Artificial Intelligence 336 (2024) 104206

Contents lists available at ScienceDirect

Artificial Intelligence

journal homepage: www.elsevier.com/locate/artint

Is it possible to find the single nearest neighbor of a query in high dimensions?

Kai Ming Ting [a,*], Takashi Washio [b], Ye Zhu [c], Yang Xu [a], Kaifeng Zhang [a]

[a] National Key Laboratory for Novel Software Technology & School of Artificial Intelligence, Nanjing University, Nanjing, 210023 China
[b] Faculty of Business and Commerce, Kansai University, Osaka, 5648680 Japan
[c] School of Information Technology, Deakin University, Geelong, 3125 Australia

A R T I C L E   I N F O

Keywords:
Curse of dimensionality
Isolation kernel
High dimensions
Nearest neighbor search
Indexed search for exact nearest neighbor search
Anomaly detection using kernel density estimation and t-SNE visualization

A B S T R A C T

We investigate an open question in the study of the curse of dimensionality: Is it possible to find the single nearest neighbor of a query in high dimensions? Using the notion of (in)distinguishability to examine whether the feature map of a kernel is able to distinguish two distinct points in high dimensions, we analyze this ability of a metric-based Lipschitz continuous kernel as well as that of the recently introduced Isolation Kernel. Between the two kernels, we show that only Isolation Kernel has distinguishability and it performs consistently well in four tasks: indexed search for exact nearest neighbor search, anomaly detection using kernel density estimation, t-SNE visualization and SVM classification in both low and high dimensions, compared with distance, Gaussian and three other existing kernels.

1. Introduction

In the last few decades, a significant amount of research has been invested in studying the curse and blessing of dimensionality [1] of a distance measure. We now have a good understanding in terms of (i) concentration of measure [2] and its consequence of being unable to find the nearest neighbor [3], and (ii) hubness [4].

Despite this advancement, breaking the curse is still elusive. A key question remains open: *is it possible to find the single nearest neighbor of a query in high dimensions?* We call it the NNHD problem.

The answer to this question is negative, based on the current analyses on distance [3,5,6]. Rather than finding the single nearest neighbor of a query, the issue has been diverted to finding all data points belonging to the same cluster as the query, in a distribution with well-separated clusters [3,5]. These nearest neighbors are ‘meaningful’ for the purpose of separating different clusters in a dataset. But the (single) nearest neighbor, in the true sense of the term, cannot be obtained using a distance measure, even in the special distribution of well-separated clusters described above [3].

We show here that the inability to address the NNHD problem can be explained using the notion of indistinguishability of a kernel, i.e., the kernel’s feature map is unable to distinguish two distinct points. This is manifested as being unable to find the single nearest neighbor of a query because there are many neighbors which cannot be distinguished from each other, especially in high dimensions.

Our theoretical analysis focuses on the NNHD problem in terms of **distinguishability** or **indistinguishability** of a kernel.

Our contributions are:

* Corresponding author.
E-mail addresses: tingkm@nju.edu.cn (K.M. Ting), washio@ar.sanken.osaka-u.ac.jp (T. Washio), ye.zhu@ieee.org (Y. Zhu), xuyang@lamda.nju.edu.cn (Y. Xu), zhangkf2022@lamda.nju.edu.cn (K. Zhang).

https://doi.org/10.1016/j.artint.2024.104206
Received 5 August 2023; Received in revised form 16 July 2024; Accepted 14 August 2024

Available online 21 August 2024

I Revealing that a recently introduced Isolation Kernel (IK) [7,8] is a measure which has **distinguishability**, but existing metric-based Lipschitz continuous kernels have **indistinguishability**.

II Determining three key factors in IK that enable **distinguishability**. (a) Space partitioning using an isolation mechanism is the key ingredient in measuring similarity between two points in IK. (b) The probability of a point falling into a partition is independent of data distribution, the distance measure used to create the partitions and the data dimensions. The partitions enable two distinct points to be differentiated, given a sufficient number of partitionings. (c) The unique feature map of IK has its dimensionality linked to a concatenation of these partitionings, and increased dimensionality increases IK's distinguishability (see the next point).

III Proving that IK's unique feature map produced by a finite dataset has **distinguishability**, independent of the number of data dimensions and data distribution. Our theorem suggests that increasing the number of partitionings (i.e., the dimensionality of the feature map) leads to increased distinguishability.

IV Verifying IK's **distinguishability** in two sets of experiments. First, IK finds the single nearest neighbor in high dimensions; while many existing measures fail under the same condition. Second, IK enables effective indexing for exact nearest neighbor search, anomaly detection using kernel density estimation, t-SNE visualization and SVM classification in both low and high dimensions.

V Showing that Euclidean distance, Gaussian and three other existing kernels have largely poor results in datasets having high input dimensions as well as intrinsic dimensions, reflecting the current understanding of these measures.

2. Related work

2.1. The curse of dimensionality

One of the early investigations of the curse of dimensionality is in terms of concentration of measure [9,2]. Talagrand [2] summarizes the intuition behind the concentration of measure as follows: "a random variable that depends (in a 'smooth' way) on the influence of many independent random variables (but not too much on any of them) is essentially constant." Translating this statement in the context of a high dimensional space, a distance measure which depends on many independent dimensions (but not too much on any of them) is essentially constant.

Beyer et al. [3] analyze the conditions under which a nearest neighbor is 'meaningful' when a distance measure is employed in high dimensions. A brief summary of the conditions in high dimensions is given as follows:

a) Finding the (single) nearest neighbor of a query is not meaningful because every point from the same cluster in the database has approximately the same distance to the query. In other words, the variance of distance distribution within a cluster approaches zero as the number of dimensions increases to infinity.
b) Despite this fact, finding the exact match is still meaningful.
c) The task of retrieving all points belonging to the same cluster as the query is meaningful because although variance of distances of all points within the cluster is zero, the distances of this cluster to the query are different from those of another cluster—making a distinction between clusters possible.
d) The task of retrieving the nearest neighbor of a query, which does not belong to any of the clusters in the database, is not meaningful for the same reason given in (a).

Beyer et al. [3] reveal that finding (non-exact match) nearest neighbors is 'meaningful' only if the dataset has clusters (or structure), where the nearest neighbors refer to any/all points in a cluster rather than a specific point which is the nearest among all points in the dataset. In short, a distance measure cannot produce the 'nearest neighbor' in high dimensions, in the true sense of the term. Hereafter the term 'nearest neighbor' is referred to the single nearest neighbor in a dataset, not many nearest neighbors in a cluster.

Beyer et al. [3] describe the notion of instability of a distance measure as follows: "*A nearest neighbor query is unstable for a given ϵ if the distance from the query point to most data points is less than $(1+\epsilon)$ times the distance from the query point to its nearest neighbor.*"

Current analyses on the curse of dimensionality focus on distance measures [3,10,11,4]. While random projection preserves structure in the data, it still suffers the concentration effect [12]. Current methods dealing with high dimensions are: dimension reduction [13,14]; suggestions to use non-metric distances, e.g., fractional distances [11]; and angle-based measures [15]. None of these measures have been shown to be able to find the nearest neighbor in high dimensions.

Yet, the influence of the concentration effect in practice is less severe than suggested mathematically under some condition [16]. For example, k-nearest neighbor (kNN) classifier still yields reasonable high accuracy in high dimensions [17]; while the concentration effect would have rendered kNN producing random prediction. One possible explanation of this phenomenon is that the high dimensional datasets employed have low intrinsic dimensions (e.g., [16,18]).

In a nutshell, existing works center around *reducing* the effect of concentration of measures, knowing that the nearest neighbor could not be obtained in high dimensions. A saving grace is that many high dimensional datasets used in practice have low intrinsic dimensions—leading to a significantly reduced concentration effect.

Yet, a key question remains open: *is it possible to find the single nearest neighbor of a query in high dimensions?* (The NNHD problem).

Our work shows that the NNHD problem can be addressed, i.e., finding the nearest neighbor in high dimensions is possible by using a recently introduced measure called Isolation Kernel [7,8], even in a dataset with high intrinsic dimensions.

We describe the current understanding of Isolation Kernel in the next section.

2.2. Isolation kernel

Unlike commonly used distance or kernels, Isolation Kernel [7,8] has no closed-form expression, and it is derived directly from a given dataset without learning; and the similarity between two points is computed based on the partitions created in data space.

The key requirement of Isolation Kernel (IK) is a space partitioning mechanism which isolates a point from the rest of the points in a sample set. This can be achieved in several ways, e.g., isolation forest [19,7], Voronoi Diagram [8] and isolating hyperspheres [20]. Isolation Kernel [7,8] is defined as follows.

Let $\Omega \subset \mathbb{R}^d$ be a data space, $D \subset \Omega$ be a finite dataset, and $\mathcal{P}_D$ be the probability distribution that generates D. The support of $\mathcal{P}_D$, denoted as $\mathcal{X}_D$, is an example of Ω.

Let $\mathcal{D} \subset D$, $|\mathcal{D}| = \psi$ and each point in $\mathcal{D}$ has the equal probability of being selected from D.

A partitioning H is derived from $\mathcal{D}$, where each partition $\theta[\mathbf{z}] \in H$ isolates a point $\mathbf{z} \in \mathcal{D}$ from the rest of the points in $\mathcal{D}$. The union of all partitions of H covers the entire data space.

Let $\mathbb{H}_\psi(D)$ denote the set of all possible partitionings H derived from D.

Definition 1. Isolation Kernel of any two points $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ is defined to be the expectation taken over the probability distribution $P_{\mathbb{H}_\psi(D)}$ on all partitionings $H \in \mathbb{H}_\psi(D)$ that both $\mathbf{x}$ and $\mathbf{y}$ fall into the same isolating partition $\theta[\mathbf{z}] \in H$:

$$K_\psi(\mathbf{x}, \mathbf{y} \mid D) = \mathbb{E}_{H \sim P_{\mathbb{H}_\psi(D)}}[\mathbb{1}(\mathbf{x}, \mathbf{y} \in \theta[\mathbf{z}] \mid \theta[\mathbf{z}] \in H)], \tag{1}$$

where $\mathbb{1}(\cdot)$ is an indicator function.

In practice, Isolation Kernel K_ψ is constructed using a finite number of partitionings $H_i, i = 1, \dots, t$, where each H_i is created using randomly subsampled $\mathcal{D}_i \subset D$; and θ is a shorthand for $\theta[\mathbf{z}]$:

$$\begin{aligned} K_\psi(\mathbf{x}, \mathbf{y} \mid D) &\simeq \frac{1}{t} \sum_{i=1}^{t} \mathbb{1}(\mathbf{x}, \mathbf{y} \in \theta \mid \theta \in H_i) \\ &= \frac{1}{t} \sum_{i=1}^{t} \sum_{\theta \in H_i} \mathbb{1}(\mathbf{x} \in \theta) \mathbb{1}(\mathbf{y} \in \theta). \end{aligned} \tag{2}$$

This gives a good approximation of $K_\psi(\mathbf{x}, \mathbf{y} \mid D)$ when $|D|$ and t are sufficiently large to ensure that the ensemble is obtained from a sufficient number of mutually independent $\mathcal{D}_i, i = 1, \dots, t$.

Given H_i, let $\Phi_i(\mathbf{x}|D)$ be a ψ-dimensional binary column (one-hot) vector representing all $\theta_j \in H_i$, $j = 1, \dots, \psi$; where $\mathbf{x}$ falls into only one of the ψ partitions. The j-component of the vector is: $\Phi_{ij}(\mathbf{x}|D) = \mathbb{1}(\mathbf{x} \in \theta_j \mid \theta_j \in H_i)$. Given t partitionings, $\Phi(\mathbf{x}|D)$ is the concatenation of $\Phi_1(\mathbf{x}|D), \dots, \Phi_t(\mathbf{x}|D)$.

Definition 2. Isolation Kernel K_ψ has a feature map $\Phi : \mathbf{x} \to \{0,1\}^{t \times \psi}$. K_ψ has no closed form expression and is expressed in terms of Φ as follows:

$$K_\psi(\mathbf{x}, \mathbf{y} \mid D) \simeq \frac{1}{t} \langle \Phi(\mathbf{x}|D), \Phi(\mathbf{y}|D) \rangle .$$

For brevity, we drop '$|D$' hereafter when the context is clear.

The feature map $\Phi(\mathbf{x})$ is sparse because it has exactly t out of $t\psi$ elements having 1 and the rest are zeros. This gives $\| \Phi(\mathbf{x}) \| = \sqrt{t}$. Note that while every point has $\| \Phi(\mathbf{x}) \| = \sqrt{t}$ and $\Phi_i(\mathbf{x}) = \mathbb{1}$ for all $i \in [1, t]$, they are not all the same point, where $\mathbb{1}$ is a shorthand of the value of $\Phi_i(\mathbf{x})$ such that $\Phi_{ij}(\mathbf{x}) = 1$ and $\Phi_{ik}(\mathbf{x}) = 0, \forall k \neq j$ of any $j \in [1, \psi]$.

