Finding Diverse Neighbors in High Dimensional Space

Qi Guo #1, H. V. Jagadish *2, Anthony K. H. Tung #3, Yuxin Zheng #4§

National University of Singapore, Singapore
{1qiguo, 3atung, 4yuxin}@comp.nus.edu.sg

* Univ. of Michigan, Ann Arbor, MI, USA
2 jag@umich.edu

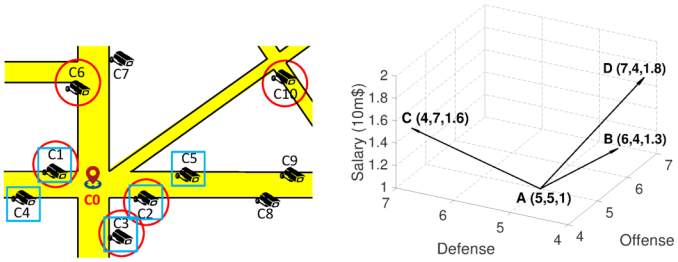
***Abstract*—Given a d-dimensional point query q, finding data items similar to q is a crucial task in many information retrieval and data mining applications. The typical approach is to find K items in a data set most similar to q, known as K nearest neighbors. Often, it is valuable to avoid too many answers that are too similar, and the importance of diversity has been considered in recent research. There are many different ways to characterize diversity, most of which depend on a notion of distance between points. In this paper, we propose a novel view of diversity based on spatial angles. This approach captures relevant and diverse results surrounding q from distinct directions even in high dimensional space. We present several algorithms to compute the diverse neighbor set, and show that it has several desirable properties. Extensive experiments demonstrate the effectiveness and efficiency of our methods on both real and synthetic data sets.**

I. Introduction

Similarity search has been studied extensively since it is a crucial task in many information retrieval and data mining applications. Usually, objects are represented as points in multi-dimensional space. Given queries in the form of points in this space, similarity search requires finding K nearest (most similar) neighbors to each query. Similarity between two points is often measured by a score function, such as cosine similarity or the inverse of Euclidean distance. Two objects are considered to be similar if their similarity score is high.

In addition to similarity, diversity has been demonstrated as a valid means to add value to the query results. Recent literature in information retrieval has extensively considered diversity when ranking text documents or web search results [1], [2], [3], [4], [5]. Search engines and recommender systems prefer relevant yet diverse results. In some scenarios, similarity search only may return redundantly similar results, while diversified results are more informative, especially when the queries are ambiguous. A keyword "Apple" may not only refer to the company, but also the fruit. The ambiguity problem is engendered because users' true intentions are unclear. Returning a collection of diverse results, which can cover various conjectures about the query, could be a reasonable strategy to resolve the problem. The importance of diversity in many scenarios is described in a recent survey [6].

Example 1.1: [7] *(Figure 1a) Suppose a criminal is spotted at location C0. The police have access to all cameras around C0, but have a limited number of screens (say $k = 5$) to display the view of different cameras simultaneously. Returning closest cameras (blue squared) may lead to information loss (no information from North and Northeast), while diverse results (red circled) can capture almost all surrounding environment.*



(a) Camera Access (b) NBA Player Trade

Figure 1: Introduction Example

In the above example, the criminal's whereabouts are unknown, but guessed roughly. Cameras well-spread in the region of interest can cover the region better than a cluster of cameras close to the center..

Example 1.2: (Figure 1b) Consider a simplified example of trading NBA players. Each player is represented by three attributes (offense, defense ability scores and salary). Trading player A for player B (direction from A to B) indicates the preference of paying a little more to improve offense but sacrificing defense. Player D is another choice, which will improve offense even more but paying more. On the other hand, trading player A for player C (direction from A to C) indicates an alternative preference of paying more to greatly improve defense but sacrificing offense. A large angle between $\vec{AD}$ and $\vec{AC}$ distinguishes two very different preferences.

In both examples, diversity in the sense of direction plays an important role. A well-spread result set to cover many directions can help users elicit their preferences. The popular distanced-based diversity [8], [9], [10], however, has no guarantees on the coverage of directions. Recall in Example 1.1, two distant points might lie in a similar direction (consider cameras C2 and C9). Furthermore, it was observed that distances lose their ability to distinguish points in high dimensional space due to the concentration phenomenon [11]. All distances between pairs of data points seem to be very

§Current affiliation: Tencent Inc.

similar. An intuitive way to measure differences between directions is to consider spatial angles. Moreover, angles are shown to be more stable than distances [12], [13]. In this paper, we propose a novel view of diversity, which is based on spatial angles. Given a query point q, we aim to find a set of close points of q that encircles q in different directions. We refer to points in this set as *angular diverse neighbors*.

To find angular diverse neighbors, we first define the angular dominance relation as follows: Given a query point q, we say a point p *angular dominates* another point p' if the following two conditions hold: (1) p is closer to q than p'; (2) the angle of $\angle pqp'$ is smaller than an angular threshold. Then, angular diverse neighbors are defined as all points that are not dominated by any other point.

Given an angular diversity threshold, we propose two algorithms to find angular diverse neighbors. The first is the Sorted-Scan algorithm which uses the property that a point can only be dominated by the query's nearer neighbors. To improve the efficiency, we propose the Two-Scan algorithm, which prepares a set of reference points beforehand and uses them to quickly prune true negatives during the first scan. As compensation, the Two-Scan algorithm needs a second scan of the data set to eliminate the false positives.

The angular threshold might be hard to set for users not familiar with a data set. One can also request for a diverse neighbor set of a specific size. To support this, we propose the Two-Stage algorithm to compute a near-optimal angular threshold and return a result set of the desired size.

We summarize the contributions of this paper as follows.

- We introduce a new concept based on spatial angles, called θ-dominance to investigate the result diversification problem in high dimensional space.
- We propose two different algorithms to compute the diverse neighbor set given the angular threshold.
- We provide an additional algorithm to compute the diverse neighbor set of the desired size without requiring specification of the angular threshold.
- We experimentally verify the effectiveness and efficiency of our proposed methods using various data sets.

II. Definition and Analysis

A. Problem Definition

Definition 2.1 (Point): Given a d-dimensional space $S = \{s_1, s_2, \ldots, s_d\}$, and a d-dimensional point p, we use $p.s_j$ to denote the j^{th} dimension value of point p.

Definition 2.2 (Distance): The distance between two points p and q is defined as the norm of the vector $\vec{qp}$, i.e. $dist(p,q) = \|\vec{qp}\| = \sqrt{\langle \vec{qp}, \vec{qp} \rangle}$, where $\langle \cdot, \cdot \rangle$ is an inner product.

Definition 2.3 (Angle): An angle $\angle pqp'$ represents the angle between vectors $\vec{qp}$ and $\vec{qp'}$, which can be computed as:

$$\angle pqp' = \arccos \frac{\left\langle \vec{qp}, \vec{qp'} \right\rangle}{\|\vec{qp}\| \cdot \|\vec{qp'}\|} \in [0, \pi], \tag{1}$$

Specifically, if q and p or p' coincide, $\angle pqp'$ is defined as π.

Table I: Table of Notations

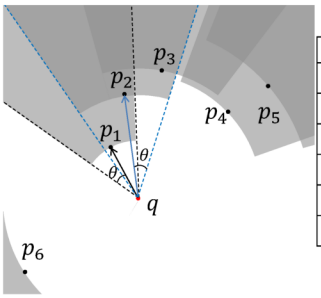
d	dimensionality
S	d-dimensional space $\{s_1, s_2, \ldots, s_d\}$
D	data set of d-dimensional points
$\lvert \cdot \rvert$	size of a set
$p.s_j$	j^{th} dimension value of point p
$dist(p,q)$	distance between points p and q
θ	angular threshold
$\vec{pq}$	directional vector from point q to point p
$\angle pqp'$	angle between vector $\vec{pq}$ and $\vec{p'q}$
$p \succ^S_{q,\theta} p'$	p θ-dominates p' w.r.t q
$p' \prec^S_{q,\theta} p$	p' is θ-dominated by p w.r.t q
$DS(D,q,\theta)$	angular θ-diverse neighbor set
$DS(D,q,k)$	sized k-diverse neighbor set
k	number of points in the diverse neighbor set
K	number of nearest neighbors
$\mathcal{N}(q,K,D)$	set of K-NNs of q in data set D
$\mathcal{T}(o,q,D)$	set of points nearer to q than o in data set D
$min\theta(o,q,D)$	$\min_{p \in \mathcal{T}(o,q,D)}(\angle oqp)$

Definition 2.4 (θ-dominate $\succ^S_{q,\theta}$): Given a query point q and an angular threshold θ, a point p θ-dominates p' w.r.t. q on space S, denoted as $p \succ^S_{q,\theta} p'$ if
(1) $dist(p,q) < dist(p',q)$ and
(2) $\angle pqp' < \theta$.
Please note that $p \succ^S_{q,\theta} p'$ is equivalent to $p' \prec^S_{q,\theta} p$.

Definition 2.5 (θ-dominating Area): Given a query point q and an angular threshold θ, the θ-dominating area w.r.t q on space S of a point p is the set of points: $\{x : x \in S \text{ and } p \succ^S_{q,\theta} x\}$. Note the θ-dominating area of one point increases faster when getting further to the query due to the sectorial shape.

