\beta} = \boldsymbol{H}^{\dagger}\boldsymbol{Y}$ where $\boldsymbol{H}^{\dagger}$ is the Moore-Penrose [19] of $\boldsymbol{H}$. In order to avoid over-fitting, they introduced two parameters, $\boldsymbol{e}_i$ and C. $\boldsymbol{e}_i$ is the error vector with respect to the *i*th training sample, and C is a penalty coefficient on the training errors. Then the following equation is used to generate $\boldsymbol{\beta}$.

$$\begin{aligned} &\min_{\boldsymbol{\beta} \in \mathbb{R}^{L \times n_o}} L_{ELM} = \frac{1}{2} \parallel \boldsymbol{\beta} \parallel^2 + \frac{C}{2} \parallel \boldsymbol{Y} - \boldsymbol{H}\boldsymbol{\beta} \parallel^2 \\ &s.t. \; \boldsymbol{H}\boldsymbol{\beta} = \boldsymbol{Y} - \boldsymbol{e} \end{aligned} \tag{5}$$

where $\parallel \cdot \parallel$ denotes the Euclidean norm and $\boldsymbol{e} = [\boldsymbol{e}_1^T, \dots, \boldsymbol{e}_N^T] \in \mathbb{R}^{N \times n_o}$.

According to the ridge regression or regularized least squares principle, the gradient of L_{ELM} with respect to $\boldsymbol{\beta}$ is set to zero. We have

$$\nabla L_{ELM} = \boldsymbol{\beta} + C\boldsymbol{H}^T(\boldsymbol{Y} - \boldsymbol{H}\boldsymbol{\beta}) = 0 \tag{6}$$

If $\boldsymbol{H}$ has more rows than columns and is full of column rank, we use Eq. 7 to evaluate $\boldsymbol{\beta}$. If the number of training samples N is smaller than L, we restrict $\boldsymbol{\beta}$ to be a linear combination of the rows of $\boldsymbol{H}$: $\boldsymbol{\beta} = \boldsymbol{H}^T\boldsymbol{\alpha}(\boldsymbol{\alpha} \in \mathbb{R}^{N\times n_o})$. Then $\boldsymbol{\beta}$ can be calculated by Eq. 8.

$$\boldsymbol{\beta}^* = (\boldsymbol{H}^T\boldsymbol{H} + \frac{\boldsymbol{I}_L}{C})^{-1}\boldsymbol{H}^T\boldsymbol{Y} \tag{7}$$

$$\boldsymbol{\beta}^* = \boldsymbol{H}^T\boldsymbol{\alpha}^* = \boldsymbol{H}^T(\boldsymbol{H}\boldsymbol{H}^T + \frac{\boldsymbol{I}_N}{C})^{-1}\boldsymbol{Y} \tag{8}$$

where $\boldsymbol{I}_L$ and $\boldsymbol{I}_N$ are the identity matrices of dimensions L and N, respectively.

2.2 Unsupervised ELM

In [14], they extended ELM to process unlabeled data and made ELM a wide applications. The unsupervised learning is built on the following assumption: (1) all the unlabeled data $\boldsymbol{X}_u$ is drawn from the same marginal distribution $\mathcal{P}_{\boldsymbol{X}}$ and (2) if two points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are close to each other, then the probabilities $P(\boldsymbol{y}|\boldsymbol{x}_1)$ and $P(\boldsymbol{y}|\boldsymbol{x}_2)$ should be similar. The manifold regularization framework proposes to minimize the following cost function:

$$L_m = \frac{1}{2}\sum_{i,j} w_{ij} \parallel P(\boldsymbol{y}|\boldsymbol{x}_i) - P(\boldsymbol{y}|\boldsymbol{x}_j) \parallel^2 \tag{9}$$

where w_{ij} is the pair-wise similarity between $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$. w_{ij} is usually computed using Gaussian function $exp(- \parallel \boldsymbol{x}_i - \boldsymbol{x}_j \parallel^2 /2\sigma^2)$.