Isolation Kernel is a positive definite kernel with probability 1 in the limit of $t \to \infty$ because its Gram matrix is full rank as $\Phi(\mathbf{x})$ for all points $\mathbf{x} \in D$ are mutually independent (see [20] for details). In other words, the feature map coincides with a reproducing kernel Hilbert space (RKHS) associated with the Isolation Kernel.

This finite-dimensional feature map has been shown to be the key in leading to fast runtime in kernel-based anomaly detectors [20] and enabling the use of fast linear SVM [21,22].

We show in Section 3.2 that the feature map $\Phi(\cdot)$ of IK is instrumental in making IK to have distinguishability in high dimensions, whereas existing metric-based Lipschitz continuous kernels do not have this ability.

Fig. 1 illustrates a set of t partitionings of IK using Voronoi Diagrams. The similarity between $\mathbf{x}$ and $\mathbf{y}$ is the probability of both points falling into a same partition (Voronoi cell). The illustration shows that the dense region is partitioned into cells smaller than those in the sparse region. Therefore, data points in the dense region (being in small cells) have lower similarities than data points with the same inter-point distance in the sparse region (being in the large cells).

IK has a sharpness parameter ψ (corresponds to the bandwidth parameter in Gaussian kernel): the larger ψ is, the sharper the kernel distribution is, as illustrated in Fig. 2.[1]

The key symbols and notations used in this paper are provided in Table 1.

[1] A guide for parameter setting of IK is provided in Appendix A.

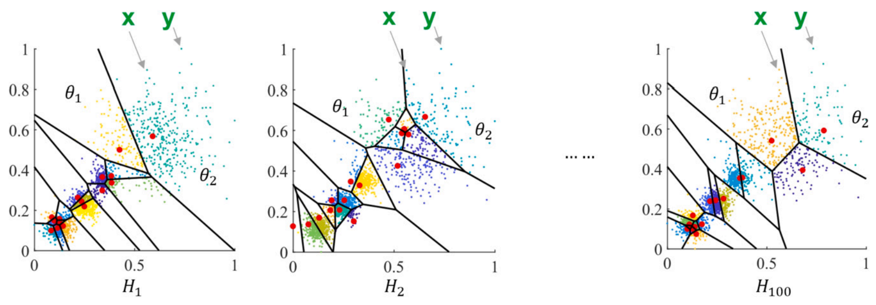


Fig. 1. An illustration of the partitionings using Voronoi Diagrams to compute the similarity as measured by IK for **x** and **y**. The red points are the ψ points used to build the ψ partitions or Voronoi cells in each partitioning H_i, where $i = 1, 2, \ldots, 100$. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

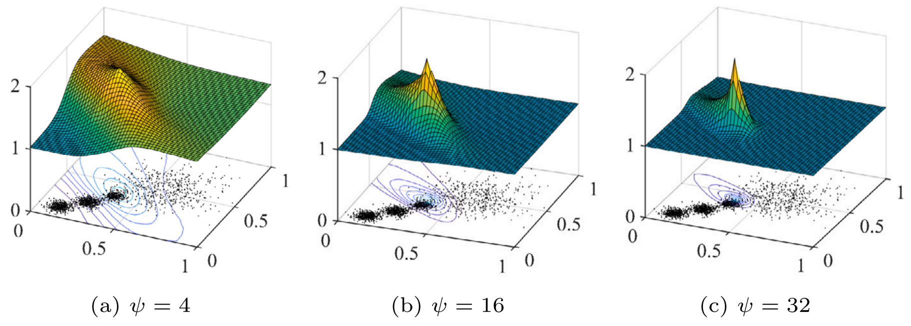


Fig. 2. An illustration of the kernel distribution of IK with different ψ values. The contours are with reference to point $\mathbf{x} = (0.36, 0.40)$.

Table 1
Key symbols and notations used.

Symbol	Description
D	A dataset in $\mathbb{R}^d$
$\mathcal{P}_D$	Probability distribution that generates D
$\mathcal{X}_D$	Support of $\mathcal{P}_D$
κ	Metric-based Lipschitz continuous kernel function
ϕ	Feature map of kernel κ
$\mathcal{H}_\kappa$	Feature space of kernel κ
K	Isolation Kernel
Φ	Feature map of Isolation Kernel
t	Number of partitionings in Isolation Kernel
H	One partitioning
θ	One isolating partition in partitioning H
ψ	Number of isolating partitions in partitioning H
$\mathcal{D}$	Subset of D used to generate H, $\lvert\mathcal{D}\rvert = \psi$
$\mathbb{H}_\psi(D)$	Set of all possible partitionings H derived from D
Ω	Data space
F	Probability measure
m	A metric
ℓ_p	ℓ_p norm
L	Lipschitz constant

3. (In)distinguishability of a kernel

In this section, we show that IK is a measure that can address the NNHD problem in any high dimensional data space; while metric-based Lipschitz continuous kernels, including Gaussian and Laplacian kernels, can not.

We introduce the notion of distinguishability of two distinct points in a feature space $\mathcal{H}$ produced by a kernel, evaluated on a finite dataset, to perform the analysis.

The intuition is given as follows. The key to distinguishability is the use of space partitioning as the means to compute the similarity. This enables points which fall into a same partition to be differentiated from points which fall outside the partition. In addition, two distinct points (even though they may fall into a same partition in some partitionings out of many) are distinguishable in the IK feature space, given a sufficient number of partitionings. A metric-based Lipschitz continuous kernel employs no such partitioning to determine its similarity, but solely relies on Euclidean distance in $\mathbb{R}^d$. The distance calculation in its feature space (or in $\mathbb{R}^d$ space) is the root cause of its indistinguishability.

This notion of distinguishability is not completely alien, and it has some relationship to instability of a measure [3,6], mentioned in Section 2.1. A distance measure is said to be unstable if a query returns many nearest neighbors which could not be differentiated. The new formal definition of distinguishability paves the way for a tractable analysis that has not been attempted before.

Given two distinct points $\mathbf{x}_a, \mathbf{x}_b \in \Omega$ and their feature vectors $\phi(\mathbf{x}_a), \phi(\mathbf{x}_b) \in \mathcal{H}_\kappa$ produced by a kernel κ, the notions of distinguishability and indistinguishability are defined as follows:

Definition 3. $\mathcal{H}_\kappa$ has **distinguishability** using a given metric m on $\mathcal{H}_\kappa$ if there exists some setting v of κ and a function $\gamma(v)$, then the following expression holds for any $e > 0$, any $0 < \delta \leq 1$ and any number of dimensions d of the data space.

$$P(m(\phi(\mathbf{x}_a), \phi(\mathbf{x}_b)) \leq e) \leq \gamma(v) \leq \delta. \tag{3}$$

Otherwise, $\mathcal{H}_\kappa$ has **indistinguishability**.

Note that when $\mathcal{H}_\kappa$ has **distinguishability**, even under $\delta \approx 0$ and very high dimensions d, Eq (3) can still be satisfied for some setting v with any e.

In the following, we first show in Section 3.1 that a feature space $\mathcal{H}_\kappa$ produced by any metric-based Lipschitz continuous kernel κ wrt a finite dataset D has indistinguishability when the number of dimensions d of the data space is very large. Then, in Section 3.2, we show that a feature space $\mathcal{H}_K$ produced by Isolation Kernel K implemented using the Voronoi Diagram [8] has distinguishability for any d.

3.1. $\mathcal{H}_\kappa$ of a metric-based Lipschitz continuous kernel κ has indistinguishability

Let a triplet (Ω, m, F) be a probability metric space where $m : \Omega \times \Omega \mapsto \mathbb{R}$ is a metric, and $F : 2^\Omega \mapsto \mathbb{R}$ is a probability measure. ℓ_p-norm is an example of m. Ω is the support of F, and thus F is assumed to have non-negligible probabilities over the entire Ω, i.e., points drawn from F are widely distributed over the entire data space.

Further, let $f : \Omega \mapsto \mathbb{R}$ be a Lipschitz continuous function, i.e., $|f(\mathbf{x}_a) - f(\mathbf{x}_b)| \leq m(\mathbf{x}_a, \mathbf{x}_b), \forall \mathbf{x}_a, \mathbf{x}_b \in \Omega$; and M be a median of f defined as:

$$F(\{\mathbf{x} \in \Omega \mid f(\mathbf{x}) \leq M\}) = F(\{\mathbf{x} \in \Omega \mid f(\mathbf{x}) \geq M\}).$$

Then, the following proposition holds [6,23].

Proposition 1. *$F(\{\mathbf{x} \in \Omega \mid f(\mathbf{x}) \in [M - \epsilon, M + \epsilon]\}) \geq 1 - 2\alpha(\epsilon)$ holds for any $\epsilon > 0$, where $\alpha(\epsilon) = C_1 e^{-C_2 \epsilon^2 d}$ with two constants $0 < C_1 \leq \frac{1}{2}$ and $C_2 > 0$.*

For our analysis on the indistinguishability of a feature space $\mathcal{H}_\kappa$ associated with any metric-based Lipschitz continuous kernel, we reformulate this proposition by replacing $f(\mathbf{x})$ and M with $m(\mathbf{x}, \mathbf{y})$ and $M(\mathbf{y})$, respectively, for a given $\mathbf{y} \in \Omega$, where $M(\mathbf{y})$ is the median of $m(\mathbf{x}, \mathbf{y})$ conditioned on $\mathbf{y}$. That is:

$$|m(\mathbf{x}_a, \mathbf{y}) - m(\mathbf{x}_b, \mathbf{y})| \leq m(\mathbf{x}_a, \mathbf{x}_b), \forall \mathbf{x}_a, \mathbf{x}_b \in \Omega, \text{ and}$$
$$F(\{\mathbf{x} \in \Omega \mid m(\mathbf{x}, \mathbf{y}) \leq M(\mathbf{y})\}) = F(\{\mathbf{x} \in \Omega \mid m(\mathbf{x}, \mathbf{y}) \geq M(\mathbf{y})\}).$$

The inequality in the first formula always holds because of the triangular inequality of m. Then, we obtain the following corollary:

Corollary 1. *Under the same condition in Proposition 1, the inequality $F(A_\epsilon(\mathbf{y})) \geq 1 - 2\alpha(\epsilon)$ holds for any $\mathbf{y} \in \Omega$ and any $\epsilon > 0$, where $A_\epsilon(\mathbf{y}) = \{\mathbf{x} \in \Omega \mid m(\mathbf{x}, \mathbf{y}) \in [M(\mathbf{y}) - \epsilon, M(\mathbf{y}) + \epsilon]\}$.*

This corollary provides an important fact on the indistinguishability of two distinct points using a class of metric-based Lipschitz continuous kernels such as Gaussian kernel:

$$\kappa(\mathbf{x}, \mathbf{y}) = f_\kappa(m(\mathbf{x}, \mathbf{y})),$$

where f_κ is Lipschitz continuous and monotonically decreasing for $m(\mathbf{x}, \mathbf{y})$.

Given a data set $D = \{\mathbf{x}_i \in \mathbb{R}^d \mid i = 1, \ldots, n\} \sim F^n$, where $\mathbf{x}_i$ is i.i.d. drawn from any probability distribution F on Ω. Let the feature vectors of $\mathbf{x}_a$ and $\mathbf{x}_b$ be $\phi(\mathbf{x}_a) = [\kappa(\mathbf{x}_a, \mathbf{x}_1), \ldots, \kappa(\mathbf{x}_a, \mathbf{x}_n)]^\top$ and $\phi(\mathbf{x}_b) = [\kappa(\mathbf{x}_b, \mathbf{x}_1), \ldots, \kappa(\mathbf{x}_b, \mathbf{x}_n)]^\top$ in a feature space $\mathcal{H}_\kappa$.

Lemma 1. *$\mathcal{H}_\kappa$ has the following property for two distinct points $\mathbf{x}_a$ and $\mathbf{x}_b$ i.i.d. drawn from F on Ω:*

$$P\left(\ell_p(\phi(\mathbf{x}_a), \phi(\mathbf{x}_b)) \leq 2L\epsilon n^{1/p}\right) \geq (1 - 2\alpha(\epsilon))^{2n},$$

for any $\epsilon > 0$, where ℓ_p is an ℓ_p-norm on $\mathcal{H}_\kappa$, and L is the Lipschitz constant of f_κ.

When casting m to ℓ_p-metric, $e = 2L\epsilon n^{1/p}$ and $\delta = (1-2\alpha(\epsilon))^{2n}$, there is no setting ν of the kernel κ satisfying the condition on $\gamma(\nu)$ in Definition 3, and thus we have the following theorem:

Theorem 1. *Under the same condition as Lemma 1, the feature space $\mathcal{H}_\kappa$ has indistinguishability using ℓ_p-metric on $\mathcal{H}_\kappa$.*

This result implies that the feature space $\mathcal{H}_\kappa$ produced by any finite dataset and any metric-based Lipschitz continuous kernels such as Gaussian and Laplacian kernels have indistinguishability when the data space has a high number of dimensions d. In other words, these kernels cannot address the NNHD problem.