Definition 2.6 (Diverse Point): A point p in D is a diverse point w.r.t q and θ on space S, if and only if there does not exist any point $p' \neq p$ in D such that $p \prec^S_{q,\theta} p'$.

Definition 2.7 (Diverse Neighbor Set $DS(D,q,\theta)$): Given a data set D, a query point q, a threshold θ, and the space S, the diverse neighbor set, denoted as $DS(D,q,\theta)$, is the set of all diverse points w.r.t q and θ on space S.



Point	coordinates	$dist(\cdot,q)$	$min\theta(\cdot,q,D)$
p_1	(-2, 4)	4.47	180°
p_2	(-1, 8)	8.06	19.44°
p_3	(2, 10)	10.20	18.43°
p_4	(6, 6)	8.49	52.13°
p_5	(10, 9)	13.46	3.01°
p_6	(-11, -6)	12.53	92.05°

Figure 2: An illustrative example, where θ = 20°

Example 2.1: Consider a data set $D = \{p_1, \ldots, p_6\}$, and a query point $q = (0,0)$ on $\mathbb{R}^2$ equipped with the dot product, shown in Figure 2. The diverse neighbor set $DS(D,q,\theta)$ contains three points: p_1, p_4, and p_6. The gray area represents the θ-dominating area of each point. Specifically, $p_2 \prec^{\mathbb{R}^2}_{q,\theta} p_1$, $p_3 \prec^{\mathbb{R}^2}_{q,\theta} p_2$, and $p_5 \prec^{\mathbb{R}^2}_{q,\theta} p_4$ It is clear that only p_1, p_4, p_6 are not contained in any gray area, which implies they are not θ-dominated by any points in D w.r.t q.

Based on the diverse neighbor set, we are particularly interested in solving the following two problems:

Problem 1 (angular θ-diverse neighbors $DS(D,q,\theta)$): Given a data set D, a query q and an angular threshold θ, compute $DS(D,q,\theta)$.

Though $DS(D,q,\theta)$ is useful on its own, the angular threshold θ might be hard to specify for those users who are not familiar with a data set. To address this, we propose Problem 2, in which users just need to specify the cardinality of the result set.

Problem 2 (sized k-diverse neighbors $DS(D,q,k)$): Given a data set D, a query point q, and a specified integer k, find a θ^*, such that $|DS(D,q,\theta^*)| = k$, and output $DS(D,q,\theta^*)$ as $DS(D,q,k)$.

For ease of presentation, we summarize the important notations in Table I.

B. Analysis

Given a query point q, for any two points a, b, the angle $\angle aqb$ provides a way to measure the difference between a and b. Firstly, the Euclidean distance between the two points has a lower bound. The lower bound is decided by the farther point and the angle. Moreover, when the angle is large, the two points must differ a lot in at least one dimension. These ideas are formally established next.

Theorem 2.1: Given three points a, b and q on $\mathbb{R}^n$ with the dot product, where $\angle aqb = \theta$ and $\|\vec{qa}\| = l_a < \|\vec{qb}\| = l_b$, then:

1) $dist(a,b) \geq l_b \cdot \sin\theta$,
2) there exists a dimension i such that $|a.s_i - b.s_i| \geq \sqrt{2/d} \cdot \min(|l_a - l_b \cdot \cos\theta|, l_b \cdot \sin\theta)$.

Proof: (1) Since points q, a, b form a triangle in the d-dimensional space, by the law of cosines,

$$\begin{aligned} dist^2(a,b) &= l_a^2 + l_b^2 - 2l_a l_b \cdot \cos\theta \\ &= (l_a - l_b \cdot \cos\theta)^2 + l_b^2 \cdot (1 - \cos^2\theta) \\ &\geq l_b^2 \cdot \sin^2\theta. \end{aligned} \quad (2)$$

Thus, $dist(a,b) \geq l_b \cdot \sin\theta$.

(2) We prove by contradiction. Suppose for all $i = 1, 2, \ldots, d$,

$$|a.s_i - b.s_i| < \sqrt{2/d} \cdot \min(|l_a - l_b \cdot \cos\theta|, l_b \cdot \sin\theta). \quad (3)$$

$$\begin{aligned} \text{Then, } 2\|\vec{ab}\|^2 &= 2\sum_{i=1}^{d}(a.s_i - b.s_i)^2 \\ &< \sum_{i=1}^{d}(\sqrt{2/d} \cdot |l_a - l_b \cdot \cos\theta|)^2 + \sum_{i=1}^{d}(\sqrt{2/d} \cdot l_b \cdot \sin\theta)^2 \\ &= 2(l_a^2 - 2l_a l_b \cdot \cos\theta + l_b^2 \cdot \cos^2\theta + l_b^2 \cdot \sin^2\theta). \end{aligned} \quad (4)$$

Hence, $\|\vec{ab}\|^2 < l_a^2 + l_b^2 - 2l_a l_b \cdot \cos\theta$, which contradicts the law of cosines. Therefore, there exists i such that $|a.s_i - b.s_i| \geq \sqrt{2/d} \cdot \min(|l_a - l_b \cdot \cos\theta|, l_b \cdot \sin\theta)$. ■

Next, we show three properties of the proposed diverse neighbor set, i.e. **Monotonicity**, **Existence** and **Uniqueness**.

Monotonicity: the size of the diverse neighbor set decreases monotonically with the increase of θ.

Lemma 2.1: Given $\theta' \geq \theta$, if $p \succ^S_{q,\theta} p'$, then $p \succ^S_{q,\theta'} p'$.

Proof: By Definition 2.4, we have $dist(p,q) < dist(p',q)$ and $\angle pqp' < \theta$. Thus, clearly, we have $\angle pqp' < \theta \leq \theta'$, again by Definition 2.4, $p \succ^S_{q,\theta'} p'$. ■

Theorem 2.2: Given $\theta' \geq \theta$, $DS(D,q,\theta') \subseteq DS(D,q,\theta)$.

Proof: (By contradiction) Consider a point $p \in DS(D,q,\theta')$. Suppose $p \notin DS(D,q,\theta)$, then $\exists o \in D$, s.t. $o \succ^S_{q,\theta} p$. By Lemma 2.1, $o \succ^S_{q,\theta'} p$. This contradicts the assumption. Thus, $p \in DS(D,q,\theta)$, and we get $DS(D,q,\theta') \subseteq DS(D,q,\theta)$ ■

From the last theorem, we can conclude that given the same data set and the same query, the size of the diverse neighbor set decreases monotonically with the increase of θ. Thus, for different purposes, users can control the size of the diverse neighbor set by choosing their own θ.

Existence: There always exists an answer to Problem 1.

Lemma 2.2: Given a data set D, a query point q, $DS(D,q,\theta)$ is non-empty for any θ.

The above lemma is obvious, since $DS(D,q,\theta)$ will always contain the first nearest neighbor of q among D. Besides, any point satisfying Definition 2.6 will be included in the result set. The existence of answer to Problem 2 will be discussed in Section IV-A.

Uniqueness: Furthermore, given specific parameters, answer to Problem 1 is unique.

Theorem 2.3: Given a data set D, a query point q, $DS(D,q,\theta)$ is unique for a specific θ.

Proof: Suppose there exist two answer sets $DS_1(D,q,\theta)$ and $DS_2(D,q,\theta)$. Consider a point $p \in DS_1(D,q,\theta)$, by Definition 2.6, there does not exist any point $p' \neq p$ such that $p \prec^S_{q,\theta} p'$. Then, by Definition 2.7, $p \in DS_2(D,q,\theta)$. Thus, $DS_1(D,q,\theta) \subseteq DS_2(D,q,\theta)$. Similarly, $DS_2(D,q,\theta) \subseteq DS_1(D,q,\theta)$. Hence, $DS_1(D,q,\theta) = DS_2(D,q,\theta)$. ■

Moreover, under certain assumptions, answer to Problem 2 is also unique, and details will be shown in Section IV-A.

III. Angular θ-Diverse Set

In this section, we present two algorithms, namely the *Sorted-Scan Algorithm*, and the *Two-Scan Algorithm*, to compute the angular θ-diverse neighbor set $DS(D,q,\theta)$.

A. Sorted-Scan Algorithm

The first approach to compute angular θ-diverse neighbor set $DS(D,q,\theta)$ is to examine each point one by one. Given a point o, we define the set of points that have smaller distance to the query compared to o:

Definition 3.1 (Closer points $\mathcal{T}(o,q,D)$): Given a data set D, a point $o \in D$ and a query point q, we use $\mathcal{T}(o,q,D)$ to denote $\{p : p \in D \wedge dist(p,q) < dist(o,q)\}$.

Fact 3.1: According to Definition 2.4, a data point o can only be dominated by another point in $\mathcal{T}(o,q,D)$.