Equation 9 can be simplified in a matrix form

$$\widehat{L}_m = Tr(\widehat{\boldsymbol{Y}}^T\boldsymbol{L}\widehat{\boldsymbol{Y}}) \tag{10}$$

where $Tr(\cdot)$ denotes the trace of a matrix, $\widehat{\boldsymbol{Y}}$ is the predictions of $\boldsymbol{X}_u$, $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$ is known as graph Laplacian, and $\boldsymbol{D}$ ia a diagonal matrix with its diagonal elements $D_{ii} = \sum_{j=1}^{u} w_{ij}$.

Hence, in unsupervised setting, the entire data set $\boldsymbol{X} = \{\boldsymbol{x}_i\}_{i=1}^N$ are unlabeled. According to Eqs. 5 and 10, the formulation of US-ELM is reduced to

$$\min_{\boldsymbol{\beta}\in\mathbb{R}^{L\times n_o}} \parallel \boldsymbol{\beta} \parallel^2 + \lambda Tr(\boldsymbol{\beta}^T\boldsymbol{H}^T\boldsymbol{L}\boldsymbol{H}\boldsymbol{\beta}) \tag{11}$$

where λ is an tradeoff parameter. Usually, Eq. 11 attains its minimum value at $\boldsymbol{\beta} = \boldsymbol{0}$. As suggested in [16], a constraint $(\boldsymbol{H}\boldsymbol{\beta})^T\boldsymbol{H}\boldsymbol{\beta} = \boldsymbol{I}_{n_o}$ is introduced. According to the conclusion in [14], if $L \leq N$, the adjustment factor $\boldsymbol{\beta}$ is given by

$$\boldsymbol{\beta}^* = [\tilde{\boldsymbol{v}}_2, \tilde{\boldsymbol{v}}_3, \dots, \tilde{\boldsymbol{v}}_{n_o+1}] \tag{12}$$

where $\tilde{\boldsymbol{v}}_i = \boldsymbol{v}_i / \parallel \boldsymbol{H}\boldsymbol{v}_i \parallel, i = 2, \ldots, n_o + 1$ is the normalized eigenvectors. γ_i is the ith smallest eigenvalues of Eq. 13 and $\boldsymbol{v}_i$ is the corresponding eigenvectors.

$$(\boldsymbol{I}_L + \lambda \boldsymbol{H}^T \boldsymbol{L}\boldsymbol{H})\boldsymbol{v} = \gamma \boldsymbol{H}^T \boldsymbol{H}\boldsymbol{v} \tag{13}$$

If $L > N$, Eq. 13 is underdetermined. In this case, the following alternative formulation is given by using the same trick

$$(\boldsymbol{I}_N + \lambda \boldsymbol{L}\boldsymbol{H}\boldsymbol{H}^T)\boldsymbol{u} = \gamma \boldsymbol{H}\boldsymbol{H}^T \boldsymbol{u} \tag{14}$$

Also, $\boldsymbol{u}_i$ is the generalized eigenvectors corresponding the ith smallest eigenvalues of Eq. 14. Then, the final solution is given by

$$\boldsymbol{\beta}^* = H^T[\tilde{\boldsymbol{u}}_2, \tilde{\boldsymbol{u}}_3, \ldots, \tilde{\boldsymbol{u}}_{n_o+1}] \tag{15}$$

where $\tilde{\boldsymbol{u}}_i = \tilde{\boldsymbol{u}}_i / \parallel \boldsymbol{H}\boldsymbol{H}^T \tilde{\boldsymbol{u}}_i \parallel, i = 2, \ldots, n_o + 1$ are the normalized eigenvectors.

The US-ELM is described in Algorithm 1.

3 *k*-DRS Query Processing Based on US-ELM

First, we describe the formal definition of the *k*-DRS in Sect. 3.1. Then, our proposed algorithm DRSELM is presented in Sect. 3.2.