3.2. $\mathcal{H}_K$ of isolation kernel K has distinguishability

We show that the feature map of Isolation Kernel implemented using the Voronoi Diagram [8], which is not a metric-based Lipschitz continuous kernel, has distinguishability.

Let $\mathbf{z}$ be i.i.d. drawn from any probability distribution F on Ω. Devroye et al. [24] have showed that the F-measure of the Voronoi cell $\theta(\mathbf{z})$, which is equivalent to $P(\mathbf{x} \in \theta(\mathbf{z}))$ under $\mathbf{x} \sim F$ on Ω, asymptotically converges to $1/\psi$ as $\psi \to \infty$, and moreover its variance asymptotically converges to 0 as well. Based on this finding, we remark with the following statement.

Remark 1. Given $\mathcal{D} = \{\mathbf{z}_1, \dots, \mathbf{z}_\psi\} \sim F^\psi$ for large ψ, the probability that $\mathbf{z}_j$ is the nearest neighbor of $\mathbf{x} \sim F$ on Ω for any index $j = 1, \dots, \psi$ is given as:

$$P(\mathbf{x} \in \theta_j) \simeq 1/\psi,$$

where every $\mathbf{z}_j$ forms its Voronoi partition $\theta_j \subset \Omega$.

Note that Remark 1 relies on nearest neighbor search to determine $\mathbf{x}$'s nearest neighbor in $\mathcal{D}$, and this process is equivalent to identifying the Voronoi partition which contains $\mathbf{x}$. In plain language, any point in the space must fall into one of the ψ partitions with equal probability in a partitioning to enable two distinct points to be distinguishable, as implied in the above-mentioned intuition.

Remark 1 points to a simple but nontrivial result that $P(\mathbf{x} \in \theta_j)$ is independent of F, $m(\mathbf{x}, \mathbf{y})$ and data dimension d.

Let the feature vectors of two distinct $\mathbf{x}_a$ and $\mathbf{x}_b$ be $\Phi(\mathbf{x}_a)$ and $\Phi(\mathbf{x}_b)$ in a feature space $\mathcal{H}_K$ associated with Isolation Kernel K, implemented using the Voronoi Diagram (as given in Definition 2) using $\mathcal{D}_i = \{\mathbf{z}_1, \dots, \mathbf{z}_\psi\} \sim F^\psi$ defined in Remark 1.

Lemma 2. *$\mathcal{H}_K$ has the following property for two distinct points $\mathbf{x}_a$ and $\mathbf{x}_b$ i.i.d. drawn from F on Ω: $\forall e > 0, 0 < \delta \le 1, \exists \psi > 1, \exists t > e^p$ such that*

$$P(\ell_p(\Phi(\mathbf{x}_a), \Phi(\mathbf{x}_b)) \le e) \le 2exp(-\frac{t(1-\frac{e^p}{t})^2}{2}) < \delta,$$

where t and ψ are parameters of Isolation Kernel K, used to determine its feature map Φ; and p is the parameter in the ℓ_p metric.

According to Definition 3, when casting m to ℓ_p-metric and $\gamma(\nu) = 2\exp(-\frac{t(1-\frac{e^p}{t})^2}{2})$, where $\nu = t$ with some ψ setting of Isolation Kernel K, we have the following theorem:

Theorem 2. *Under the same condition as Lemma 2, the feature space $\mathcal{H}_K$ has distinguishability using ℓ_p-metric on $\mathcal{H}_K$.*

Proofs of Lemmas 1 to 2 and Theorems 1 & 2 are given in Appendix B.

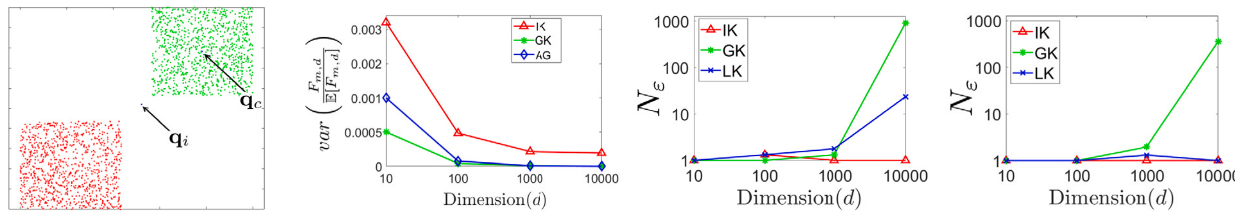
In summary, **the IK implemented using the Voronoi Diagram has distinguishability for any probability distribution of the data space**.

3.3. The roles of ψ & t and the time complexity of isolation kernel

The sample size (or the number of partitions in one partitioning) ψ in IK has a function similar to the bandwidth parameter in Gaussian kernel, i.e., the higher ψ, the sharper the IK distribution [7] (the smaller the bandwidth, the sharper the Gaussian kernel distribution). In other words, ψ needs to be tuned for a specific dataset to produce a good task-specific performance.

The number of partitionings t has three roles. First, the higher t is, the better Eq (2) is in estimating the expectation described in Eq (1). High t also leads to low variance of the estimation. Second, with its feature map, t can be viewed as IK's user-definable effective number of dimensions in Hilbert space. Third, Theorem 2 reveals that increasing t leads to increased distinguishability. The first two roles were uncovered in the previous studies [7,8,20]. We reveal the last role here in Theorem 2. These effects can be seen in the experiment reported in Appendix C.

In summary, **IK, which has the dimensionality of its feature map linked to distinguishability, is unique among existing measures**. In contrast, Gaussian kernel (and many existing metric-based kernels) have a feature map with intractable dimensionality; yet, they have indistinguishability.



(a) Dataset used (b) (In)stability of a measure (c) $\mathbf{q}_i$: between 2 clusters (d) $\mathbf{q}_c$: cluster center

Fig. 3. The instability of a measure κ: Gaussian kernel (GK) and Linear kernel (LK) versus IK. The dataset used has 1,000 data points per cluster: the number of dimensions is increased from 10 to 10,000. Query $\mathbf{q}_c$ is used in subfigure (b). $\varepsilon = 0.005$ is used in subfigures (c) & (d).

Time complexity: IK, implemented using the Voronoi Diagram, has time complexity $O(ndt\psi)$ to convert n points in data space to n vectors in its feature space. Note that each D implemented using one-nearest-neighbor produces a Voronoi Diagram implicitly; no additional computation is required to build the Voronoi diagram explicitly (see Qin et al. [8] for details). In addition, as this implementation is amenable to acceleration using parallel computing, it can be reduced by a factor of w parallelizations to $O(\frac{n}{w}dt\psi)$.

4. Empirical verification

We verify that IK has **distinguishability** by conducting two experiments. In Section 4.1, we verify the theoretical results in terms of instability of a measure, as used in a previous study [3]. This examines the (in)distinguishability of Linear, Gaussian and Isolation Kernels. In the second experiment in Section 4.2, we explore the impact of IK having distinguishability in four tasks: ball-tree indexing [25] exact nearest neighbor search, anomaly detection using kernel density estimation [26], t-SNE visualization [14], and SVM classification.

4.1. Instability of a measure

The concentration effect is assessed in terms of the variance of measure distribution $F_{m,d}$ wrt the mean of the measure, i.e., $var\left(\frac{F_{m,d}}{\mathbb{E}[F_{m,d}]}\right)$, as the number of dimensions d increases, based on a measure $m(\mathbf{x},\mathbf{y}) = 1 - \kappa(\mathbf{x},\mathbf{y})$. A measure is said to have the concentration effect if the variance of the measure distribution in dataset D approaches zero as $d \to \infty$.

Definition 3 on the distinguishability of a kernel κ can be interpreted in terms of instability of a measure. Following [3], we use the same notion of instability of a measure as a result of a nearest neighbor query. Let $m(\mathbf{q}|D)$ be the distance as measured by κ from query $\mathbf{q}$ to its nearest neighbor in dataset D; and N_ε be the number of points in D having distances wrt $\mathbf{q}$ less than $(1+\varepsilon)m(\mathbf{q}|D)$.

When there are many close nearest neighbors for any small $\varepsilon > 0$, i.e., N_ε is large, then the nearest neighbor query is said to be unstable.

By casting $(1+\varepsilon)m(\mathbf{q}|D)$ to e and $\mathbf{x}_a$ be any data point other than the nearest neighbor of $\mathbf{x}_b = \mathbf{q}$ in Definition 3, if some setting v of κ can be found to satisfy Eq (3), then almost no data points exist within the range $[m(\mathbf{q}|D), (1+\varepsilon)m(\mathbf{q}|D)]$, *i.e.*, the instability is avoided for any large d. On the other hand, if no setting v of κ can be found to satisfy Eq (3), then $\mathcal{H}_\kappa$ has indistinguishability and the kernel κ is unstable.

Thus we use N_ε as a proxy for indistinguishability in the empirical verification in this section: for small ε, high N_ε signifies indistinguishability; and $N_\varepsilon = 1$ or close to 1 denotes distinguishability.

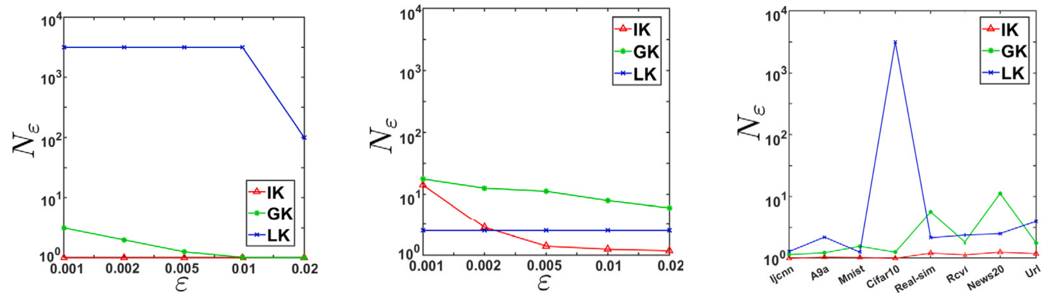
Fig. 3 shows the verification outcomes using the dataset shown in Fig. 3(a). Fig. 3(b) shows that Gaussian kernel (GK) is unstable, that is, $var\left(\frac{F_{m,d}}{\mathbb{E}[F_{m,d}]}\right) \to 0$ as $d \to \infty$. Yet, using Isolation Kernel (IK), the variance approaches a non-zero constant under the same condition.

Figs. 3(c) & 3(d) show that N_ε increases as d increases for GK for two different queries; and LK is unstable in one query. Yet, IK has almost always maintained a constant N_ε $(=1)$ as d increases.

Fig. 4 shows the average N_ε over ten query points randomly selected in a dataset; it is conducted over eight real-world datasets. Across varying values of ε and on different real datasets, IK almost always has better distinguishability than both LK and GK. On some high-dimensional datasets, LK and GK has a significant higher N_ε than IK. The results reaffirm that GK and LK are both unstable on some high-dimensional datasets, i.e., GK is most unstable on all high dimensional datasets (Real-sim, Rcv1, News20 and Url); and LK is most unstable on the dense dataset such as Cifar10 (having the highest non-zero values (%nnz), out of all real datasets, shown in Table 7).

SNN, AG, Linear kernel and fractional distances: Here we examine the instability of Shared Nearest Neighbor (SNN) [27,28], Adaptive Gaussian kernel (AG) [29] and fractional distances [11]. The first two measures employ k nearest neighbors as a means to make the measures data dependent.

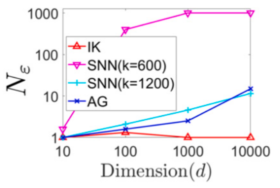
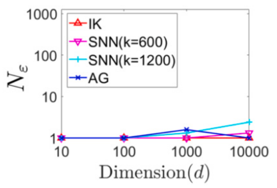
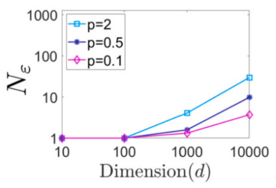
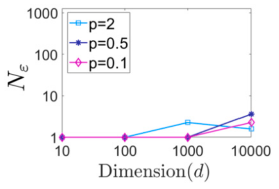
Our experiment using SNN has shown that it has query stability if the query is within a cluster, but it has the worst query instability when the query is outside any clusters when k is set less than the cluster size (1000 in this case). This result is shown in the first two subfigures in Table 2. AG ($k = 200$) has a similar behavior as SNN that uses $k > 1000$.