Algorithm 1 shows the details of the Sorted-Scan algorithm, and its correctness is based on the above fact. Each point p is only compared to q's nearer neighbors (steps 5 to 8). If there exists a point p' such that p' θ-dominates p w.r.t q, then p is surely not a diverse point, and we can jump to examine the

Algorithm 1: Sorted-Scan Algorithm $DS(D, q, \theta)$

```
1  sort D in non-descending order of distance to q ;
2  initialize result set T ← ∅ ;
3  for every point p ∈ D do
4      initialize isDiverse ← true ;
5      for every point p' ∈ T(p, q, D) do
6          if ∠pqp' < θ then
7              isDiverse ← false ;
8              break out of inner for-loop ;
9      if isDiverse then
10         insert p into T ;
11 return T ;
```

Algorithm 2: Two-Scan Algorithm $DS(D, q, \theta)$

```
1  initialize result set T ← ∅ ;
2  choose a set of reference points R from D ;
3  for every point p ∈ D do
4      if p is not θ-dominated by any point in R then
5          insert p into T ;
6  for every point p' ∈ D do
7      for every point p ∈ T, p ≠ p' do
8          if p ≺^S_{q,θ} p' then
9              remove p from T ;
10 return T ;
```

next point. Otherwise, if none of the nearer neighbors can θ-dominate p, then p can be safely added to T (steps 9 to 10). Once all the points in D are checked, T is outputted as the final answer, i.e. $DS(D, q, \theta)$. In general, the complexity of the Sorted-Scan algorithm should be $O(n\log(n)) + O(1+2+\ldots+n) = O(n^2)$ in worst cases.

Example 3.1: Consider the example in Figure 2. The Sorted-Scan algorithm will first sort points as $\{p_1, p_2, p_4, p_3, p_6, p_5\}$ based on their Euclidean distance to q. Next, the algorithm checks whether a point p is θ-dominated by another point in $\mathcal{T}(p, q, D)$. As a result, the algorithm progressively adds p_1, p_4, p_6 into the result set. Finally, it returns $DS(D, q, \theta) = \{p_1, p_4, p_6\}$ after checking all points.

B. Two-Scan Algorithm

Although the Sorted-Scan algorithm is workable, its efficiency is unsatisfactory: in the worst case, the total number of comparisons is $O(n^2)$. Next, we present our second algorithm (shown in Algorithm 2) which improves the efficiency.

This algorithm can be naturally divided into two parts. In the first scan of D (steps 3 to 5), we quickly eliminate true negatives with the help of a reference set R, which is chosen beforehand. R here is static, and the comparisons against R is one-way. Thus, we can progressively compute a set of candidate diverse points by comparing each point $p \in D$ to R. A point p can become a candidate if and only if none of the reference points in R θ-dominates p. T will contain all the candidates after the first scan of D.

Due to the existence of false positives in T, it is necessary to scan D once more (steps 6 to 9). If a candidate is a false positive, it must be θ-dominated by some points in D. Therefore, after the second scan, T is guaranteed to be the diverse neighbor set.

Example 3.2: If R is initialized as $\{p_1, p_2, p_4\}$, which are 3-NNs of q, then after the first scan, points p_2, p_3, p_5 will be eliminated, since $p_2 \prec^{\mathbb{R}^2}_{q,\theta} p_1$, $p_3 \prec^{\mathbb{R}^2}_{q,\theta} p_2$, and $p_5 \prec^{\mathbb{R}^2}_{q,\theta} p_4$. Hence, T contains p_1, p_4, p_6. Next, no false negative is found in the second scan.

It is clear that the efficiency of the Two-Scan Algorithm is highly dependent on the choice of R, the set of reference points. If the total dominating areas of points in R can cover most of D, then less candidates need to be checked in the second scan. However, if R contains too many points, the first scan will slow down. Here we provide three possible choices of R. One natural choice can be the K-nearest neighbors of the query among D. K-NNs have their own advantages since they are nearer to the query than any other points. Secondly, a simple choice can be K random points selected from D. By randomly selecting, it is likely to eliminate bias. Thirdly, points can be allocated into K different sections based on their distances to the query, and R can select one from each section randomly. In this way, R is expected to capture the "distribution nature" of D. Detailed comparisons of the proposed three choices of R are presented in Section V. Although the complexity of the Two-Scan Algorithm is $O(|R|n + n^2)$ in the worst case, it can perform much better in practice with proper choice of R.

C. Acceleration Using GPU

From Fact 3.1, we can conclude that to determine whether a point diverse or not is independent of the determination process of any other point. This means we can parallel scan the data set instead of checking each point sequentially.

In order to effectively parallelize the computations of the diverse neighbor set, we choose to implement our algorithms on the Graphics Processing Units (GPUs). It has become more popular to use GPUs for data processing in recent years, since GPUs have experienced a tremendous growth in terms of computational power and memory capacity.

We adopt the most direct strategy to accelerate the computation. For Algorithm 1, firstly the computation of all distances can be fully parallelized. Then, we use the *thrust* library to sort all points based on their distances to the query. Next, we use one block of threads to undertake the task of justifying one point from D. Each thread takes charge of comparing current point against one of those nearer neighbors. For each block, *isDiverse* becomes a shared-memory variable so that the whole block can stop in time when one thread finds a dominating relationship.

For Algorithm 2, the parallelization methodology is similar. In the first scan, each block of threads only need to compare current point against all points in the reference set. In the second scan, each block of threads further justify a candidate by comparing it against all points in the data set.

The numbers of blocks and threads are adjusted, so that the computation power is utilized. The strategy we adopt is

not the only but the most direct approach to parallelize the algorithms. Since it is not the major problem of this paper, we do not conduct further discussion.

IV. Size k-Diverse Neighbor Set

It has been shown by Theorem 2.2 that the size of the diverse neighbor set decreases with the increase of θ. How to get the diverse neighbor set of a desired size by choosing the right θ becomes a new problem. To address the $DS(D, q, k)$ problem, we first introduce a naive algorithm and then propose our *Two-Stage Algorithm*.

A. A Naive Algorithm

Definition 4.1 (Minimum dominated angle, $min\theta(o, q, D)$): Given a data set D, a point $o \in D$ and a query point q, we use $min\theta(o, q, D)$ to denote the minimum dominated angle of o, namely, $\min_{p \in \mathcal{T}(o,q,D)} \angle oqp$. Specially, if o is q's nearest neighbor, we set $min\theta(o, q, D) = \pi$.

$$min\theta(o, q, D) = \begin{cases} \min_{p \in \mathcal{T}(o,q,D)} \angle oqp & \text{if } \mathcal{T}(o, q, D) \neq \emptyset \\ \pi & \text{if } \mathcal{T}(o, q, D) = \emptyset \end{cases}$$

Based on the definition, we can derive the following lemma.

Lemma 4.1: If $o \in DS(D, q, \theta)$, then $min\theta(o, q, D) \geq \theta$.

Example 4.1: Consider the example in Figure 2, we have $min\theta(p_1, q, D) = \pi$, $min\theta(p_2, q, D) = \angle p_1qp_2 = 19.44°$, $min\theta(p_3, q, D) = \angle p_2qp_3 = 18.43°$, $min\theta(p_4, q, D) = \angle p_2qp_4 = 52.13°$, $min\theta(p_5, q, D) = \angle p_4qp_5 = 3.01°$, *and* $min\theta(p_6, q, D) = \angle p_1qp_6 = 92.05°$.

Clearly, $min\theta(p_5, q, D) < min\theta(p_3, q, D) < min\theta(p_2, q, D) < \theta = 20° < min\theta(p_4, q, D) < min\theta(p_6, q, D) < min\theta(p_1, q, D)$.

Next, we illustrate a naive solution to compute $DS(D, q, k)$. Suppose points in the diverse neighbor set are sorted in descending order of the minimum dominated angles, i.e. $DS(D, q, \theta) = \{o_1, \ldots, o_n\}$, where $min\theta(o_1, q, D) > \ldots > min\theta(o_n, q, D) \geq \theta$. In this way, the size of the diverse neighbor set can be reduced by simply increasing θ to a specific value. Any $\theta' \in (min\theta(o_{k+1}, q, D), min\theta(o_k, q, D)]$ should be a feasible solution.

Therefore, one naive way to compute $DS(D, q, k)$ is to start with a small angular threshold θ_{small} such that $|DS(D, q, \theta_{small})| > k$ and keep track of the minimum dominated angles for all points in the diverse neighbor set when computing $DS(D, q, \theta_{small})$. Once we finish computing $DS(D, q, \theta_{small})$, we can immediately retrieve a diverse neighbor set of a desired size k by increasing θ_{small}.

Ideally, if the minimum dominated angles of all points are different, then the returned diverse neighbor set can have any size by choosing proper angle threshold. Under such assumption, it can be proved that the answer to Problem 2 exists and is unique for any specific k. The existence can be directly proved using the Naive algorithm.

Theorem 4.1: Given a data set D, a query point q, $DS(D, q, k)$ is unique for a specific k if all points have different minimum dominated angles.