Algorithm 1: US-ELM Algorithm [14]

```
input  : The training data: $\boldsymbol{X} \in \mathbb{R}^{N \times n_i}$.
output : The label vector of cluster $\boldsymbol{y}_i$ corresponding to $\boldsymbol{x}_i$
1  Step 1: Construct the graph Laplacian $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$ from $\boldsymbol{X}$;
2  Step 2: For each hidden neuron, generate a pair of random values $\{\boldsymbol{a}_i, b_i\}$; Calculate the
   output matrix $\boldsymbol{H} \in \mathbb{R}^{N \times L}$;
3  Step 3:
4  if $L \leq N$ then
5      Find the generalized eigenvectors $\boldsymbol{v}_2, \ldots, \boldsymbol{v}_{n_o+1}$ of Eq. 13. Let $\boldsymbol{\beta} = [\tilde{\boldsymbol{v}}_2, \tilde{\boldsymbol{v}}_3, \ldots, \tilde{\boldsymbol{v}}_{n_o+1}]$.
6  else
7      Find the generalized eigenvectors $\boldsymbol{u}_2, \ldots, \boldsymbol{u}_{n_o+1}$ of Eq. 14. Let $\boldsymbol{\beta} = H^T[\tilde{\boldsymbol{u}}_2, \tilde{\boldsymbol{u}}_3, \ldots, \tilde{\boldsymbol{u}}_{n_o+1}]$;
8  Step 4: Calculate the embedding matrix: $\boldsymbol{E} = \boldsymbol{H}\boldsymbol{\beta}$;
9  Step 5: Each row of $\boldsymbol{E}$ is treated as a point, and then cluster these $N$ points into $K$ clusters
   using the $k$-means algorithm. Let $\boldsymbol{y}_i$ be the label vector of cluster index for $\boldsymbol{x}_i$.
10 return $\boldsymbol{Y}$;
```

3.1 Problem Statement

Given a data set D in the d-dimensional space, and two points $p_i = \langle p_i[1], \ldots, p_i[d] \rangle$ and $p_j = \langle p_j[1], \ldots, p_j[d] \rangle$, then p_i *dominates* p_j (denoted as $p_i \prec p_j$) if $\forall m \in [1, d]$, $p_i[m] \leq p_j[m]$ and $\exists n \in [1, d], p_i[n] < p_j[n]$. The *skyline* of D consists of the points which are not dominated by others, denoted as $\mathcal{S} = \{p_i | \nexists p_j \in D, p_j \prec p_i\}$. Next, we give the formal definition of the k-DRS.

Definition 1 (*Representation Error*) Given the full skyline set $\mathcal{S}$ and a subset $\mathcal{K}$ of $\mathcal{S}$ with k skyline points, the *representation error* $Er(\mathcal{K}, \mathcal{S})$ quantifies the representation quality as the maximum distance between a non-representative skyline point in $\mathcal{S} - \mathcal{K}$ and its nearest representative in $\mathcal{K}$, which is formally denoted as:

$$Er(\mathcal{K}, \mathcal{S}) = \max_{p \in \mathcal{S} - \mathcal{K}} \{ \min_{p' \in \mathcal{K}} \parallel p, p' \parallel \} \tag{16}$$

Definition 2 (*Distance-based Representative Skyline*) The distance-based representative skyline (k-DRS) is the set $\mathcal{K}$ with the minimum representation error $Er(\mathcal{K}, \mathcal{S})$.

As shown in Fig. 1, the skyline set is $\mathcal{S} = \{p_1, \ldots, p_{10}\}$. Given $k = 3$, and two subsets $\mathcal{K}_1 = \{p_2, p_6, p_9\}$ and $\mathcal{K}_2 = \{p_2, p_5, p_9\}$, the representation errors $Er(\mathcal{K}_1, \mathcal{S}) = \parallel p_4, p_6 \parallel = 0.141$ and $Er(\mathcal{K}_2, \mathcal{S}) = \parallel p_5, p_7 \parallel = 0.135$. Consequently, $\mathcal{K}_2$ is the k-DRS because its representation error is the minimum.