(a) Cifar: $d = 3,072$ (b) News20: $d = 1,355,191$ (c) $\varepsilon = 0.005$

Fig. 4. The instability of a measure κ (IK, GK and LK) on real datasets. In (a) and (b), ε is varied from 0.001 to 0.02. In (c), we use $\varepsilon = 0.005$ for each of the eight datasets. The datasets in x-axis are ordered in increasing number of dimensions (as in Table 3). Each average N_ε is computed from 10 query points randomly selected from a dataset.

Table 2
The instability of a measure. The dataset shown in Fig. 3(a) is used here. $\varepsilon = 0.005$.

SNN, AG and IK		ℓ_p-norm for $p = 0.1, 0.5$ & 2	
$\mathbf{q}_i$: between 2 clusters	$\mathbf{q}_c$: cluster center	$\mathbf{q}_i$: between 2 clusters	$\mathbf{q}_c$: cluster center
			

The last two subfigures in Table 2 show a comparison between Euclidean distance and fractional distances ($\ell_p, p = 0.1, 0.5$). It is true that the fractional distances are better than Euclidean distance up to $d = 1000$, but they are still unstable for higher dimensions.

Summary. Many existing measures have indistinguishability in high dimensions that have prevented them from finding the nearest neighbor if the query is outside of any clusters. **IK has distinguishability** because IK has a unique t-dimensional feature map that has increased distinguishability as t increases, described in Section 3.2. No existing metric-based Lipschitz continuous kernels have a feature map with this property.

4.2. The impact of (in)distinguishability on four traditional tasks

We use the same ten datasets in the following four tasks. All have been used in previous studies (see Appendix D), except two synthetic datasets: Gaussians and ω-Gaussians. The former has two well-separated Gaussians, both on the same 10,000 dimensions; and the latter has two ω-dimensional Gaussians which overlap at the origin only, as shown in the figure in the first column in Table 5. Notice below that algorithms which employ distance and three other existing kernels have problems with the ω-Gaussians dataset in all four tasks, but they have no problems with the Gaussians dataset, despite they both have high input and intrinsic dimensions, i.e., high dimensions are not the only factor influencing a task-specific performance.

4.2.1. Exact nearest neighbor search using ball-tree indexing

Existing indexing techniques for exact NN search are sensitive to dimensionality, data size and structure of the dataset [30,31]. They work in low dimensional datasets of a moderate size only where a dataset has clusters.

Table 3 shows the comparison between the brute-force search and an indexed search based on ball-tree [25], in terms of runtime, using Euclidean distance, IK and linear kernel (LK). Using LK, the ball-tree index always takes longer than the brute-force search in all datasets, except the lowest dimensional dataset.[2] This is also true for distance (with one exception on the Gaussian dataset which we will further its discussion in Section 6.3).

Yet, with an appropriate value of ψ, IK yielded a faster search using the ball-tree index than the brute-force search in all datasets, without exception. Note that the comparison using IK is independent of the feature mapping or preprocessing time of IK because both the brute-force and the ball-tree indexing require the same feature mapping time (shown in the 'Map' column in Table 3).

It is interesting to note that the feature map of IK uses an effective number of dimensions of $t = 200$ in all experiments. While one may tend to view this as a dimension reduction method, it is a side effect of the kernel, not by design. The key to IK's success

[2] See an additional comparison result on a large high dimensional dataset in Appendix E.

Table 3
Runtimes of exact 5-nearest-neighbors search: brute-force search vs ball-tree index. Boldface indicates faster runtime between brute-force and index, or better precision. The 'Map' column indicates the preprocessing of IK to produce its feature map. The same feature map is required for both brute-force search and ball-tree index. The experimental details are in Appendix D. The Gaussians dataset is as used in Fig. 3; w-Gaussians (where $w = 5,000$) is shown in Table 5. The last two columns show the retrieval precision of 5 nearest neighbors. Every point in a dataset is used as a query; and the reported result is an average over all queries in the dataset. $\#C$ denotes the number of classes.

Dataset	*#points*	*#dimen.*	$\#C$	Distance		IK			LK		Precision@5	
				Brute	Balltr	Brute	Balltr	Map	Brute	Balltr	ℓ_2	IK
Url	3,000	3,231,961	2	**38,702**	41,508	112	**109**	9	**39,220**	43,620	.88	**.95**
News20	10,000	1,355,191	2	**101,243**	113,445	3,475	**2,257**	16	**84,986**	93,173	.63	**.88**
Rcv1	10,000	47,236	2	**3,615**	4,037	739	**578**	11	**5,699**	6,272	.90	**.94**
Real-sim	10,000	20,958	2	**2,541**	2,863	4,824	**4,558**	21	**2,533**	2,853	.50	**.90**
Gaussians	2,000	10,000	2	46	**41**	229	**213**	46	**45**	54	**1.00**	**1.00**
w-Gaussians	2,000	10,000	2	**53**	77	210	**205**	45	**56**	73	.90	**1.00**
Cifar-10	10,000	3,072	10	**340**	398	7,538	**7,169**	67	**367**	439	.25	**.27**
Mnist	10,000	780	10	**58**	72	1,742	**1,731**	2	**87**	106	**.93**	**.93**
A9a	10,000	122	2	**10**	13	5,707	**5,549**	1	**12**	16	**.79**	**.79**
Ijcnn1	10,000	22	2	2.3	**1.3**	706	**654**	1	2.2	**1.8**	**.97**	.96
Average											.77	**.86**

in enabling efficient indexing is the kernel characteristic in high dimensions (in Hilbert space). IK is not designed for dimension reduction.

Note that IK works not because it has used a low-dimensional feature map. Our result in Table 3 is consistent with the previous evaluation (up to 80 dimensions only [31]), i.e., indexing methods ran slower than brute-force when the number of dimensions is more than 16. If the IK's result is due to a low dimensional feature map, it must be about 20 dimensions or less for IK to work. But all IK results in Table 3 have used $t = 200$.

Nevertheless, the use of $t = 200$ effective dimensions has an impact on the absolute runtime. Notice that, when using the brute-force method, the actual runtime of IK was one to two orders of magnitude shorter than that of distance in $d > 40,000$; and the reverse is true in $d < 40,000$. We thus recommend using IK in high dimensional ($d > 40,000$) datasets, not only because its indexing runs faster than the brute-force search, but IK runs significantly faster than distance too. In $d \leq 40,000$ dimensional datasets, distance is preferred over IK because the former runs faster, if runtime is the only concern.

The last two columns in Table 3 show the retrieval result in terms of precision of 5 nearest neighbors. It is interesting to note that IK has better retrieval outcomes than distance in all cases, where large differences can be found in News20, Realsim and w-Gaussians. The only exception is Ijcnn1 which has the lowest number of dimensions, and the difference in precision is small.

This shows that the NNHD problem influences not only the indexing runtime but also the precision of retrieval outcomes.

4.2.2. *Anomaly detection using kernel density estimators*

Kernel density estimators (KDE) are known to perform poorly in high dimensions (see e.g., [32,33]). Here we examine whether the use of IK in KDE will change this long-established understanding.

We use the ordinary KDE (OKDE), KDEOS [26] and three recent fast versions that have linear time complexity, i.e., RACE [34], HBE [35] and `IKDE` [36]. RACE and HBE use locality-sensitive hashing to calculate the kernel summary of a data independent kernel. RACE employs the p-stable LSH kernel (LSHK) for Euclidean distance [37], and HBE employs Laplacian kernel (LapK) (as used in their papers [35,34]). `IKDE` employs IK instead of a data independent kernel. OKDE employs Gaussian kernel (GK); and KDEOS employs an adaptive Gaussian kernel (AGK) that replaces the bandwidth setting with kNN distance such that it is adaptive to local density. Out of the five estimators, only `IKDE` and KDEOS use an adaptive kernel. Note that the key difference among these estimators is the kernel employed, as they all use the same KDE formulation (see the details in Appendix F).

Using KDE for anomaly detection is straightforward: a point, which is estimated to have low density, is declared to be a likely anomaly.

Table 4 shows the anomaly detection accuracy in terms of area under ROC curve (AUC).

It is interesting to note that `IKDE` using IK has significantly higher detection accuracy than at least two other KDEs in all datasets with more than 100 dimensions.[3] Note that each of OKDE, KDEOS, RACE and HBE perform poorly in at least one of the two artificial datasets, i.e., Gaussians and w-Gaussians. In addition, RACE has the worst accuracies in seven out of the ten datasets in which it could complete the runs within 2 days.

Note that the Mnist_230 dataset has anomalies that can be detected easily by all estimators. But, on the w-Gaussians dataset in which anomalies are hard to detect, OKDE, RACE and HBE performed significantly poorer than `IKDE` and KDEOS. Yet, KDEOS and HBE performed significantly worse than others on the easy Gaussians dataset. Only `IKDE` performed consistently well on ten datasets.

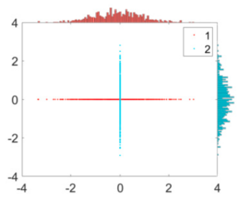
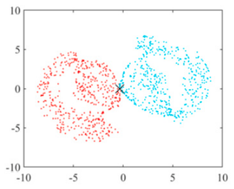
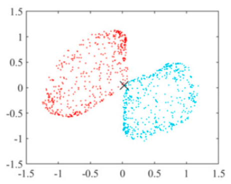
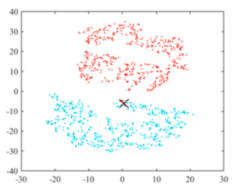
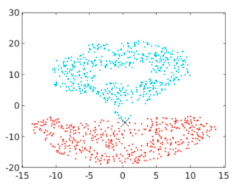
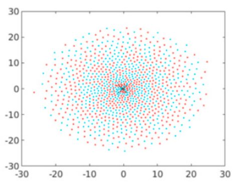
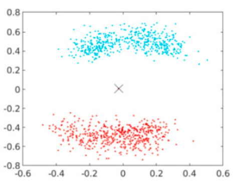
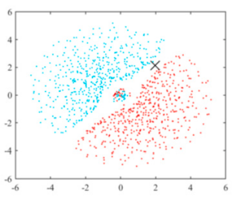
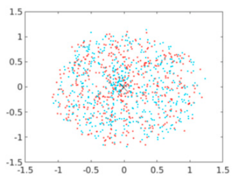
These results of OKDE, RACE, HBE and KDEOS come as no surprise because it is common knowledge that the KDE which employs a commonly used kernel does not perform well in high dimensions (see e.g., [32,33]). Contrary to this long-established understanding,

[3] See an extensive comparison of these KDEs on low dimensional datasets in [36]. Low dimensions are outside the scope of this paper.

Table 4
Anomaly detection using kernel density estimators of different kernels. The anomaly detection accuracy is in terms of AUC. > 2d denotes that the algorithm could not complete in less than 2 days. The second row indicates the kernel used in each KDE.

Dataset	#points	#dimensions	#Anomalies	OKDE	KDEOS	RACE	HBE	IKDE
				GK	AGK	LSHK	LapK	IK
Url	1,615	3,231,961	20	> 2d	> 2d	.67	> 2d	**.73**
News20	9,997	1,355,191	300	> 2d	> 2d	> 2d	> 2d	**.67**
Rcv1	10,596	47,236	105	> 2d	> 2d	.55	.72	**.78**
Real-sim	40,457	20,958	400	> 2d	> 2d	.53	> 2d	**.86**
Gaussians	1,000	10,000	10	**1.00**	.50	.93	.69	**1.00**
w-Gaussians	1,000	10,000	10	.62	**.98**	.72	.73	.93
Cifar-10	45,450	3,072	450	> 2d	> 2d	.57	> 2d	**.63**
Mnist_479	12,139	784	50	.69	**.86**	.54	.57	.69
Mnist_230	12,117	784	10	**.97**	**.97**	.90	.94	**.97**
A9a	24,720	123	500	.47	.53	.56	.49	**.59**
Ijcnn1	45,137	22	500	.78	.72	.65	**.80**	.72

Table 5
t-SNE: Gaussian Kernel versus Isolation Kernel on the w-Gaussians datasets having two w-dimensional subspace clusters (shown in the first column). Both clusters are generated using $\mathcal{N}(0,1)$. "×" is the origin—the only place in which the two clusters overlap in the data space.

$w = 1$ dataset		t-SNE(GK)	t-SNE(IK)	FIt-SNE	q-SNE
	$w = 2$				
	$w = 5000$				

we show that Isolation Kernel is a game-changer that enables KDE to deal with high dimensional datasets. This is consistent with the analytical result we have provided in Section 3.2.

Another standout observation in Table 4 is that IKDE is the only estimator that could complete the runs in all datasets within 2 days. The next contender in this respect is RACE, i.e., it could not complete the run within 2 days on the News20 dataset only. OKDE, KDEOS and HBE could not complete in at least four datasets within the same time constraint.

4.2.3. *Visualization using t-SNE*

Studies in visualization [38,14] often ignore the issues of the curse of dimensionality which raise doubt about the assumption made by visualization methods. For example, t-SNE [14] employs Gaussian kernel as a means to measure similarity in the high dimensional data space. No studies have examined the effect of its use in the presence of the curse of dimensionality, as far as we know. Here we show one example misrepresentation from t-SNE, due to the use of GK to measure similarity in the high dimensional data space on the w-Gaussians datasets.