Proof: (By contradiction) Suppose we have two answer sets $DS(D, q, \theta_1^*)$ and $DS(D, q, \theta_2^*)$ such that $|DS(D, q, \theta_1^*)| = |DS(D, q, \theta_2^*)| = k$ and $DS(D, q, \theta_1^*) \neq DS(D, q, \theta_2^*)$. W.L.O.G assume $\theta_1^* \leq \theta_2^*$. Since the two sets are not same, they must differ at least one point. Assume $p \in DS(D, q, \theta_2^*)$ but $p \notin DS(D, q, \theta_1^*)$. Then, there must exist a point o such that $o \succ^S_{q,\theta_1^*} p$. By Lemma 2.1, $o \succ^S_{q,\theta_2^*} p$, which contradicts $p \in DS(D, q, \theta_2^*)$. Hence, the answer is unique. ■

However, there may exist cases such that two or more points share minimum dominated angles with the same value, just like points can have same distance to the query when performing K-NN search. In these cases, we can either expand result sets to include all these points or break the tie randomly. However, by bringing in randomness, the answers will then become non-unique.

B. Two-Stage Algorithm

Although the naive approach can solve the problem, its efficiency cannot be guaranteed. As mentioned before, if the starting angle is too small, the returned diverse neighbor set will have a lot of points, and more importantly, the computation cost will increase dramatically (for both comparing and maintaining the order). Thus, given a specific integer k, an ideal starting angle θ_{start} should be close to θ^* as much as possible, where $DS(D, q, \theta^*) = DS(D, q, k)$. Then, the question becomes how to find such a θ^*. Next we present how to approximate such a θ^*.

Definition 4.2 (K-NNs, $\mathcal{N}(q, K, D)$): We use $\mathcal{N}(q, K, D)$ to denote the set of K nearest neighbors of q among D.

Theorem 4.2: If $K_1 < K_2$ and $|DS(\mathcal{N}(q, K_1, D), q, \theta_{L_{K_1}})| = |DS(\mathcal{N}(q, K_2, D), q, \theta_{L_{K_2}})|$, then $\theta_{L_{K_1}} \leq \theta_{L_{K_2}}$.

Proof: Consider a point $p \in DS(\mathcal{N}(q, K_1, D), q, \theta)$. Firstly, p cannot be θ-dominated by any point in $\mathcal{N}(q, K_2, D) \setminus \mathcal{N}(q, K_1, D)$ (Fact 3.1). Secondly no point in $\mathcal{N}(q, K_1, D)$ can θ-dominate p w.r.t q (otherwise p can not be in $DS(\mathcal{N}(q, K_1, D), q, \theta)$. Hence, p must be included in $DS(\mathcal{N}(q, K_2, D), q, \theta)$, which implies

$$DS(\mathcal{N}(q, K_1, D), q, \theta) \subseteq DS(\mathcal{N}(q, K_2, D), q, \theta). \quad (5)$$

Therefore,

$$\begin{aligned} |DS(\mathcal{N}(q, K_1, D), q, \theta_{L_{K_2}})| &\leq |DS(\mathcal{N}(q, K_2, D), q, \theta_{L_{K_2}})| \\ &= |DS(\mathcal{N}(q, K_1, D), q, \theta_{L_{K_1}})| \end{aligned}$$

By Theorem 2.2, we conclude $\theta_{L_{K_1}} \leq \theta_{L_{K_2}}$. ■

Corollary 4.1: If $|DS(\mathcal{N}(q, K, D), q, \theta_{L_K})| = |DS(D, q, \theta)|$, then $\theta_{L_K} \leq \theta$.

In summary, the above theorem and corollary show that:

1) θ_{L_K} can be used as a reliable lower bound for θ^*.
2) the larger K is, the tighter the angle lower bound is.

Consider a sequence of values, where $K_i < K_j$ if $i < j$. If for all i, $|DS(\mathcal{N}(q, K_i, D), q, \theta_{L_{K_i}})| = |DS(D, q, \theta^*)| = k$, then $\theta_{L_{K_i}} \leq \theta_{L_{K_j}} \leq \theta^*$ where $i < j$. In particular, when $K = |D|$, $DS(D, q, \theta_{L_K}) = DS(D, q, \theta^*) = DS(D, q, k)$.

To compute $DS(D, q, k)$, we propose Algorithm 3, which can be divided into two stages: (1) compute the lower-bound

Algorithm 3: Two-Stage Algorithm (D, q, k, K)

1 initialize result set $T \leftarrow \emptyset$;
2 retrieve K-NNs of q among D to get $\mathcal{N}(q, K, D)$;
3 compute $DS(\mathcal{N}(q, K, D), q, 0)$ and record th k-th largest minimum dominated angle $min\theta(o_k, q, D)$;
4 input D,q and $min\theta(o_k, q, D)$ into the Sorted-Scan Algorithm or the Two-Scan Algorithm ;
5 select the first k points from $DS(D, q, min\theta(o_k, q, D))$, add into T ;
6 **return** T ;

(steps 1-3); and (2) compute the diverse neighbor set (steps 4-6). Firstly, the algorithm retrieves $\mathcal{N}(q, K, D)$ from D. Then, the algorithm inputs $\mathcal{N}(q, K, D)$, q, and an angular threshold with value zero into Algorithm 1 or Algorithm 2. The reason why we set the angular threshold to be zero is that we can maintain the minimum dominated angle for every point in $\mathcal{N}(q, K, D)$. In this way, we can obtain the k-th largest minimum dominated angle $min\theta(o_k, q, D)$, which is proved to be a reliable lower bound in Corollary 4.1. As mentioned before, the larger the value K takes, the tighter the angle lower bound is. However, too large a K will slow down the computation of $DS(\mathcal{N}(q, K, D), q, 0)$. We will discuss the choice of K in the next section.

Next, we input the $min\theta(o_k, q, D)$ we got from the last step as well as D and q into the Sorted-Scan Algorithm or the Two-Scan Algorithm. Once $DS(D, q, min\theta(o_k, q, D))$ is computed, we can retrieve points with the top-k largest minimum dominated angles as the final result.

Example 4.2: Consider finding a diverse neighbor set containing 3 points from Figure 2. With $K = k = 3$, points p_1, p_2, p_4 are retrieved as $\mathcal{N}(q, 3, D)$. Next, their minimum dominated angles are computed and sorted in descending order: $min\theta(p_1, q, D) > min\theta(p_4, q, D) > min\theta(p_2, q, D)$. The third largest angle $min\theta(p_2, q, D)$ is then used as the lower bound, and $DS(D, q, min\theta(p_2, q, D))$ is computed, which contains p_1, p_6, p_4, and p_2. Again, they are sorted in descending order of the minimum dominated angles. At last, the first three points p_1, p_6, p_4 can be returned.

V. Experiments

In this section, we study the following two issues: (1) The effectiveness of the proposed angular diverse neighbors compared to the state-of-the-art methods of result diversification, and (2) The efficiency of the proposed methods.

A. Experimental settings

1) Data Sets: In the experiments, both synthetic data sets and real data sets are used. We focus on high dimensional data sets. The properties of all the data sets are summarized in Table II and III respectively, where d represents the dimensionality and $Card.$ denotes the cardinality of the data set.

MovieLens[14]: This data set describes free-text tagging activity from MovieLens, a movie recommendation service. It contains 465,564 tag applications across 27,278 movies. We group all tags by each movie, i.e. each movie is described by a set of tags. We further build a LDA topic model. The model generates 100 topics, and within each there are 20 keywords. Afterwards, each movie is inferred as a 100-dimensional point by the trained model. The value of each dimension indicates the probability of a movie belonging to corresponding topic. Queries are firstly constructed using a limited number of keywords, so that they are ambiguous. We then pass queries to the trained LDA model to obtain the corresponding 100-dimensional points.

Wine:[15]: Wine is a data set recording physicochemical properties of red and white wines (e.g. residual sugar and pH values). There are 11 attributes, and 5,318 unique records. We randomly pick 500 records to serve as queries.

Motor[16]: Motor is a data set containing 58,509 records, each of which consists of 48 features extracted from electric current drive signal. The current signals are measured with a current probe and an oscilloscope on two phases. 500 queries are randomly picked from the data set.

Galaxy: This data set is generated from the Sloan Digital Sky Survey (SDSS) with data release 14 [1]. We sample the "PhotoObjAll" table with sampling ratio 1%. Sixteen numerical columns are chosen, such as the galactic latitude and longitude. At last, this data set contains 1.9 million tuples with dimensionality 16. We further sample 500 tuples from the base table extracting same columns as queries.

Sun[17]: Sun is a data set containing 88,903 images, each of which is attached with a class label to indicate which scene category the image belongs to. The GIST feature [18] is obtained. SUN contains a test set of 19,850 points, and we randomly choose 500 points as queries.

Synthetic data sets: Synthetic data sets are used to study the influence of the dimensionality and the cardinality. We consider three kinds of data distribution. In details, points are generated with uniform distribution ($[0, 1]$), normal distribution ($\mu = 0, \sigma^2 = 1$) or skewed normal distribution ($\mu = 0, \sigma^2 = 1, \alpha = 1$) respectively. For each type of distribution, we generate data sets by fixing a property as default (underlined) and varying the other. For each parameter setting, we randomly generate 50 points as queries.

Table II: Real data sets

Data set	d	$Card.$
MovieLens	100	27k
Wine	11	5.3k
Motor	48	58k
Galaxy	16	1.9m
Sun	512	89k

Table III: Synthetic data sets

Properties	Values
$Card.$	100k, 200k, <u>400k</u>, 800k, 1.6m
d	50, 100, <u>200</u>, 400, 800
Distribution	Uniform, Normal, Skewed

2) Evaluation Measures: We first adopt relevance and vector diversity, which are existing evaluation measures [7].