3.2 DRSELM Algorithm

Reviewing the conclusion in [2], the k-DRS problem is NP-hard in d-dimensional ($d \geq 3$) space. Hence, it is impossible to calculate the exact k-DRS. In this paper, we answer the k-DRS problem from another perspective.

Since the initial objective of the k-DRS is to avoid selecting k points that appear in an arbitrarily tiny cluster, we first divide the full skyline points into k clusters using the US-ELM algorithm (introduced in Sect. 2.2). Specifically, given a data set D in d-dimensional space, each point $p_i \in D$ is considered as a d-dimensional vector. $p_i[j]$ denotes the jth dimension value of p_i. According to the Algorithm 1, there is a corresponding output y_i with regard to p_i. Because the full skyline needs to be divided into k clusters, each output y_i is a k-dimensional vector. Only one dimension value is 1, and the other dimension values are 0. As shown in Fig. 1, p_1, p_2, p_3 have the same outputs $y_1 = y_2 = y_3 = [1, 0, 0]$. p_4, p_5, p_6, p_7 have the same outputs $y_4 = y_5 = y_6 = y_7 = [0, 1, 0]$. p_8, p_9, p_{10} have the same outputs $y_8 = y_9 = y_{10} = [0, 0, 1]$.

Given a cluster $c_i = \{p_1, p_2, \ldots, p_{|c_i|}\}$ with $|c_i|$ points, the centroid point m_i of c_i can be calculated by the formula below:

$$m_i[j] = \frac{\sum_{p \in c_i} p[j]}{|c_i|} \tag{17}$$

As shown in Fig. 1, given the cluster $c_1 = \{p_1, p_2, p_3\}$, its centroid point m_1 is calculated as: $m_1[1] = \frac{0.13+0.16+0.20}{3} \approx 0.16$ and $m_1[2] = \frac{0.92+0.87+0.82}{3} = 0.87$. Hence, p_2 is regarded as the centroid point in c_1. Similarly, the centroid points of c_2 and c_3 are $m_2 = \langle 0.345, 0.4425 \rangle$ and $m_3 = \langle 0.70, 0.15 \rangle$.

After the clustering, we have the following properties.

Observation 1 *Given a point p_1 comes from cluster c_1, and a point p_2 comes from cluster c_2, m_1 and m_2 are the centroid points of c_1 and c_2, respectively. Then $\| p_1, m_1 \| < \| p_1, m_2 \|$.*

We have divided the full skyline into k clusters. The target of the k-DRS wants to get the minimum representation error $Er(\mathcal{K}, \mathcal{S})$. In order to obtain this goal, all the points should come from different clusters. Given two clusters c_i and c_j, we should select any 2 points $\mathcal{K}^2$ from $C = c_i \bigcup c_j$, in order to obtain the minimum value $Er(\mathcal{K}^2, C) = \max_{p \in C - \mathcal{K}^2} \{ \min_{p' \in \mathcal{K}^2} \| p, p' \| \}$. The two points in $\mathcal{K}^2$ should come from c_i and c_j, respectively.

Theorem 1 *Give two clusters c_1 and c_2, and two sets $S_1 = \{m_1, m_2\}$ and $S_2 = \{p_m, p_n\}$, m_1 and m_2 are the centroid points of c_1 and c_2. p_m and p_n are any two points come from c_1. Then $Er(S_1, C) < Er(S_2, C)$.*

Proof Suppose the point in c_1 with the largest distance to m_1 is p_1, and the point in c_2 with the largest distance to m_2 is p_2, then $Er(S_1, C) = \max\{\| p_1, m_1 \|, \| p_2, m_2 \|\}$. Obviously, a good clustering method can ensure that $\| m_1, m_2 \| > \max\{\| p_1, m_1 \|, \| p_2, m_2 \|\}$. Hence, there must exist a point $p' \in c_2$, the distance between p' and any point $p_m \in c_1$ is larger than $Er(S_1, C)$. The theorem can be proven.