The t-SNE visualization results comparing Gaussian kernel and Isolation Kernel are shown in the middle two columns in Table 5. While t-SNE using GK has preserved the structure in low dimensional ($w = 2$) data space, it fails completely to separate the two clusters in high dimensional ($w = 5,000$) data space. In contrast, t-SNE with IK correctly separates the two clusters in both low and high dimensional data spaces.

Recent improvements on t-SNE, e.g., FIt-SNE [39] and q-SNE [40], explore better local structure resolution or efficient approximation. As they employ Gaussian kernel, they also produce misrepresented structures, as shown in Table 5.

Table 6 shows the results on a variant w-Gaussians dataset where the two clusters have different variances. Here t-SNE using GK produces misrepresented structure in both low and high input dimensions. It has misrepresented the origin in the data space as belonging to the red cluster only, i.e., no connection between the two clusters. Note that, using GK, all points in the red cluster are concentrated at the origin in the transformed space for $w = 100$. There is no such issue with IK.

In low dimensional ($w = 2$) input data space, this misrepresentation is due solely to the use of a data independent kernel. To make GK adaptive to local density in t-SNE, the bandwidth is learned locally for each point. As a result, the only overlap between the two

Table 6
t-SNE: GK versus IK on w-Gaussians datasets having two w-dimensional subspace clusters (different variances). The red and blue clusters are generated using $\mathcal{N}(0,1)$ and $\mathcal{N}(0,32)$, respectively. "×" is the origin—the only place in which the two clusters overlap in the data space.

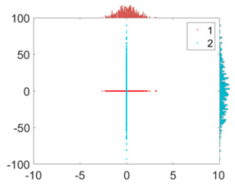
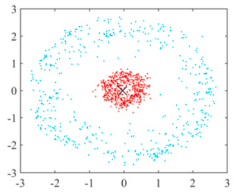
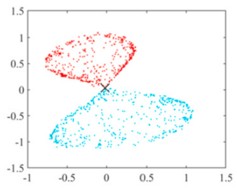
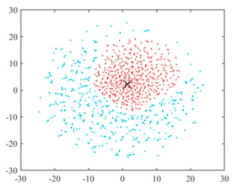
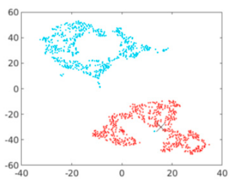
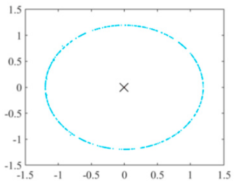
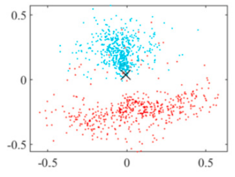
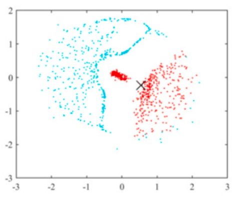
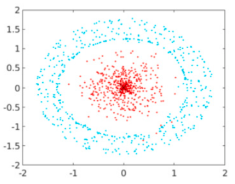
$w = 1$ dataset		t-SNE(GK)	t-SNE(IK)	FIt-SNE	q-SNE
	$w = 2$				
	$w = 100$				

Table 7
SVM classification accuracy & runtime (CPU seconds). Linear kernel (LK), IK and Gaussian kernel (GK). $nnz\% = \#nonzero\_values/((\#train + \#test) \times \#dimensions) \times 100$. IK ran five trials to produce [mean]± [standard error]. IK mapping time is shown in Table D.11 in Appendix D. For example, even including feature mapping time on the Cifar-10 dataset, IK took a total of 1087 seconds which is still one order magnitude faster than GK. Cifar-10 is a dense dataset (with $nnz\% = 99.8$) and has complex concepts. This is the reason why SVM took longest to complete, i.e., SVM took significantly more iterations than on other datasets.

Dataset	*#train*	*#test*	*#dimensions*	*nnz%*	Accuracy			Runtime		
					LK	IK	GK	LK	IK	GK
Url	2,000	1,000	3,231,961	.0036	**.98**	**.98**±.001	**.98**	2	1	14
News20	15,997	3,999	1,355,191	.03	.85	**.92**±.007	.84	38	19	528
Rcv1	20,242	677,399	47,236	.16	.96	.96±.013	**.97**	111	26	673
Real-sim	57,848	14,461	20,958	.24	**.98**	**.98**±.010	**.98**	49	13	2114
Gaussians	1,000	1,000	10,000	100.0	**1.00**	**1.00**±.000	**1.00**	14	0.5	78
w-Gaussians	1,000	1,000	10,000	100.0	.49	**1.00**±.000	.62	20	0.5	79
Cifar-10	50,000	10,000	3,072	99.8	.37	**.56**±.022	.54	3,808	493	29,322
Mnist	60,000	10,000	780	19.3	.92	.96±.006	**.98**	122	17	598
A9a	32,561	16,281	123	11.3	**.85**	**.85**±.012	**.85**	1	22	100
Ijcnn1	49,990	91,701	22	59.1	.92	.96±.006	**.98**	5	40	95

subspace clusters, i.e., the origin, is assigned to the dense cluster only. The advantage of IK in $w = 2$ is because it is data dependent kernel (see Section 5.1 for more details).

In high dimensional ($w = 100$) input data space, the effect due to the curse of dimensionality obscures the similarity measurements made. Both this high-dimension effect and the data independent effect collude in $w = 100$ when distance-based measures are used, leading to poor result in the transformed space.

4.2.4. *SVM classification*

Table 7 shows the comparison between linear kernel, IK and Gaussian kernel using an SVM classifier in terms of accuracy and runtime. The experimental details are in Appendix D.
It is interesting to note that IK produced consistently high (or the highest) accuracy in all datasets. In contrast, both linear and Gaussian kernels have a mixed bag of low and high accuracies. For example, they both perform substantially poorer than IK on News20 and w-Gaussians (where $w = 5,000$.) Recall that each w-Gaussians dataset has two w-dimensional subspace clusters. The $w = 1$ version of the dataset is shown in the first column in Table 5.
The previous work on IK and its impact on SVM [7] has revealed that IK is better than Gaussian and Laplacian kernels on datasets with varied densities. We postulate that the small difference in accuracy on some datasets, shown in Table 7, suggests that these datasets have no huge varied densities; therefore, IK's data dependent property has no impact. In other words, SVM with Gaussian kernel is robust to high dimensional datasets if they do not have the data characteristic as shown in w-Gaussians or/and varied densities. Examples are the Gaussians, Url, Rcv1 and Real-sim datasets, where the clusters in each dataset are pretty well separated. Recall (in Section 2.1) that the indistinguishability/unstable/meaningless issue occurs in distinguishing different points within one cluster, but there is no such issue in differentiating points from well-separated clusters.
The details of the data dependent property and its relationship with the distinguishability property are provided in Section 5.1.

4.2.5. *Further investigation using a low dimensional w-Gaussians ($w = 2$) dataset*

It is interesting to note when an w-Gaussians ($w = 2$) dataset is used, (i) all three measures used in Table 3 enable faster indexed search than the brute-force search and have equally high precision; (ii) all kernels used in the KDEs shown in Table 4 produce high accuracies (i.e., all estimators produce AUC = 1, except HBE which yields AUC = 0.99); and (iii) all the three kernels (LK, IK and GK) used in SVM shown in Table 7 produce perfect accuracy of 1. But only IK can do well in the high dimensional w-Gaussians ($w \geq 100$) in all four tasks, as shown in Tables 3 to 7.

The w-Gaussians datasets provide an example condition in which existing measures performed well in low dimensions but poorly in high dimensions in indexed search, KDE anomaly detection, t-SNE visualization and SVM classification. This is a manifestation that the existing measures used in these algorithms have indistinguishability in high dimensions.

Section Summary:

- IK is the only measure in our experiment that consistently (a) enabled faster indexed search and more precise retrieval search outcomes than the brute-force search in high dimensions for exact nearest neighbor search, (b) provided higher accuracy in KDE anomaly detection than existing KDEs, (c) yielded matching structures before and after t-SNE transformation; and (d) produced high accuracy in SVM classification in high and low dimensions.
- Euclidean distance or existing kernels performed poorly in most high dimensional datasets in all tasks. We show one condition, using the artificial w-Gaussians datasets, in which they did well in low dimensions but performed poorly in high dimensions in all four tasks. The Gaussians dataset is one condition in which Euclidean distance or existing kernels can do well in high dimensions (see Section 6.3 for further discussion).

5. Other characteristics of IK

5.1. *IK's data dependent property relies on adaptive partitionings but not its distinguishability*

IK has been previously shown to be better than Gaussian and Laplacian kernels in SVM classification [7], better than Euclidean distance in density-based clustering [8], and better than Gaussian kernel in kernel-based anomaly detection [20].

IK's superiority has been attributed to its data dependent similarity, given in the following lemma.

Let $D \subset \mathcal{X}_D \subseteq \mathbb{R}^d$ be a dataset sampled from an unknown distribution $\mathcal{P}_D$ where $\mathcal{X}_D$ is the support of $\mathcal{P}_D$ on $\mathbb{R}^d$; and let $\rho_D(\mathbf{x})$ denote the density of $\mathcal{P}_D$ at point $\mathbf{x} \in \mathcal{X}_D$, and ℓ_p be the ℓ_p-norm. The unique data dependent characteristic of Isolation Kernel [7,8] is:

Lemma 3. *Isolation Kernel K_ψ has the data dependent characteristic:*

$$K_\psi(\mathbf{x}, \mathbf{y} \mid D) > K_\psi(\mathbf{x}', \mathbf{y}' \mid D) \equiv P(\mathbf{x}, \mathbf{y} \in \theta) > P(\mathbf{x}', \mathbf{y}' \in \theta)$$

for $\ell_p(\mathbf{x} - \mathbf{y}) = \ell_p(\mathbf{x}' - \mathbf{y}')$, $\forall \mathbf{x}, \mathbf{y} \in \mathcal{X}_\mathsf{S}$ *and* $\forall \mathbf{x}', \mathbf{y}' \in \mathcal{X}_\mathsf{T}$ *subject to* $\forall \mathbf{z} \in \mathcal{X}_\mathsf{S}, \mathbf{z}' \in \mathcal{X}_\mathsf{T},\ \rho_D(\mathbf{z}) < \rho_D(\mathbf{z}')$.

As the partitionings are adaptive to local density, partition θ is large (small) if the probability distribution around $\mathbf{x}$ and $\mathbf{y}$ is sparse (dense). Thus, $\mathbf{x}$ and $\mathbf{y}$ have a larger chance to be in the large partition in sparse region than $\mathbf{x}'$ and $\mathbf{y}'$ (of equal inter-point distance) have in the small partition in dense region.

As a result, $K_\psi(\mathbf{x}, \mathbf{y} \mid D)$ is not translation invariant, unlike the data independent Gaussian and Laplacian kernels.

Yet, the proof of Lemma 2 on IK's distinguishability relies on a condition of the partitionings that any point $\mathbf{x}$ has equal probability of being in partition θ_j out of ψ partitions, i.e., $P(\mathbf{x} \in \theta_j) = 1/\psi$ (Remark 1); but it does not rely on the partitions being adaptive to local density.

These differences are demonstrated in Section 4.2.3 via t-SNE, where the impact due solely to the property of IK's distinguishability is shown in Table 5 (on the $w = 5000$ dataset, where the two clusters have the same density), and that due to the data dependent property of IK is illustrated in Table 6 (on the $w = 2$ dataset, where the two clusters have different densities).

5.2. *Do other implementations of IK have distinguishability?*

In addition to Voronoi Diagram, two other (currently known) implementations of IK are Isolation Forest [7] and Isolating hyperspheres [20].

As stated earlier, a necessary condition to have the distinguishability property is that any point $\mathbf{x}$ has equal probability of being in partition θ_j out of ψ partitions in the entire data space. Because the Isolation Forest and Isolating hyperspheres implementations of IK do not satisfy this condition, Remark 1 and Lemma 2 are not applicable to these two partitioning methods.

However, this does not mean that an alternative Lemma does not exist, which is applicable to these two implementations, in order to show their (in)distinguishability.

Nevertheless, we have conducted an empirical comparison of all three implementations of IK in two tasks. In retrieval task, all three implementations have comparable performance, though the Voronoi Diagram implementation usually has slightly better performance than the other two implementations. In anomaly detection task, the Isolating hyperspheres implementation usually has better detection accuracy than the other two implementations. The Isolation Forest implementation is the weakest in general in these two tasks because it selects a subset of attributes only to build the trees. In high dimensions, this is often insufficient to distinguish two different points. The detailed results of these two tasks are given in Appendix H.