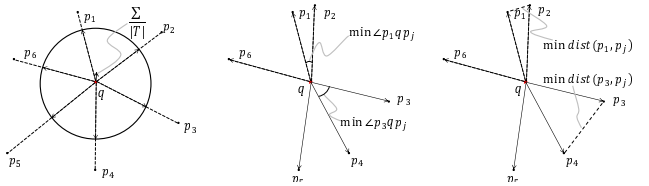
Definition 5.1 (Relevance): Given a query point q, its k-nearest neighbors $\mathcal{N}(q, k, D)$, and a set of results T, where $|T| = k$, relevance is defined as:

$$Rel(q, \mathcal{N}(q, k, D), T) = \frac{\sum_{p_j \in \mathcal{N}(q,k,D)} \|q\vec{p}_j\|}{\sum_{p_i \in T} \|q\vec{p}_i\|} \in [0, 1]. \quad (6)$$

[1] http://www.sdss.org/dr14/

Definition 5.2 (Vector Diversity): Given a query point q and a set of results T, vector diversity is defined as:

$$VDiv(q,T) = 1 - \frac{\|\sum_{p_i \in T} \frac{q\vec{p}_i}{\|q\vec{p}_i\|}\|}{|T|} \in [0,1]. \quad (7)$$



(a) VDiv (b) AvgADiv (c) AvgDDiv

Figure 3: Diversity measures

The intuition is that if the result set is diverse, the sum of the "forces" will be closer to the center [7], so that $VDiv$ will be close to 1. However, Figure 3b shows an example such that a good vector diversity could be insufficient. Although the value of the vector diversity is nearly 1, there exist pairs such as p_1, p_2 or p_4, p_5, which are not diverse enough. Thus, we introduce the average angular diversity as below:

Definition 5.3 (Average Angle Diversity): Given a query point q and a set of results T, the average angular diversity is:

$$AvgADiv(q,T) = \frac{\sum_{p_i \in T} \min_{p_j \in T, j \neq i} \angle p_i q p_j}{|T|}. \quad (8)$$

Each point in the result set has a most "similar" neighbor. The similarity can be measured by the angle, whose endpoint is the query. Intuitively, if $AvgADiv$ is bigger, points of the result set spread out more in the space centering at the query. One can also use classical distance metric instead of angle to measure the diversity:

Definition 5.4 (Average Distance Diversity): Given a query point q and a set of results T, the average distance diversity is computed as:

$$AvgDDiv(q,T) = \frac{\sum_{p_i \in T} \min_{p_j \in T, j \neq i} dist(p_i, p_j)}{|T|}. \quad (9)$$

By considering $VDiv$, $AvgADiv$ and $AvgDDiv$, we are more clear about how the result points are distributed around the query, and how diverse they are.

3) Competitors: We choose four competitors from various approaches to result diversification problem in the literature.

KNDN-IG (Immediate Greedy) and **KNDN-BG** (Buffered Greedy) [9]. The KNDN methods are also known as Motely. In our experiments, instead of using R-tree, points are sorted directly based on their Euclidean distance to the query, which allows the KNDN methods to handle high dimensional data. The threshold parameter *MinDiv* is set to 0.1 as suggested.

GG (Gabriel Graph based method) [7]. Since generating Gabriel graph for high dimensional data set efficiently is challenging, we extract only the necessary Gabriel-edges as the authors mentioned.

MMD (Maximize Minimum pairwise Distance) [10]. To obtain a sized-k result set, we provide $5k$-NNs to MMD for a post-processing step to diversify results.

All the experiments were conducted on a CPU-GPU platform with NVIDIA GeForce GTX TITAN X (with 12 GB memory) GPU, Intel Core i7-3820 CPU, and 64GB main memory, running CentOS 6.5. GPU is used to accelerate the computation of distances and angles. All GPU codes were implemented with CUDA 7. Other programs and competitors were implemented in C++.

B. Effectiveness

To validate the statement that our angular diverse neighbors provide more reasonable results compared to the distance-based diversification methods, we first conduct a user study. Then, we study the effectiveness of our proposed angular diverse neighbors compared to the competitors in terms of Rel (relevance), VDiv (vector diversity), AvgADiv (average angle diversity) and AvgDDiv (average distance diversity) on other real data sets.

1) User Study: In this study, we use MovieLens data set, and there are 15 queries in total. We hire 20 students with different backgrounds as the participants. For each query consisting of some keywords, we generate six sets (each containing 10 movies) of results using different methods, and each participant is required to select the one which is most relevant and diverse in his/her opinion. Table IV shows the selection percentage. We also present the results of two sample queries ("superhero" and "space and alien") in Figure 4 and Figure 5. Note the results of the KNDN methods are omitted here, since they are nearly same as K-NN results. Possible reasons are discussed in following experiments.

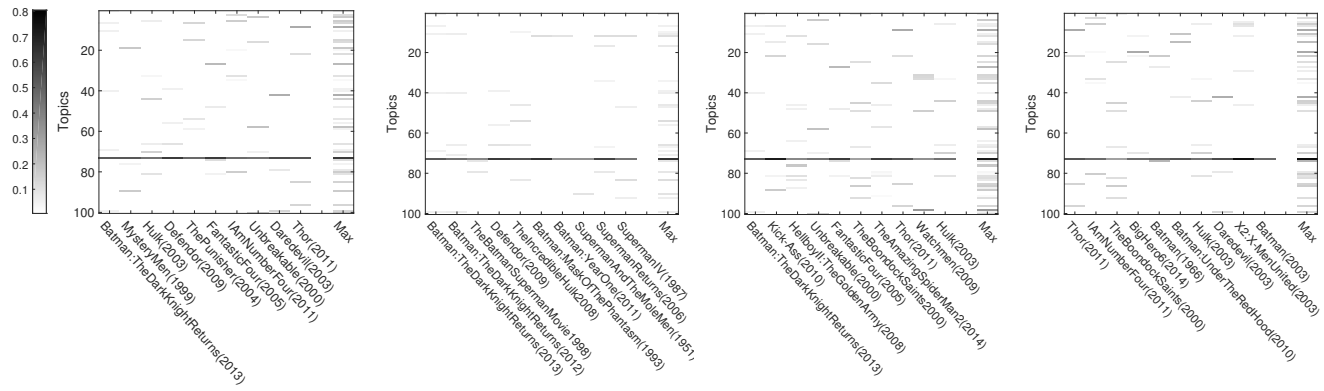
Table IV: User study results

DS	GG	MMD	K-NN, KNDN-IG & KNDN-BG
57.58%	24.24%	18.18%	0%

Figure 4 and Figure 5 demonstrate the effectiveness of our proposed diverse neighbors. The results are suggested to be viewed in color mode. Our method consistently returns results with diversely distributed topics.

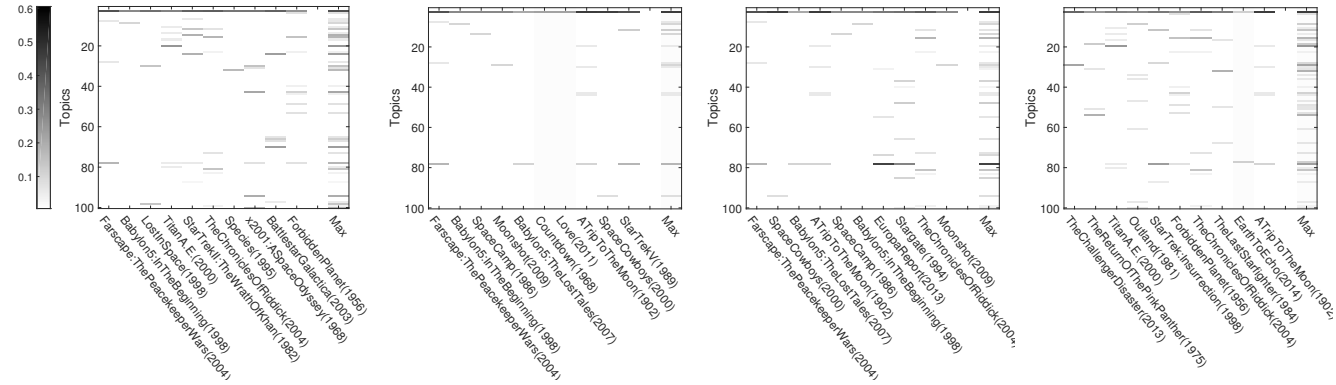
2) Result Quality: Next, we study the effectiveness in terms of Rel (relevance), VDiv (vector diversity), AvgADiv (average angle diversity), and AvgDDiv (average distance diversity) on the four real data sets. KNDN-IG and KNDN-BG require *MinDiv* to be set. To ensure that enough results are returned, we add a function to dynamically adjust *MinDiv*. For GG, firstly it is very time-consuming to verify Gabriel-edges for large-size data sets. Moreover, due to the sparsity of high dimensional space, optimizing the last layer becomes very slow. Thus, we do not apply GG on Galaxy and Sun due to its own limitations.