Lemma 1 *Given k clusters $c_1, \ldots, c_k$ of the full skyline $\mathcal{S}$, in order to obtain the minimum representation error $Er(\mathcal{K}, \mathcal{S})$, the selected k skyline points should come from different k clusters.*

Proof This lemma can be obtained directly from Theorem 1.

According to Lemma 1, the selected points come from different clusters. As shown in Fig. 1, the full skyline is divided into 3 clusters. The selected 3 skyline points should come from different clusters. For each cluster $c_i, i \in [1, 3]$, we choose one point.

Next, we introduce how to select a point from a cluster. According to the objective function $Er(\mathcal{K}, \mathcal{S})$, the selected point p_i from c_i should have the minimum value $Er(p_i, c_i) = \max_{p_i \in c_i, p' \in c_i - \{p_i\}} \{\| p_i, p' \|\}$. The details to calculate the k-DRS based on ELM is described in Algorithm 2.

The calculation in a cluster needs to compute the distances between any two points in a cluster. Hence, the time cost of calculating an appropriate in a cluster is $O(|c_i|^2)$ where $|c_i|$ is the size of the largest cluster.

Algorithm 2: The DRSELM Algorithm

```
input  : the data set $D$ in $d$-dimensional space; the parameter $k$
output: The distance-base representative set $k$-$DRS(D)$;
1 Using BNL algorithm [1] to calculate the skyline $\mathcal{S}$ of $D$;
2 Using Algorithm 1 to divide the full skyline $\mathcal{S}$ into $k$ clusters;
3 for each cluster $c_i$ do
4     From all the points in $c_i$, add the one with the minimum value $MaxDis(p, c_i)$ to the
      $k$-$DRS$;
5 return $k$-$DRS$;
```

4 Experimental Evaluation

In this section, we demonstrate the efficiency and effectiveness of the DRSELM. We test 3 algorithms: 2d-opt, I-greedy, DRSELM. Specifically, 2d-opt and I-greedy are the algorithms in [2] for 2-dimensional dataset and d-dimensional ($d \geq 3$) dataset, respectively.

DataSets. We apply the same datasets in [2], a synthetic dataset *Island* and a real dataset *NBA*. *Island* follows a cluster distribution along the anti-diagonal, which is shown in Fig. 3. There are 27868 points in the *Island*, and the skyline of the *Island* consists of 110 points. *NBA* is downloadable at http://www.databasebasketball.com. It includes 17265 5-dimensional points and skyline of *NBA* consists of 494 points.

The distance-base representative skyline of *Island* is shown in Fig. 4 when k varies. As shown in Fig. 5, our DRSELM shows outstanding performances. Comparing with 2d-opt, DRSELM has more efficiency and good accuracy. In Fig. 5a,

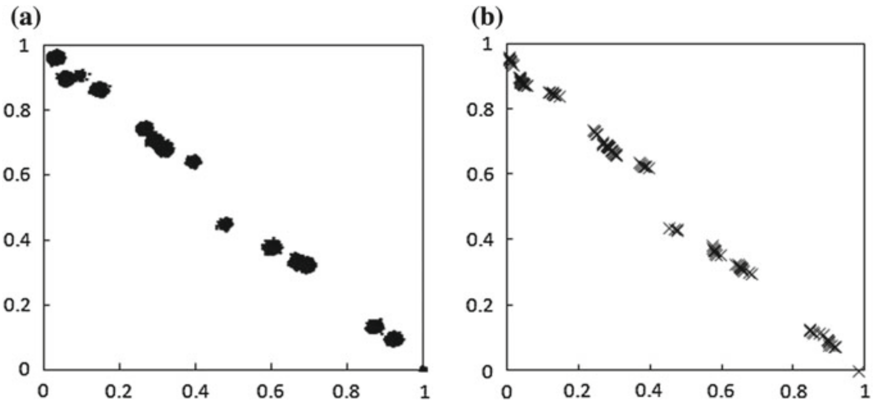


Fig. 3 The synthetic dataset *Island*. **a** The *Island* Dataset. **b** The Skyline of *Island*