It is instructive to note that some points are outside the coverage areas of Isolating hyperspheres. This means that these outlying points are not retrievable in a retrieval task; but the Isolating hyperspheres are perfect in anomaly detection task in detecting these outliers.

In a nutshell, these empirical results and analyses suggest that the Isolating hyperspheres implementation of IK may have the distinguishability property. Its formal analysis is an interesting future work.

5.3. *What about other space partitioning methods?*

Note that not all space partitioning methods work for IK. Any IK implementation shall have the data dependent property (mentioned in Section 5.1) and the distinguishability property, i.e., the partitioning must produce small partitions in dense regions and large partitions in sparse regions, and any point $\mathbf{x}$ has equal probability of being in partition θ_j out of ψ partitions.
We can now examine whether a partitioning is suitable for IK implementation based on these criteria.

6. Discussion

6.1. *Issues in the curse of dimensionality*

We have limited our discussion on the curse of dimensionality based on nearest neighbors, i.e., the NNHD problem (stated in Section 1) in terms of the (in)distinguishability of a kernel. But the effect of the curse is broader than this. For example, it has effects on the consistency of estimators, an aspect not discussed in this article.

In addition to the concentration effect [2,3], the curse of dimensionality has also been studied under different phenomena, e.g., the correlation between attributes [41] and hubness [4]. The result of an investigation on the hubness effect [4] can be found in Appendix G. It would be interesting to analyze whether IK can also deal with these issues effectively in high dimensions.

Our results in Tables 3 and 4 show that some existing measures may perform well in high dimensions under special conditions. However, we are not aware of analyses that examine whether any existing measures have **distinguishability**.

Some papers claim to have overcome the curse of dimensionality (e.g., [42–47]). These are often applicable to specific methods with either limited empirical supports or scarce/no theoretical supports. For example, Hendrikse et al. [42] and Cabannes et al. [46] propose methods to overcome the curse that have desirable theoretical properties in a semi-supervised learning method that employs Laplacian regularization and a parametric likelihood-ratio-based verification scheme, respectively. But they provide scarce empirical evidence to support the theories.

In another paper, Sarkar and Ghosh [45] introduce a new measure that takes advantage of the concentration property. Though the measure performs poorly in low dimensions, it works well in high dimensions in three existing clustering algorithms on some artificial datasets. However, the applications of the proposed measure MADD are limited for two reasons. First, MADD is a semi-metric. As a result, MADD could not be used for ball-tree indexed search which requires a metric. Second, its time complexity to compute a distance matrix is $O(n^3)$. The authors suggest that a block method can improve the time complexity, but it is unclear how this can be done, and the authors consider that as a future work.

In yet other papers, they provide some empirical evidence for some specific methods with no or insufficient theoretical assurances. Examples are: (a) a hierarchical Bayesian model is proposed to solve the problem of multi-view clustering in high dimensions [43]; (b) a distributed architecture is proposed to deal with gene-related datasets [44]; and (c) the generalization performance of neural networks [47].

6.2. *Relation to the concentration effect and blessing of dimensionality*

Note that the NNHD problem is a special case of the concentration effect which deals with the ability to find the single nearest neighbor in high dimensions specifically.

The concentration effect has a broader implication which includes the blessing of dimensionality [1]; and the latter refers to the “constant” nature of many independent dimensions [2] which can be exploited to aid mathematical analysis in high dimensions, even though the analysis is hard in a low dimensional context where the “constant” nature is non-existence.

In terms of addressing the NNHD problem under investigation in this paper, we are not aware of any existing work in which the blessing or the concentration effect can be of any help.

6.3. *Intrinsic dimensions*

A previous work on intrinsic dimensionality [18] has shown that nonparametric distance-based regressors could escape the curse of dimensionality if a high dimensional dataset has low intrinsic dimensions. This is consistent with the finding that the concentration effect depends more on the intrinsic dimensions I_d than on the input dimensions $d > I_d$ [16].

An earlier work has specifically referred the curse to the intrinsic dimensions rather than the input dimensions: ‘a high intrinsic dimension of a dataset reflects the presence of the curse of dimensionality’ [48]. Thus, one can only find ways to mitigate the curse of dimensionality in a dataset with high intrinsic dimensionality (e.g., via a distribution of distance [49]).

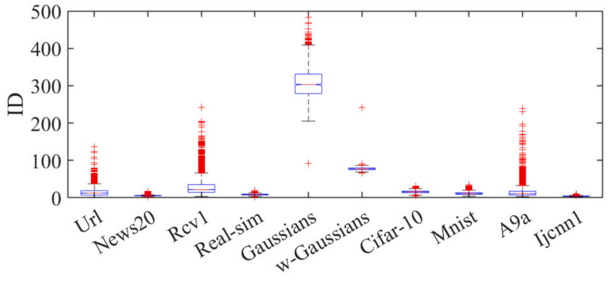


Fig. 5. Estimated IDs for 10 datasets using TLE with $k = 50$.

Fig. 5 shows that intrinsic dimensions (IDs) as estimated by a recent local ID estimator TLE [50].

One can use these estimations to explain why the ball-tree indexed search failed or succeeded using the distance or linear kernel to gain the expected speedup in almost all datasets shown in Table 3. It failed in all datasets which have more than 5 IDs; and it succeeded in Ijcnn1 which has 4 IDs. This is consistent with the understanding based on intrinsic dimensions.

The only exception is in the Gaussians dataset which is estimated to have 300 IDs, where the ball-tree indexed search using the distance has succeeded, outside the expectation.

IK is unique because the distinguishability of IK is independent of dimensionality and data distribution, and IK does not rely on low intrinsic dimensions to perform well, including both the Gaussians and w-Gaussians datasets which are estimated to have the highest intrinsic dimensions in Fig. 5. This occurs in all four tasks shown in Tables 3 to 7.

6.4. *Relation to locality-sensitive hashing and approximate nearest neighbor search*

On the surface, each isolating partition of Isolation Kernel (IK) may be regarded as a 'bucket' used in locality-sensitive hashing (LSH). But this is where the resemblance ends.[4] Recall that the partitions in IK are used to compute the similarity between two points, not identifying a bucket of points which is similar to a query.

IK is neither an indexing method such as LSH nor a kernelized LSH method. IK is a kernel, on par with a distance function used in an indexing method. As shown in Section 4.2.1 and Table 3, IK can be used in an indexing method replacing a distance function.

A kernelized LSH method [51] generalizes LSH to be applicable to a kernel's feature space where the feature map is implicit or incomputable. The feature map of IK is explicit and computable.

The central issue in this paper is the NNHD problem. Existing LSH works aim to provide an efficient *inexact* NN search after hashing the points to a low dimensional space, and none of the works have claimed to address the NNHD problem, i.e., the *exact* NN search. In fact, the efficiency of LSH degrades to random sampling or a linear search in high dimensions (see e.g., [49]).

Some approximate nearest neighbor (ANN) search algorithms have been shown to work well in high dimensions (see a comparison in [52]). Like LSH, they aim for inexact NN search only, and none of these ANN methods can say anything about the exact NN search or the distinguishability of the distance function employed in high dimensions.

To demonstrate their relative retrieve performance, we compare the retrieval precision of 5 nearest neighbors of exact NN searches using distance and IK with an approximation search using a kNN-graph method[5] which uses Euclidean distance to construct its kNN graph in the preprocessing before building its index.[6] This method is selected because kNN-graph based index has been shown to be one of the top performers [52].

The result in Table 8 shows that an approximate NN search produces worse retrieval outcomes than any of the two exact searches in four of the five datasets. This occurs for both low (Ijcnn1) and high dimensional datasets, as expected.[7]

This 5-nearest neighbor search result also shows that 1-nearest neighbor (1NN) search, which is the focus of discussion in this paper, is the basis for $k > 1$ nearest neighbor search (kNN). Without an accurate 1NN search, kNN search cannot be accurate.

[4] A previous work [36] on kernel density estimation using IK has compared and analyzed the differences with kernel-based LSH (i.e., RACE [34] and HBE [35], as we have used in Section 4.2.2): There are three key differences. First, IK is a *data dependent* kernel. But, an implementation of kernelized LSH rely on a *data independent* kernel to determine the collision probability of the hash scheme. Second, random projection is often a must-have intermediate process in LSH in order to produce the buckets; but not in IK where the partitions in the Voronoi Diagram are obtained for free after the ψ points have been selected (i.e., the Voronoi Diagram needs not be built explicitly). Third, the number of buckets in LSH has no relation to the kernel; but the number of 'buckets' (partitions) ψ is IK's kernel parameter, equivalent to the sharpness parameter of Gaussian or Laplacian kernel. Some details of RACE and HBE can be found in Appendix F.

[5] KGraph: A library for approximate nearest neighbor search, https://github.com/aaalgo/kgraph.

[6] Although the kNN-graph indexing method has fast retrieve time, it has a huge memory requirement for storing high-dimensional data. Only five out of eleven datasets could be completed using a machine with 32 GB RAM. It has out of memory errors on the other six datasets.

[7] Previous comparisons (e.g., [52]) between different approximate NN search methods focus on the recall of nearest neighbors, i.e., comparing the overlap of the identified nearest neighbors between an approximate NN search and its exact NN search using the same distance measure for a query point. However, different distance measures may produce different exact nearest neighbors of the same query point in a dataset. This kind of comparison has the correct assumption that the exact search produces better NNs than approximate search, which is consistent with our result, though based on a different experimental setting. In our experiments, we report the precision of the nearest neighbors based on the class labels given in a given dataset, i.e., the percentage of the 5 nearest neighbors having the same class label of a query point, averaged over all points in a dataset.

Table 8
Approximate NN search versus exact NN search: The retrieval precision of 5 nearest neighbors. Every point in a dataset is used as a query; and the reported result is an average over all queries in each dataset.

	Approximate NN search	Exact NN search	
	kNN-graph	ℓ_2 distance	IK
Rcv1	0.51	0.90	0.94
Gaussians	1.00	1.00	1.00
w-Gaussians	0.79	0.90	1.00
A9a	0.64	0.79	0.79
Ijcnn1	0.81	0.97	0.96

6.5. *Other issues*

Our applications in ball-tree indexing for exact NN search, KDE anomaly detection and t-SNE here, and previously in DBSCAN clustering [8] and multi-instance learning [21], suggest that many existing algorithms can get performance improvement by simply replacing distance/kernel with IK. However, a caveat is in order: not all existing algorithms can achieve that. For example, OCSVM [53] and OCSMM [54] have been shown to work poorly with IK [55]. This is because the learning in these two algorithms is designed to have a certain geometry in Hilbert space in mind; and IK does not have such a geometry (see the details in [55]).

The isolating mechanism used has a direct influence on IK's distinguishability. Note that the proof of Theorem 2 relies on the isolating partitions being the Voronoi Diagram. It remains an open question whether IK with a different implementation could be proven to have distinguishability.

Kernel functional approximation is an influential approach to obtain an approximate finite-dimensional feature map from an infinite-dimensional feature map of a kernel such as Gaussian kernel. Its representative methods are Nyström [56] and random features [57,58]. The former is often dubbed data dependent and the latter data independent. These terms refer to the use (or no use) of data samples to derive the approximate feature map. In either case, the kernel employed is data independent and translation invariant, unlike the data dependent IK in which the similarity depends on local data distribution and it is not translation invariant.

Two recent works [59,36] have already shown that IK improves the effectiveness and efficiency of t-SNE and KDE on low dimensional datasets, by simply replacing Gaussian kernel. However, they did not address the issue related to the NNHD problem, which is the focus of this paper.

7. Conclusions

We study the open NNHD problem in the context of curse of dimensionality: is it possible to find the single nearest neighbor of a query in high dimensions?

We show that Isolation Kernel (IK) has **distinguishability**, addressing the NNHD problem for the first time. It is possible because (a) IK measures the similarity between two points based on the space partitionings produced from an isolation mechanism; (b) the probability of a point falling into an isolation partition is independent of data distribution, the distance measure used to create the partitions and the data dimensions, implied in Remark 1; and (c) IK's unique feature map has its dimensionality linked to a concatenation of these partitionings. Theorem 2 suggests that increasing the number of partitionings t (i.e., the dimensionality of the feature map) leads to increased distinguishability, independent of the number of dimensions and distribution in data space.

Isolation Kernel, with its feature map having the distinguishability stated in Theorem 2, is the key to consistently producing better task-specific performances in high dimensions in four tasks: (i) faster indexed search than brute-force search, and high retrieval precision in exact nearest neighbor search, (ii) higher detection accuracy in KDE anomaly detection than existing KDEs, (iii) matching structures before and after t-SNE transformation, and (iv) SVM classification.

We are not aware of any existing distance/kernel which can achieve the same outcomes. Euclidean distance, Gaussian and three other existing kernels produce largely poor results in datasets having high input dimensions as well as intrinsic dimensions in our experiments. This is not a surprising outcome of the curse of dimensionality, echoing the current state of understanding of these measures.