Relevance: Figure 6 shows how the relevance of result set changes when the size of result set varies. The horizontal axis represents the size of result set. For all data sets, the size varies from 5 to 50. It is observed that the relevance of our proposed diverse neighbor set is better than GG on the two



(a) DS results (b) KNN results (c) GG results (d) MMD results

Figure 4: Query "superhero": *We use a color matrix to visually show the results. Each column of the matrix represents the probability distribution of a movie among all 100 topics. The darker the color is, the higher the probability is. In the last column, we record the maximum probability of all the 10 returned movies for each topic. A diverse result is expected to have more colors in the Max column. It is clear that K-NN returns non-diverse results, which are mostly from Batman series and Superman series. Only the main topic is clear in the Max column, while all other three sets of results are diverse. However, for MMD, there is a gap from topic around 50 to around 70, i.e. no results cover this range. Indeed, movies, like The Incredible Hulk, Defendor, and The Amazing Spider Man 2, can be both related to "superhero" and cover this range. Besides, there are movies returned by both GG and MMD, such as The Boondock Saints, which is not a classic "superhero" movie, and hence the main topic has lighter color.*



(a) DS results (b) KNN results (c) GG results (d) MMD results

Figure 5: Query "space and alien": *In this example, our angular diverse set returns more diverse results than the other three methods from the view of the color matrices. Besides the main topic, our DS results cover at least 5 minor topics (dark gray color). Similar as the above query, K-NN contains the largest area of white space. GG and MMD returns diverse results, but not as good as DS. Besides, the main topic of their results are not as clear as DS.*

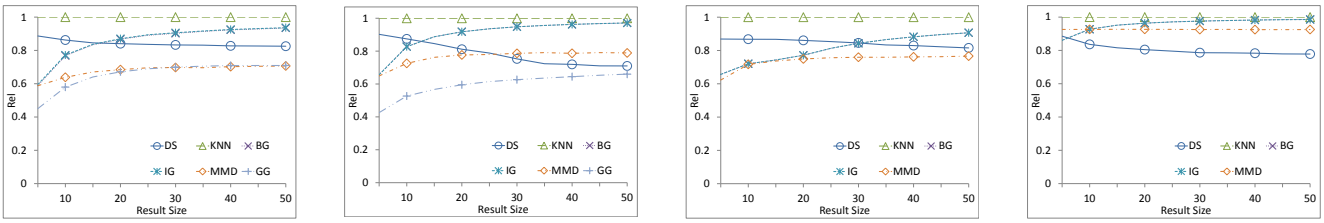
available data sets. Depending on the data sets, there is no all-time winner between MMD and DS in terms of relevance. Another observation is that KNDN methods seem to return nearly K-NNs when the size of results goes larger, especially for higher dimensional data sets Motor and Sun. This may due to the searching strategy of the KNDN methods. They adopt distance browsing, and the criteria is sensitive to the *MinDiv* threshold. In high dimensional spaces, classic distance can hardly distinguish points well.

Vector diversity: Figure 7 shows how different methods perform in terms of VDiv. Although our proposed method is slightly worse than GG, it is more scalable than GG. It is also observed that our method is consistently better than MMD and KNDN methods. It achieves at least 15% improvements in terms of VDiv.

Average angle diversity: Figure 8 plots the average angle diversity of different result sets from different methods. Our proposed diverse neighbor set has the largest average angle diversity, which means the returned points spreads out more around the query than results of other methods. The improvements of our method is roughly $3\% \sim 7\%$ comparing to the second runner-up method.

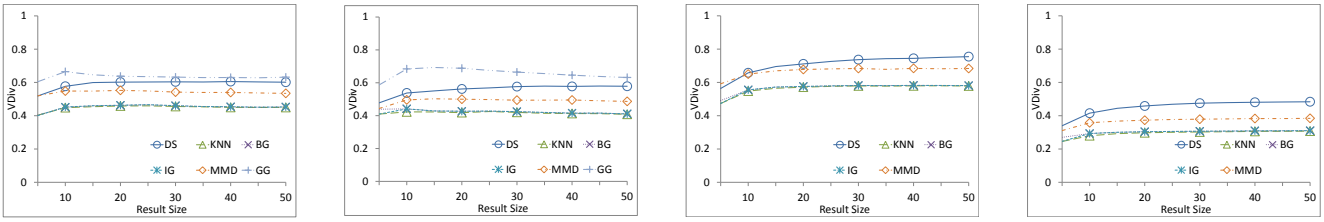
Average distance diversity: By considering the average distance diversity, both GG and MMD perform mostly better, while KNDN methods are only slightly better than K-NN for these high dimensional data sets. Our proposed DS has close performance as GG and MMD, especially when the size of retrieved points is large. It even outperforms others on Sun, which has the largest dimensionality.

In summary, good performance in VDiv and AvgADiv indicates that our method returns most well-spread results. It is not surprising since these two measurements are more or less related to directions. Furthermore, our method performs well in terms of AvgDDiv. This also justifies Theorem 2.1.



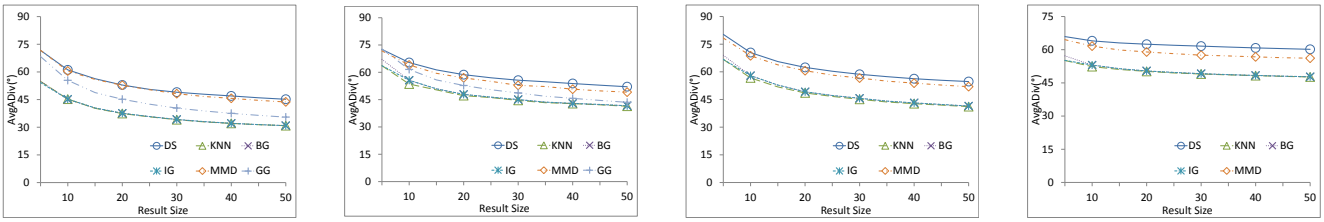
(a) Wine (b) Motor (c) Galaxy (d) Sun

Figure 6: Relevance of result set on real data sets w.r.t required result size



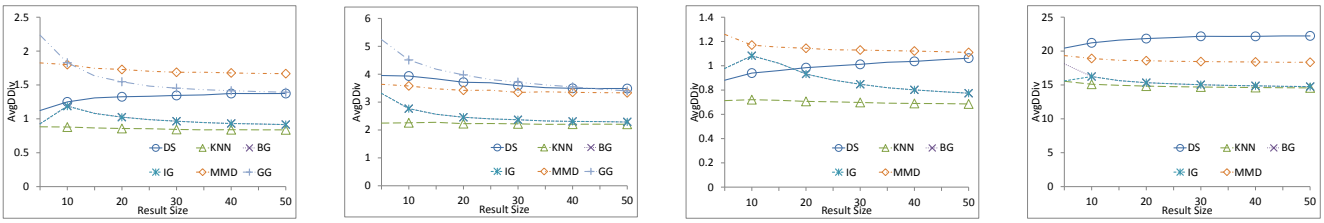
(a) Wine (b) Motor (c) Galaxy (d) Sun

Figure 7: Vector diversity of result set on real data sets w.r.t required result size



(a) Wine (b) Motor (c) Galaxy (d) Sun

Figure 8: Average angle diversity of result set on real data sets w.r.t required result size



(a) Wine (b) Motor (c) Galaxy (d) Sun

Figure 9: Average distance diversity of result set on real data sets w.r.t required result size

Angular diversity can indeed imply distance-based diversity.

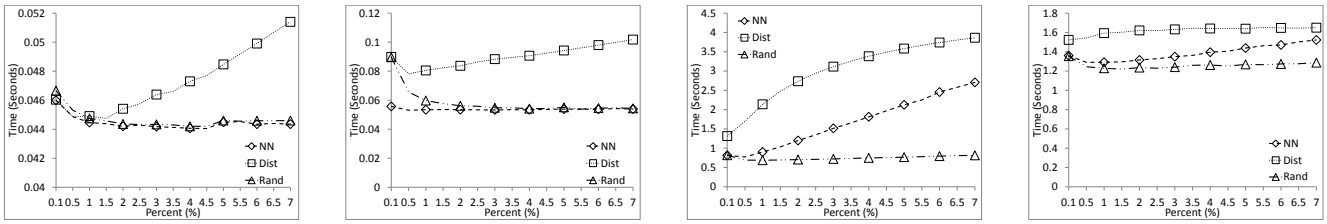
C. Efficiency

We first present how to tune the parameters for the proposed algorithms, and then evaluate their performance.

1) Parameter Tunning: **Choice of reference sets in Two-Scan Algorithm:** As mentioned before in Section III-B, the efficiency of the Two-Scan Algorithm is dependent on the reference set. Here, we evaluate three choices: (1) NN (Nearest neighbors); (2) Rand (Random); (3) Dist (Distance-based partitioned).