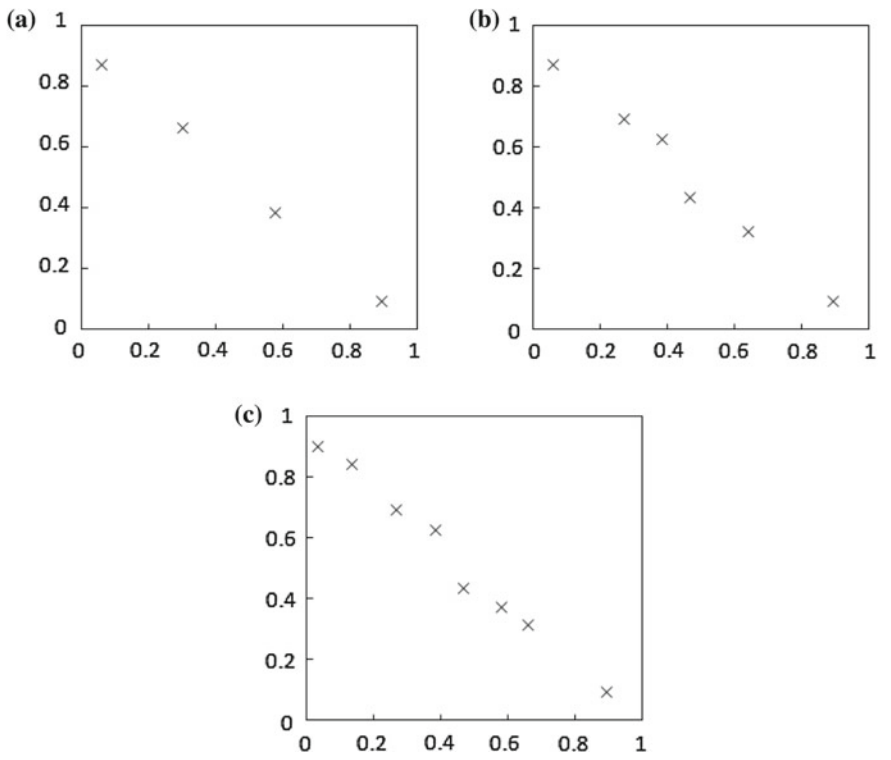


Fig. 4 The Distance-based Representative Skyline of *Island* for Different k. **a** $k = 4$. **b** $k = 6$. **c** $k = 8$.

with the increase of k, the running time of DRSELM and 2d-opt has little change. In Fig. 5b, as k grows, the representation error becomes smaller. DRSELM has the same representation errors with 2d-opt. Since 2d-opt is an exact algorithm, DRSELM has good accuracy in 2-dimensional datasets.

The experimental results of *NBA* is shown in Fig. 6. According to Fig. 6a, the running time of DRSELM is shorter than that of I-greedy. With the increase of k, the running time of DRSELM is stable, and the running time of I-greedy raises slightly. Hence, the efficiency of DRSELM is better than that of I-greedy. As illustrated in Fig. 6b, the representation errors of DRSELM and I-greedy are close. Therefore, comparing with I-greedy, the accuracy of DRSELM is competitive.

Comparing Fig. 5 with Fig. 6, with the increase of dimensionality, the running time of DRSELM has a little increment. Based on analysis above, it can be concluded that our DRSELM can process the k-DRS effectively.