CRediT authorship contribution statement

Kai Ming Ting: Writing – review & editing, Writing – original draft, Validation, Supervision, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Takashi Washio:** Writing – original draft, Validation, Methodology, Formal analysis. **Ye Zhu:** Writing – review & editing, Visualization, Software, Resources, Project administration, Investigation, Formal analysis. **Yang Xu:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources. **Kaifeng Zhang:** Writing – review & editing, Validation, Methodology, Investigation, Formal analysis.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

We have provided all the codes and datasets used in this paper on GitHub: https://github.com/IsolationKernel/High-dimensional_Evaluation.

Acknowledgement

Kai Ming Ting is supported by the National Natural Science Foundation of China (Grant No. 62076120).

Appendix A. A guide for parameter setting of IK

Some advice on parameter setting is in order when using IK in practice. First, t can often be set as default to 200 initially. Then, search for the 'right' ψ for a dataset. This is equivalent to searching the bandwidth parameter for Gaussian kernel. Once the ψ setting has been determined, one may increase t to examine whether a higher accuracy can be achieved (not attempted in our experiments).

Second, finding the 'right' parameters for IK may not be an easy task for some applications such as clustering (so as indexing). This is a general problem in the area of unsupervised learning, not specific to the use of IK. For any kernel employed, when no or insufficient labels are available in a given dataset, it is unclear how an appropriate setting can be found in practice.

Appendix B. Proofs of lemmas and theorems

Given a data set $D = \{\mathbf{x}_i \in \mathbb{R}^d \mid i = 1, \dots, n\} \sim F^n$, where $\mathbf{x}_i$ is i.i.d. drawn from any probability distribution F on Ω. Let the feature vectors of two distinct $\mathbf{x}_a$ and $\mathbf{x}_b$ be $\phi(\mathbf{x}_a) = [\kappa(\mathbf{x}_a, \mathbf{x}_1), \dots, \kappa(\mathbf{x}_a, \mathbf{x}_n)]^\top$ and $\phi(\mathbf{x}_b) = [\kappa(\mathbf{x}_b, \mathbf{x}_1), \dots, \kappa(\mathbf{x}_b, \mathbf{x}_n)]^\top$ in a feature space $\mathcal{H}_\kappa$.

Lemma 1. *$\mathcal{H}_\kappa$ has the following property for two distinct points $\mathbf{x}_a$ and $\mathbf{x}_b$ i.i.d. drawn from F on Ω:*

$$P\left(\ell_p(\phi(\mathbf{x}_a), \phi(\mathbf{x}_b)) \le 2L\epsilon n^{1/p}\right) \ge (1 - 2\alpha(\epsilon))^{2n},$$

for any $\epsilon > 0$, where ℓ_p is an ℓ_p-norm on $\mathcal{H}_\kappa$, and L is the Lipschitz constant of f_κ.

Proof. *Since f_κ is continuous and monotonically decreasing for $m(\mathbf{x}, \mathbf{y})$, the following holds for every $\mathbf{x}_i$:*

$$\begin{aligned} A_\epsilon(\mathbf{x}_i) &= \{\mathbf{x} \in \Omega \mid f_\kappa(m(\mathbf{x}, \mathbf{x}_i)) \in [f_\kappa(M(\mathbf{x}_i) + \epsilon), f_\kappa(M(\mathbf{x}_i) - \epsilon)]\} \\ &= \{\mathbf{x} \in \Omega \mid m(\mathbf{x}, \mathbf{x}_i) \in [M(\mathbf{x}_i) - \epsilon, M(\mathbf{x}_i) + \epsilon]\} \end{aligned}$$

Thus, $F(A_\epsilon(\mathbf{x}_i)) \ge 1 - 2\alpha(\epsilon)$ holds for every $\mathbf{x}_i$ from Corollary 1.

Further, $P(\mathbf{x}_a,\ \mathbf{x}_b \in A_\epsilon(\mathbf{x}_i)) = F(A_\epsilon(\mathbf{x}_i))^2$ holds for every $\mathbf{x}_i$, since $\mathbf{x}_a$ and $\mathbf{x}_b$ are i.i.d. drawn from F, and $\mathbf{x}_i$ is i.i.d. drawn from F. Therefore, the following holds:

$$\begin{aligned} P(\mathbf{x}_a, \mathbf{x}_b \in A_\epsilon(\mathbf{x}_i) \text{ for all } i = 1, \dots, n) &= \Pi_{i=1}^n P(\mathbf{x}_a, \mathbf{x}_b \in A_\epsilon(\mathbf{x}_i)) \\ &= \Pi_{i=1}^n F(A_\epsilon(\mathbf{x}_i))^2 \\ &\ge (1 - 2\alpha(\epsilon))^{2n} \end{aligned}$$

Accordingly, the following holds:

$$\begin{aligned} P\left(\ell_p(\phi(\mathbf{x}_a) - \phi(\mathbf{x}_b)) \le \left(\sum_{i=1}^n |f_\kappa(M(\mathbf{x}_i) - \epsilon) - f_\kappa(M(\mathbf{x}_i) + \epsilon)|^p\right)^{1/p}\right) \\ \ge (1 - 2\alpha(\epsilon))^{2n} \end{aligned}$$

Moreover, the following inequality holds from the Lipschitz continuity of f_κ.

$$|f_\kappa(M(\mathbf{x}_i) - \epsilon) - f_\kappa(M(\mathbf{x}_i) + \epsilon)| \le L|(M(\mathbf{x}_i) + \epsilon) - (M(\mathbf{x}_i) - \epsilon)| = 2L\epsilon$$

The last two inequalities derive Lemma 1. □

Theorem 1. *Under the same condition as Lemma 1, the feature space $\mathcal{H}_\kappa$ has indistinguishability using ℓ_p-metric on $\mathcal{H}_\kappa$.*

Proof. *Given a constant $\beta > 0$, by choosing d as $d \ge \beta/\epsilon^2$, we obtain the following inequalities from Lemma 1.*

$$\alpha(\epsilon) = C_1 e^{-C_2\epsilon^2 d} \le C_1 e^{-C_2\beta}, \text{ and}$$
$$P\left(\ell_p(\phi(\mathbf{x}_a) - \phi(\mathbf{x}_b)) \le 2L\epsilon n^{1/p}\right) \ge (1-2\alpha(\epsilon))^{2n} \ge \xi,$$

where $\xi = (1-2C_1e^{-C_2\beta})^{2n} \in (0,1)$ *which is a constant. By casting* ℓ_p *and* $2L\epsilon n^{1/p}$ *to* m *and* e *in Definition 3, respectively, there is no* $\delta(<\xi)$ *holding Eq. (3) for any condition* v *in Definition 3. Accordingly,* $\mathcal{H}_\kappa$ *has indistinguishability.* □

Let the feature vectors of two distinct $\mathbf{x}_a$ and $\mathbf{x}_b$ be $\Phi(\mathbf{x}_a)$ and $\Phi(\mathbf{x}_b)$ in a feature space $\mathcal{H}_K$ associated with Isolation Kernel K, implemented using the Voronoi Diagram (as given in Definition 2) using $\mathcal{D}_i = \{\mathbf{z}_1, \dots, \mathbf{z}_\psi\} \sim F^\psi$ where each $\mathbf{z}$ is i.i.d. drawn from any probability distribution F on Ω.

Lemma 2a. *For a distribution* F *with a bounded open domain* $dom(F)$*, and any* $\mathbf{x}_a, \mathbf{x}_b$ *that sampled from* F*,* $\mathbf{x}_a \ne \mathbf{x}_b$*,* $\forall\epsilon > 0$*, there exists a* ψ*, such that Isolation Kernel* $K_\psi(\mathbf{x}_a, \mathbf{x}_b \mid D) < \epsilon$.

Proof. The set of ψ *seeds is denoted as* $\mathcal{D} = (\mathbf{z}_1, \mathbf{z}_2, ..., \mathbf{z}_\psi)$*. If* $\mathbf{z}_i (i \in [\psi])$ *is the nearest neighbor of* $\mathbf{x}$ *in* $\mathcal{D}$*, then we denote this event as* $(\mathbf{z}_i, \mathbf{x})_{\mathcal{D}}$*. A ball centered at* x *with radius* r *is denoted as* $B(\mathbf{x}, r)$*.* $\partial dom(F)$ *is the boundary of* F*'s domain* $dom(F)$*. For two points* $\mathbf{x}, \mathbf{y}$*,* $\mathbf{d}(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2$ *is Euclidean distance between* $\mathbf{x}$ *and* $\mathbf{y}$*. For a point* $\mathbf{x}$ *and a set* Y*,* $\mathbf{d}(\mathbf{x}, Y) = \min_{\mathbf{y}\in Y} \|\mathbf{x} - \mathbf{y}\|_2$.

By the definition of Isolation Kernel and Remark 1, it holds that

$$K_\psi(\mathbf{x}_a, \mathbf{x}_b) = \sum_{i=1}^{\psi} P((\mathbf{z}_i, \mathbf{x}_a)_{\mathcal{D}} \cap (\mathbf{z}_i, \mathbf{x}_b)_{\mathcal{D}}) = \sum_{i=1}^{\psi} P((\mathbf{z}_i, \mathbf{x}_a)_{\mathcal{D}}) P((\mathbf{z}_i, \mathbf{x}_b)_{\mathcal{D}} | (\mathbf{z}_i, \mathbf{x}_a)_{\mathcal{D}})$$
$$= \frac{1}{\psi} \sum_{i=1}^{\psi} P((\mathbf{z}_i, \mathbf{x}_b)_{\mathcal{D}} | (\mathbf{z}_i, \mathbf{x}_a)_{\mathcal{D}}).$$

Assume that ψ *is large enough to ensure that* $\mathcal{D} \cap B(\mathbf{x}_a, \mathbf{d}(\mathbf{x}_a, \partial dom(F))) \ne \emptyset$ *and* $\mathcal{D} \cap B(\mathbf{x}_b, \mathbf{d}(\mathbf{x}_b, \partial dom(F))) \ne \emptyset$*. Then if* $\mathbf{z}_i$ *is* $\mathbf{x}_a$*'s nearest neighbor in* $\mathcal{D}$*,* $\mathbf{z}_i \in B(\mathbf{x}_a, \partial dom(F))$ *and* $B(\mathbf{x}_a, \mathbf{d}(\mathbf{x}_a, \mathbf{z}_i)) \subset B(\mathbf{x}_a, \mathbf{d}(\mathbf{x}_a, \partial dom(F))) \subset dom(F)$*. Then we have*

$$P((\mathbf{z}_i, \mathbf{x}_b)_{\mathcal{D}} | (\mathbf{z}_i, \mathbf{x}_a)_{\mathcal{D}})$$
$$= E_{\mathbf{z}_i \sim F}\left[\mathbb{1}(\mathbf{z}_i \in B(\mathbf{x}_a, \mathbf{d}(\mathbf{x}_a, \partial dom(F))))(\frac{1 - \mathcal{M}(B(\mathbf{x}_a, \mathbf{d}(\mathbf{x}_a, \mathbf{z}_i)) \cup B(\mathbf{x}_b, \mathbf{d}(\mathbf{x}_b, \mathbf{z}_i))}{1 - \mathcal{M}(B(\mathbf{x}_a, \mathbf{d}(\mathbf{x}_a, \mathbf{z}_i)))})^{\psi-1}\right]$$
$$\le E_{\mathbf{z}_i \sim F}\left[(\frac{1 - \mathcal{M}(B(\mathbf{x}_a, \mathbf{d}(\mathbf{x}_a, \mathbf{z}_i)) \cup B(\mathbf{x}_b, \mathbf{d}(\mathbf{x}_b, \mathbf{z}_i))}{1 - \mathcal{M}(B(\mathbf{x}_a, \mathbf{d}(\mathbf{x}_a, \mathbf{z}_i)))})^{\psi-1}\right]$$
$$= E_{\mathbf{z} \sim F}\left[(\frac{1 - \mathcal{M}(B(\mathbf{x}_a, \mathbf{d}(\mathbf{x}_a, \mathbf{z})) \cup B(\mathbf{x}_b, \mathbf{d}(\mathbf{x}_b, \mathbf{z}))}{1 - \mathcal{M}(B(\mathbf{x}_a, \mathbf{d}(\mathbf{x}_a, \mathbf{z})))})^{\psi-1}\right],$$

where $\mathcal{M}(S) = \int_{s\in S} F(s)ds$ *and* $\mathbf{z}$ *is a random variable whose pdf is* F*. So it holds that*

$$K_\psi(\mathbf{x}_a, \mathbf{x}_b) \le E_{\mathbf{z} \sim F}\left[(\frac{1 - \mathcal{M}(B(\mathbf{x}_a, \mathbf{d}(\mathbf{x}_a, \mathbf{z})) \cup B(\mathbf{x}_b, \mathbf{d}(\mathbf{x}_b, \mathbf{z}))}{1 - \mathcal{M}(B(\mathbf{x}_a, \mathbf{d}(\mathbf{x}_a, \mathbf{z})))})^{\psi-1}\right]$$

Since $\mathbf{x}_a \ne \mathbf{x}_b$*,* $\forall \mathbf{z}$*, it holds that* $0 < \frac{1 - \mathcal{M}(B(\mathbf{x}_a, \mathbf{d}(\mathbf{x}_a, \mathbf{z})) \cup B(\mathbf{x}_b, \mathbf{d}(\mathbf{x}_b, \mathbf{z}))}{1 - \mathcal{M}(B(\mathbf{x}_a, \mathbf{d}(\mathbf{x}_a, \mathbf{z})))} < 1$.