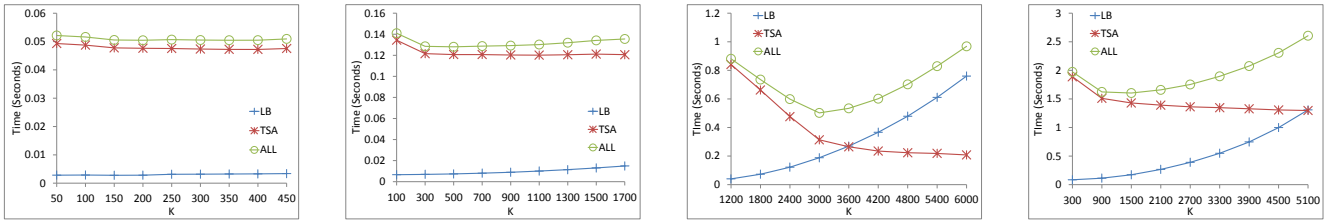
Figure 10 plots the average time costs w.r.t the size of reference set for all the three approaches. For Wine and Motor, nearest neighbors as the reference set can achieve the best performance, while for Galaxy and Sun, random points perform best. If points of a reference set lie in similar directions relative to the query, then such a reference set may not be able to prune true negatives in the first scan. Depending on the dimensionality and the distribution of points in the data sets, nearest neighbors are more likely to suffer this bad case compared to the other two approaches. Thus, in general, nearest neighbors are not recommended as a reference set, especially for large-size and high dimensional data sets.

Size of K-NN to compute angle lower bound: Figure 11 plots the average time cost when answering the sized k-diverse neighbor set problem w.r.t the number of K-NNs used to compute the angle lower bound. As mentioned in Section IV-B, the process is divided into two stages: (1) computing an angle lower bound (LB); (2) computing the diverse neighbor set (TSA, here we adopt the Two-Scan Algorithm). For small-size data sets, hundreds of data points can provide a good lower bound. Thus, to keep increasing the size of K-NNs will not improve the TSA much, but will introduce extra cost in the first stage. In contrast, it is worth adjusting the trade-offs between two stages for large-size high dimensional data sets. An optimized choice can save up to half of the time.



(a) Wine (b) Motor (c) Galaxy (d) Sun

Figure 10: Time costs on real data sets w.r.t size of reference set



(a) Wine (b) Motor (c) Galaxy (d) Sun

Figure 11: Time costs on real data sets w.r.t size of K-NN for angle lower bound

2) Performance Evaluation: We use synthetic data sets to show the efficiency of our proposed algorithms w.r.t different parameter settings. By answering Problem 2, the Naive Algorithm and the Two-Stage Algorithm (TST) are directly compared. In addition, by adopting different approaches in the second stage of TST, the Sorted-Scan Algorithm (SSA) and the Two-Scan Algorithm (TSA) are compared. Since it has been shown that the KNDN methods lose their effectiveness in high dimensional space (returning only K-NNs), while the GG method is inefficient in high dimensional space, we do not consider the competitors in this section. The default setting in our experiments is: d = 200, $|D|$ = 400k, and *Result Size* = 100. For TST, 1500-NNs are used to compute the angle lower bound. For TSA, the set of reference points is chosen randomly of size 0.3% of the data size.

Table V: Time costs (Seconds) w.r.t dimensionality

Dimensionality			50	100	200	400	800
Uniform	Naive		598.98	1504.4	9112.3	N.A.*	N.A.
	TST	LB	0.066	0.09	0.248	0.567	1.209
		SSA	0.556	1.106	2.627	7.401	18.12
		TSA	0.479	0.557	1.004	1.418	2.714
Normal	Naive		595.13	1492.5	9115.1	N.A.	N.A.
	TST	LB	0.057	0.089	0.248	0.564	1.203
		SSA	0.571	1.162	2.627	9.064	22.12
		TSA	0.596	0.811	1.512	1.918	4.011
Skew	Naive		599.39	1503.3	9113.3	N.A.	N.A.
	TST	LB	0.062	0.089	0.249	0.572	1.215
		SSA	0.574	1.163	3.142	9.409	22.22
		TSA	0.641	0.813	1.593	2.121	4.138

*It costs more than 24,000 seconds to run one query, thus, we omit the results here

Effect of the dimensionality: Table V shows the influence of dimensionality on the efficiencies of the two algorithms. An obvious observation is that TST is more scalable on high dimensional data sets. The computation time of Naive grows dramatically when dimensionality increases. Moreover, TSA is superior to SSA for the second stage of TST.

Effect of the data size: We observe from Table VI that the results remain largely the same as that in the previous experiment. TST turns out to be faster for all cases. The impact of data size increasing is larger for the TSA stage compared against the LB stage. This can be explained because the number of nearest neighbors used to compute the angle lower bound is fixed as 1500. For smaller data sets, 1500-NNs may provide more accurate angle lower bound, and thus save time for the second stage. Due to space limitations, we report results for the "Uniform" data set. Similar results are obtained for the other two as well.

Table VI: Time costs (Seconds) w.r.t data size

Data Size			100k	200k	400k	800k	1.6m
Uniform	Naive		147.01	720.84	9114.6	N.A.	N.A.
	TST	LB	0.107	0.113	0.251	0.43	0.734
		SSA	0.532	1.077	2.631	7.332	17.64
		TSA	0.316	0.567	1.032	1.868	3.746

Table VII: Time costs (Seconds) w.r.t result size

Result Size			25	50	100	200	400
Uniform	Naive		9113.3	9113.3	9113.3	9113.3	9113.3
	TST	LB	0.247	0.249	0.249	0.248	0.249
		SSA	2.504	2.511	2.631	3.093	4.982
		TSA	0.747	0.822	0.988	1.376	2.22

Effect of the result size: Table VII presents how the required result size influences the computing time. Since the Naive Algorithm spends nearly all its time on computing and sorting the minimum dominated angles for all points, its computing time is almost constant. Similarly, the LB stage adopts the same strategy but only on a small part of the whole data set, thus its time also remains stable. However, to get a larger result set, the angle lower bound becomes loose. Hence, the second stage of TST costs more time.

VI. Related Work

A. Result Diversification

There has been growing interest in the problem of result diversification. Its goal is to find relevant but also informative answers to users' queries. The answers are expected to be close to queries in the sense of relevance, and to be sufficiently different from each other to provide users an informative view.

Researchers have attacked the problem in various ways. We refer interested readers to these reviews [19], [20] for more details. To summarize existing works in result diversification, we divide them into the following categories:

Score optimization: One of the most popular approaches to diversification is to transform the problem to an optimization problem by associating a diversity score with each point and then try to maximize the total score of a result set. Jain et al. define the K-Nearest Diverse Neighbor (KNDN) problem [9]. Two greedy algorithms are proposed: *KNDN-IG (Immediate Greedy)* approach incrementally adds the next nearest points to the result set only if they are diverse enough from the existing results, while *KNDN-BG (Buffered Greedy)* approach adopts some heuristic to delay the growing of the result set so that it can include better answers in near future. Both methods are integrated with the R-tree index, which can only support up to 10 dimensions practically. The drawback of IG and BG is that they are required to set a diversity threshold, which is non-trivial for an ordinary user especially when the dimensionality goes large. Yu et al. [21], [22], from the perspective of recommendation systems, introduce two heuristic algorithms. They either include new points greedily like [9] or swap points with those are more likely to contribute to the set diversity. Kucuktunc et al. [7] use a linear combination of relevance and diversity measure as their score function. They incrementally explore points around the query by adopting geometric browsing. They also introduce an index-based algorithm, which extends the distance browsing feature of R-trees with diverse choices. Borodin et al. [23] study the max-sum p diversification problem to find a diverse subset, and propose two approximate algorithms for different constraints. Their valuation functions are normalized, monotone submodular set functions, which are general. However, such problem is orthogonal to ours, since it is not query-based.

Representative approach: Another approach to diversification is to find representatives of the data set. A representative is defined as a point that can represent other points. Liu and Jagadish [24] solve the *Many-Answers Problem* by showing users representatives of a data set, where they cluster points and select k-medoids as representatives. A tree-based approach is proposed to find representatives efficiently. Drosou et al. [8] introduce the *DisC diversity* and the *Minimum r-DisC Diverse Subset* problem. The representation relation is defined as: a point can be represented by another point if their distance is within a user-defined threshold r. The goal is to find a minimum-sized subset of points that can represent the whole data set given the user-specific threshold r, while the results are not represented by each other. In machine learning, there is also an important topic known as volume-based diversity, whose goal is to sample a diverse subset of a data set. A set S of k points are selected to maximize the volume of the k-dimensional parallelepiped formed by the corresponding vectors in S. The probability distribution over subsets of a data set is studied as determinantal point processes (DPP). Recently, Deshpande et al. [25] and Anari et al. [26] develop efficient sampling methods for DPP. Above approaches are orthogonal to our work, since their problems are not query-based. Their goal is to find a diverse summary of a data set. Kucuktunc et al. [7] bring the idea of geometric browsing to result diversification. After finding the nearest point p_{1NN} to the query q, all other points are naturally clustered based on their Gabriel neighbor degrees w.r.t p_{1NN}. Points from outer layers are treated to be represented by inner ones. The Gabriel graph-based method (GG) browses the graph layer-by-layer until enough points are retrieved. To return exactly k results, one may need to further choose the most diverse representatives from the last visited layer. There are some drawbacks of the GG method: Firstly, to generate Gabriel graph in high dimensional space is intractable, and GG may not be effective in high dimensional space due to the sparsity. Secondly, it is not appropriate for dynamic data sets.