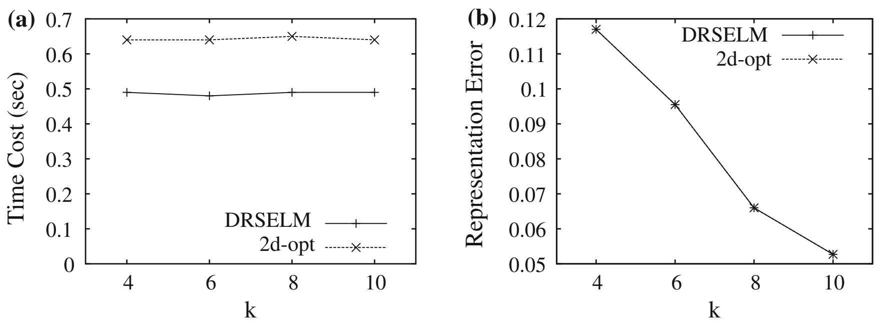


Fig. 5 The Experimental Results of *Island*. **a** Running time versus *k*. **b** Representation Error versus *k*

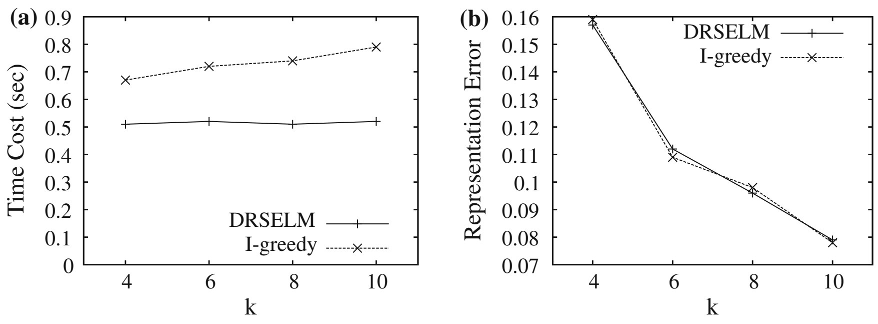


Fig. 6 The Experimental Results of *NBA*. **a** Running time versus *k*. **b** Representation Error versus *k*

5 Related Work

The skyline operator was first introduced by Börzönyi et al. [1]. Then many efficient skyline algorithms [20–23] have been proposed. Algorithms BNL and D&C [1], SFS [20], Bitmap [24] and NN [21] can process skyline query in the datasets without indexes. BBS [22] calculate the skyline query using R-tree index and ZINC [23] apply the Z-order index to process the skyline query. When the full skyline set is large, it is difficult to understand the full skyline. Thus, selecting *k* representative points is significant. There are some definitions [2, 25] about the representative skylines. In this paper, we focus on the distance-based representative skyline (*k*-DRS). *k*-DRS is NP-hard when $d \geq 3$. By in-depth analysis of the properties of the *k*-DRS, first, we use the clustering algorithms to cluster the full skyline set. Second, we calculate the representative point in each cluster. So far, there are some state-of-the-art clustering algorithms [15–18]. The experimental results show that US-ELM [14] is competitive in terms of both accuracy and efficiency. Hence, in this paper, we apply US-ELM to cluster the full skyline set.

6 Conclusion

As an important variant of skyline, the k representative skyline is a useful tool if the size of the full skyline is large. In this paper, we focus on the distance-based representative skyline (k-DRS). Since k-DRS is a NP-hard problem in d-dimensional ($d \geq 3$) space, we design a 2-step algorithm DRSELM to solve the k-DRS problem efficiently. Step 1 divides the full skyline set into k clusters using US-ELM algorithm. In step 2, a point in each cluster is selected as the representative skyline point. The k selected skyline points consist of the k-DRS. Comprehensive experimental results demonstrate that DRSELM is competitive with the state-of-the-art algorithm in terms of both accuracy and efficiency.

Acknowledgments This research was partially supported by the National Natural Science Foundation of China under Grant Nos. 61472069, 61402089, 61100022 and 61173029; the National High Technology Research and Development Plan (863 Plan) under Grant No. 2012AA011004; and the Fundamental Research Funds for the Central Universities under Grant No. N130404014.

References

1. Stephan, B.: Donald Kossmann, Konrad Stocker: The Skyline Operator. ICDE, pp. 421–430 (2001)
2. Tao, Y., Ding, L., Lin, X., Pei, J.: Distance-Based Representative Skyline. ICDE, pp. 892–903 (2009)
3. Huang, G., Zhu, Q., Siew, C.-K.: Extreme learning machine: a new learning scheme of feedforward neural networks. Proc. Int. Jt. Conf. Neural Netw. **2**, 985–990 (2004)
4. Huang, G., Zhu, Q., Siew, C.-K.: Extreme learning machine: theory and applications. Neurocomputing **70**, 489–501 (2006)
5. Huang, G.: What are extreme learning machines? Filling the gap between Frank Rosenblatt's Dream and John von Neumann's Puzzle. Cogn. Comput. **7**(3), 263–278 (2015)
6. Cherkassky, V.: The nature of statistical learning theory. IEEE Trans. Neural Netw. **8**(6), 1564 (1997)
7. Cortes, C., Vapnik, V.: Support-vector networks. Mach. Learn. **20**(3), 273–297 (1995)
8. Shi, L., Baoliang, L.: EEG-based vigilance estimation using extreme learning machines. Neurocomputing **102**, 135–143 (2013)
9. Huang, G., Zhou, H., Ding, X., Zhang, R.: Extreme learning machine for regression and multiclass classification. IEEE Trans. Syst. Man Cybern. **Part B 42**(2), 513–529 (2012)
10. Rong, H., Huang, G., Sundararajan, N., Saratchandran, P.: Online sequential fuzzy extreme learning machine for function approximation and classification problems. IEEE Trans. Syst. Man Cybern. **Part B 39**(4), 1067–1072 (2009)
11. Zhao, J., Wang, Z.: Dong Sun Park: online sequential extreme learning machine with forgetting mechanism. Neurocomputing **87**, 79–89 (2012)
12. Liang, N., Huang, G., Saratchandran, P., Sundararajan, N.: A fast and accurate online sequential learning algorithm for feedforward networks. IEEE Trans. Neural Netw. **17**(6), 1411–1423 (2006)
13. He, Q., Changying, D., Wang, Q., Zhuang, F., Shi, Z.: A parallel incremental extreme SVM classifier. Neurocomputing **74**(16), 2532–2540 (2011)
14. Huang, G., Song, S., Gupta, J.N.D., Wu, C.: Semi-Supervised and unsupervised extreme learning machines. IEEE Trans. Cybern. **44**(12), 2405–2417 (2014)

15. Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R., Wu, A.Y.: An efficient k-means clustering algorithm: analysis and implementation. IEEE Trans. Pattern Anal. Mach. Intell. **24**(7), 881–892 (2001)
16. Belkin, M., Niyogi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput. **15**(6), 1373–1396 (2003)
17. Ng, A.Y., Jordan, M.I., Weiss, Y.: On spectral clustering: analysis and an algorithm. Adv. Neural Inform. Process. Syst. **2**, 849C856 (2002)
18. Bengio, Y.: Learning deep architectures for AI. Found. Trends Mach. Learn. **2**(1), 1–127 (2009)
19. Serre, D.: TMatrices: Theory and Applications. Springer, Berlin (2002)
20. Chomicki, J., Godfrey, P., Gryz, J., Liang, D.: Skyline with presorting. ICDE 717–719 (2003)
21. Kossmann, D., Ramsak, F., Rost, S.: Shooting stars in the sky: an online algorithm for skyline queries. VLDB 275–286 (2002)
22. Papadias, D., Tao, Y., Fu, G., Seeger, B.: An optimal and progressive algorithm for skyline queries. SIGMOD Conference 467–478 (2003)
23. Liu, B., Chan, C.-Y.: ZINC: efficient indexing for skyline computation. PVLDB, **4**(3), 197–207 (2020)
24. Pei, J., Jin, W., Ester, M., Tao, Y.: Catching the best views of skyline: a semantic approach based on decisive subspaces. VLDB 253–264 (2005)
25. Lin, X., Yuan, Y., Zhang, Q., Zhang, Y.: Selecting stars: the *k* most representative skyline operator. ICDE 86–95 (2007)