Neighborhood approach: There also exist approaches of diversifying query results in a post-processing step. A popular approach is the MMD (Maximize Minimum pairwise Distance) method [10], which first includes the neighborhood of the query as the candidate set, and then selects points in the candidate set such that the minimum pairwise distance in the result is maximized.

There may exist other techniques in the information retrieval literature. However, they usually consider the ranking of returned results as an additional problem, which is out of scope of this paper. In the experiments, we compare our method with KNDN-IG (Immediate Greedy), KNDN-BG (Buffered Greedy), GG (Gabriel Graph), and MMD (Maximize Minimum pairwise Distance) because all these methods are query-based: given different queries, the result sets are computed accordingly. To cater for the computation of high dimensional data, we abandon the R-tree-based indexes proposed for competitors, because R-tree-based indexes are of limited use in low dimensional data.

B. The Dominance Relation and Skyline

The dominance relation is widely used in the Skyline operator [27]. Given two points p and q, p is defined to dominate q if p is not worse than q in all dimensions and better than q in at least one dimension. The points that are not dominated by any other point are called Skyline points. These Skyline points are expected to be informative, and the Skyline operator is proposed to find all the Skyline points in a data set. Skyline query processing has received considerable attention in multidimensional databases and has been studied extensively in recent years. Several efficient algorithms have been proposed for the Skyline query. We refer interested readers to this survey [28] for more information.

However, to compute Skylines in high dimensional data sets becomes a non-trivial problem since the result could still be cumbersome [29], [30]. There may be an excessive number of Skyline points returned because as the number of dimensions increases it is unlikely to find a point that dominates another. Hence, several techniques are designed to alleviate this problem. Chan et al. in [31] relax the problem to k-dominant Skyline. A point p k-dominates another point

q if p is not worse than q in k out of d dimensions and better than q in at least one. This relaxation allows more point to be dominated when k is small, and thus reduces the size of result. Another technique was proposed by Lin et al. in [32]. They select Skyline points according to their domination capabilities, and try to maximize the total number of dominated points. In [33], Tao et al. studied a variation of Skyline queries called representative Skyline. A subset of all Skyline points are returned and any non-representative Skyline point has a nearby representative.

Among those works in Skyline, the most relevant work is the Dynamic Skyline [34], where the domination is defined with respect to the query point. Points from original data space are firstly mapped to a new space by a set of one-dimensional functions. Then, the dynamic skyline query returns the ordinary skyline of the transformed points in the new space. The major difference between our problem and the dynamic skyline problem is the concept of angle using the query point as the common endpoint. Any two points from the data set can form one angle. This is different from the mapping in dynamic skyline, where one point's transformation is independent with others. Moreover, unlike the coordinates' comparison strategy in classical skyline problem, our dominating relationship is specially designed to support diversity search in the sense of direction. It is influenced by either the angle threshold or the required results' size.

VII. Conclusion

In this paper, we proposed a novel view of diversity to investigate the result diversification problem in high dimensional space. Aiming to find nearest neighbors in different directions surrounding a given query q, we gave a definition of angular dominance relation. Based on these concepts, we addressed the angular diverse neighbor set problem and the sized diverse neighbor set problem. For each problem, we presented two algorithms. We conducted extensive experiments on synthetic and real data sets. Experimental results showed that our diverse set can capture relevant and diverse points surrounding the query even in high dimensional space.

Acknowledgment

This research is supported by the National Research Foundation, Prime Ministers Office, Singapore under its International Research Centre in Singapore Funding Initiative.

References

[1] R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong, "Diversifying search results," in *WSDM*, 2009, pp. 5–14.

[2] G. Capannini, F. M. Nardini, R. Perego, and F. Silvestri, "Efficient diversification of web search results," *VLDB*, vol. 4, no. 7, pp. 451–459, 2011.

[3] J. Carbonell and J. Goldstein, "The use of mmr, diversity-based reranking for reordering documents and producing summaries," in *SIGIR*, 1998, pp. 335–336.

[4] C. L. Clarke, M. Kolla, G. V. Cormack, O. Vechtomova, A. Ashkan, S. Büttcher, and I. MacKinnon, "Novelty and diversity in information retrieval evaluation," in *SIGIR*, 2008, pp. 659–666.

[5] V. Dang and W. B. Croft, "Diversity by proportionality: an election-based approach to search result diversification," in *SIGIR*, 2012, pp. 65–74.

[6] M. Drosou, H. Jagadish, E. Pitoura, and J. Stoyanovich, "Diversity in big data: A review," *Big Data*, vol. 5, no. 2, pp. 73–84, 6 2017.

[7] O. Kucuktunc and H. Ferhatosmanoglu, "λ-diverse nearest neighbors browsing for multidimensional data," *TKDE*, vol. 25, no. 3, pp. 481–493, 2013.

[8] M. Drosou and E. Pitoura, "Disc diversity: result diversification based on dissimilarity and coverage," *PVLDB*, vol. 6, no. 1, pp. 13–24, 2012.

[9] A. Jain, P. Sarda, and J. R. Haritsa, "Providing diversity in k-nearest neighbor query results," in *PAKDD*, 2004, pp. 404–413.

[10] S. S. Ravi, D. J. Rosenkrantz, and G. K. Tayi, "Heuristic and special case algorithms for dispersion problems," *Operations Research*, vol. 42, no. 2, pp. 299–310, 1994.

[11] D. Francois, V. Wertz, and M. Verleysen, "The concentration of fractional distances," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 7, pp. 873–886, 2007.

[12] H.-P. Kriegel, P. Kröger, and A. Zimek, "Outlier detection techniques," in *Tutorial at SIGKDD*, 2010.

[13] N. Pham and R. Pagh, "A near-linear time approximation algorithm for angle-based outlier detection in high-dimensional data," in *SIGKDD*, 2012, pp. 877–885.

[14] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *ACM Transactions on Interactive Intelligent Systems (TiiS)*, vol. 5, no. 4, p. 19, 2016.

[15] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, "Modeling wine preferences by data mining from physicochemical properties," *Decision Support Systems*, vol. 47, no. 4, pp. 547–553, 2009.

[16] M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[17] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba, "Sun database: Large-scale scene recognition from abbey to zoo," in *CVPR*, 2010, pp. 3485–3492.

[18] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *International journal of computer vision*, vol. 42, no. 3, pp. 145–175, 2001.

[19] M. Drosou and E. Pitoura, "Search result diversification," *SIGMOD Record*, vol. 39, no. 1, pp. 41–47, 2010.

[20] M. Drosou, H. Jagadish, E. Pitoura, and J. Stoyanovich, "Diversity in big data: A review," *Big Data*, vol. 5, no. 2, pp. 73–84, 2017.

[21] C. Yu, L. Lakshmanan, and S. Amer-Yahia, "It takes variety to make a world: diversification in recommender systems," in *EDBT*, 2009, pp. 368–378.

[22] C. Yu, L. V. Lakshmanan, and S. Amer-Yahia, "Recommendation diversification using explanations," in *ICDE*, 2009, pp. 1299–1302.

[23] A. Borodin, A. Jain, H. C. Lee, and Y. Ye, "Max-sum diversification, monotone submodular functions, and dynamic updates," *ACM Transactions on Algorithms (TALG)*, vol. 13, no. 3, p. 41, 2017.

[24] B. Liu and H. V. Jagadish, "Using trees to depict a forest," *VLDB*, vol. 2, no. 1, pp. 133–144, 2009.

[25] A. Deshpande and L. Rademacher, "Efficient volume sampling for row/column subset selection," in *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*. IEEE, 2010, pp. 329–338.

[26] N. Anari, S. O. Gharan, and A. Rezaei, "Monte carlo markov chain algorithms for sampling strongly rayleigh distributions and determinantal point processes," in *Conference on Learning Theory*, 2016, pp. 103–115.

[27] S. Börzsönyi, D. Kossmann, and K. Stocker, "The skyline operator," in *ICDE*, 2001, pp. 421–430.

[28] K. Hose and A. Vlachou, "A survey of skyline processing in highly distributed environments," *The VLDB Journal*, vol. 21, no. 3, pp. 359–384, Jun. 2012.

[29] D. Kossmann, F. Ramsak, and S. Rost, "Shooting stars in the sky: An online algorithm for skyline queries," in *VLDB*, 2002, pp. 275–286.

[30] Y. Yuan, X. Lin, Q. Liu, W. Wang, J. X. Yu, and Q. Zhang, "Efficient computation of the skyline cube," in *VLDB*, 2005, pp. 241–252.

[31] C.-Y. Chan, H. Jagadish, K.-L. Tan, A. K. Tung, and Z. Zhang, "Finding k-dominant skylines in high dimensional space," in *SIGMOD*, 2006, pp. 503–514.

[32] X. Lin, Y. Yuan, Q. Zhang, and Y. Zhang, "Selecting stars: The k most representative skyline operator," in *ICDE*, 2007, pp. 86–95.

[33] Y. Tao, L. Ding, X. Lin, and J. Pei, "Distance-based representative skyline," in *ICDE*, 2009, pp. 892–903.

[34] D. Papadias, Y. Tao, G. Fu, and B. Seeger, "An optimal and progressive algorithm for skyline queries," in *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*. ACM, 2003, pp. 467–478.