As a consequence, $\forall\epsilon > 0$*, there always exists a* ψ*, such that Isolation Kernel* $K_\psi(\mathbf{x}_a, \mathbf{x}_b \mid D) < \epsilon$. □

Lemma 2. $\mathcal{H}_K$ *has the following property for two distinct points* $\mathbf{x}_a$ *and* $\mathbf{x}_b$ *i.i.d. drawn from* F *on* Ω*:* $\forall e > 0, \exists\psi > 1$*,* $t > e^p$*,* $0 < \delta \le 1$ *such that*

$$P(\ell_p(\Phi(\mathbf{x}_a), \Phi(\mathbf{x}_b)) \le e) \le 2exp(-\frac{t(1-\frac{e^p}{t})^2}{2}) < \delta,$$

where t *and* ψ *are parameters of Isolation Kernel* K*, used to determine its feature map* Φ*; and* p *is the parameter in the* ℓ_p *metric.*

Proof. Recall IK's feature map $\Phi : \mathbf{x} \to \{0,1\}^{t\times\psi}$ *(see Definition 2). There are only three types of feature value-pairs when comparing the IK feature maps of* $\mathbf{x}_a$ *and* $\mathbf{x}_b$*: 0-1 (or 1-0), 1-1 and 0-0, as shown in Fig. B.6. The number of 0-1 pairs equals to* $\ell_p(\Phi(\mathbf{x}_a), \Phi(\mathbf{x}_b))^p$*; and the number of 1-1 pairs equals to* $t\hat{K}_\psi(\mathbf{x}_a, \mathbf{x}_b)$*. The total number of 0-1 pairs and 1-1 pairs is at least* t*, hence we have*

$$\ell_p(\Phi(\mathbf{x}_a), \Phi(\mathbf{x}_b))^p + t\hat{K}_\psi(\mathbf{x}_a, \mathbf{x}_b) \ge t. \tag{B.1}$$

Hoeffding's inequality: Let $s_1, s_2, \dots, s_n$ *be independent random variables such that, for all i,* $a_i \le s_i \le b_i, c_i = b_i - a_i$ *and* $\forall i, c_i \le C$*. Let* S_n *be their sum,* E_n *be* S_n*'s expected value, then it holds that*

$$P(|S_n - E_n| > \epsilon) \le 2exp(-\frac{2\epsilon^2}{\sum_{i=1}^n c_i^2}) \le 2exp(-\frac{2\epsilon^2}{nC^2}).$$

In the case of IK, the random variable s *is whether two points fall into the same cell (1) or not (0). Then* $K_\psi(\mathbf{x}_a, \mathbf{x}_b) = \frac{E_t}{t}$ *and* $\hat{K}_\psi(\mathbf{x}_a, \mathbf{x}_b) = \frac{S_t}{t}$*. And* $\forall i, c_i = C = 1$.

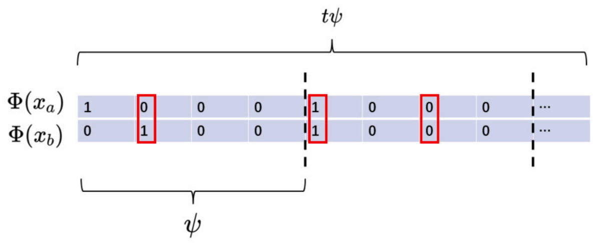


Fig. B.6. IK's feature map.

$\forall 0 < \delta \le 1, e > 0$, there exists $t > e^p$, such that $2exp(-\frac{t(1-\frac{e^p}{t})^2}{2}) < \delta$, where p is the p parameter in the ℓ_p metric.
$\forall e > 0$, from Lemma 2a, we can always find ψ such that

$$K_\psi(\mathbf{x}_a, \mathbf{x}_b) < \frac{1-\hat{e}}{2}, \text{ where } \hat{e} = \frac{e^p}{t} < 1.$$

Then from Hoeffding's inequality, we have the following:

$$\begin{aligned}
&P(\ell_p(\Phi(\mathbf{x}_a), \Phi(\mathbf{x}_b)) \le e)\\
&\le P(\hat{K}_\psi(\mathbf{x}_a, \mathbf{x}_b) \ge 1 - \hat{e})\\
&\le P(\hat{K}_\psi(\mathbf{x}_a, \mathbf{x}_b) > K_\psi(\mathbf{x}_a, \mathbf{x}_b) + \frac{1-\hat{e}}{2})\\
&= P(\hat{K}_\psi(\mathbf{x}_a, \mathbf{x}_b) - K_\psi(\mathbf{x}_a, \mathbf{x}_b) > \frac{1-\hat{e}}{2})\\
&\le P(|\hat{K}_\psi(\mathbf{x}_a, \mathbf{x}_b) - K_\psi(\mathbf{x}_a, \mathbf{x}_b)| > \frac{1-\hat{e}}{2})\\
&\le 2exp(-\frac{2(\frac{1-\hat{e}}{2}t)^2}{t})\\
&= 2exp(-\frac{t(1-\hat{e})^2}{2})\\
&= 2exp(-\frac{t(1-\frac{e^p}{t})^2}{2}) < \delta.
\end{aligned}$$

The second line holds because of the inequality in Equation (B.1). The third line holds because $K_\psi(\mathbf{x}_a, \mathbf{x}_b) < \frac{1-\hat{e}}{2}$ (that is, $K_\psi(\mathbf{x}_a, \mathbf{x}_b) + \frac{1-\hat{e}}{2} < 1 - \hat{e}$). The sixth line holds by substituting n with t, S_n with $t\hat{K}_\psi(\mathbf{x}_a, \mathbf{x}_b)$, E_n with $tK_\psi(\mathbf{x}_a, \mathbf{x}_b)$, ϵ with $\frac{1-\hat{e}}{2}t$, C with 1 in Hoeffding's inequality. □

According to Definition 3, when casting m to ℓ_p-metric and $\gamma(v) = 2\exp(-\frac{t(1-\frac{e^p}{t})^2}{2})$, where $v = t$ with some ψ setting of Isolation Kernel K, we have the following theorem:

Theorem 2. *Under the same condition as Lemma 2, the feature space $\mathcal{H}_K$ has distinguishability using ℓ_p-metric on $\mathcal{H}_K$.*

Proof. By casting ℓ_p and the second term in the above inequality to m and $\gamma(v)$ in Definition 3, respectively, and by noting that e takes any positive value, we know that the condition v represented by some ψ and t satisfying Eq. (3) always exists for any e and any number of dimensions d. Accordingly, $\mathcal{H}_K$ has distinguishability for ℓ_p-metric. □

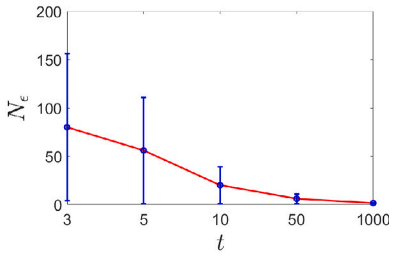
Appendix C. The influence of the number of partitionings on distinguishability

This section investigates two influences of partitionings on N_ϵ used in Section 4.1: (i) the number of partitionings t; and (ii) the partitions are generated using a dataset different from the given dataset.

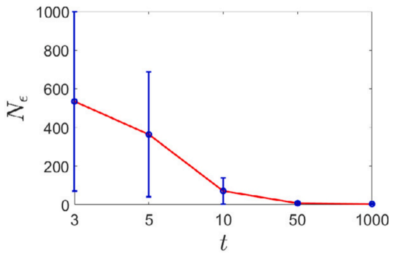
Fig. C.7(a) shows that N_ϵ decreases as t increases. This effect is exactly as predicted in Theorem 2, i.e., increasing t leads to increased distinguishability.

It is possible to derive an IK using a dataset of uniform distribution (which is different from a given real dataset). Fig. C.7(b) shows the outcome, i.e., it exhibits the same phenomenon as the IK derived from the given dataset, except that small t leads to poorer distinguishability (having higher N_ϵ and higher variance).

At high t, there is virtually no difference between the two versions of IK. This is the direct outcome of Lemma 2: the probability of any point in the data space falling into one of the ψ partitions is independent of the dataset used to generate the partitions.



(a) IK (given dataset)



(b) IK (uniform data distribution)

Fig. C.7. N_ϵ ($\epsilon = 0.005$) as a result of varying t on $d = 10000$. The same dataset in Fig. 1(a) is used. The result for each t value is the average and standard error over 10 trials.

Table D.9
Parameter search ranges for KDEs.

Algorithm	Parameter search ranges
IKDE	$\psi \in \{2^q \mid q = 1, \ldots, 12\}$; $t = 100$
OKDE	$\sigma \in \{2^m \mid m = -5, \ldots, 5\}$
KDEOS	$\sigma \in \{2^m \mid m = -5, \ldots, 5\}$;
	$k \in \{1, 3, 5, 7, 11, 21, 51, 101, 201, 501, 1001, 2001\}$
RACE	$\epsilon = 0.05$; $R = \{2^g \mid g = 3, \ldots, 10\}$; plus HBE settings
HBE	$L = 100$; $\sigma = \{2^m \mid m = -5, \ldots, 5\}$

Table D.10
Kernel parameters and their search ranges in SVM.

	Parameters and their search range
IK	$\psi \in \{2^m \mid m = 2, 3, \ldots, 12\}$
GK	$\sigma = d \times \mu$; $\mu \in \{2^{-5}, 2^{-4}, ..., 2^5\}$ (dense datasets)
	$\sigma \in \{2^{-5}, 2^{-4}, ..., 2^5\}$ (sparse datasets: $nnz\% < 1$)

Appendix D. Experimental settings

We implement Isolation Kernel with the Matlab R2021a. The parameter $t = 200$ is used for all the experiments using Isolation Kernel (IK).

For the experiments on instability of a measure (reported in Section 4.1), the similarity scores are normalized to [0,1] for all measures.

In the ball-tree search experiments, IK's feature map is used to map each point in the given dataset into a point in Hilbert space; and for linear kernel (LK), each point is normalized with the square root of its self similarity using LK. Then exactly the same ball-tree index using the Euclidean distance is employed to perform the indexing for LK, IK and distance. The parameter search ranges for IK are $\psi \in \{3, 5, .., 250\}$.

We used the package "sklearn.neighbors" from Python to conduct the indexing task. The leaf size is 15 and we query the 5 nearest neighbors of each point from the given dataset using the brute-force and ball-tree indexing. The dataset sizes have been limited to 10,000 or less because of the memory requirements of the indexing algorithm. We have also omitted the comparison with AG and SNN [27] because they require k-nearest neighbor (kNN) search. It does not make sense to perform indexing using AG/SNN to speed up an NN search because AG/SNN requires a kNN search (which itself requires an index to speed up the search).

The parameter search ranges used for the KDE experiment are shown in Table D.9.

To create a dataset for anomaly detection from a source dataset, a small number of points in the smallest class are randomly selected as anomalies; and the other classes are treated as normal points.

For the t-SNE [14] experiment, we set $tolerance = 0.00005$. We report the best visualized results from a search in $[5, 20, 40, 60, 80, 100, 250, 500, 800]$ for both ψ (using IK) and $perplexity$ (using GK). When using IK in t-SNE visualization, we replace the similarity matrix calculated by GK with the similarity matrix calculated by IK. For both FIt-SNE [39] and q-SNE [40], we report the best visualization result from the same $perplexity$ search range and fix all other parameters to the default values.

In Section 4.2, SVM classifiers in scikit-learn [60] are used with IK/LK and Gaussian kernel. They are based on LIBLINEAR [61] and LIBSVM [62].

Table D.10 shows the search ranges of the kernel parameters for the SVM classifier. A 5-fold cross validation on the training set is used to determine the best parameter. The reported accuracy in Table 7 is the accuracy obtained from the test set after the final model is trained using the training set and the 5-fold CV determined best parameter.

Table D.11 presents the separate runtimes of IK feature mapping and SVM used to complete the experiments in SVM classification.

Table D.11
Runtimes of IK feature mapping and SVM in CPU seconds. Note that the mapping time can be significantly reduced by using GPU, as shown in Table 7 in [22].

Dataset	ψ	Mapping	SVM
Url	32	2	1
News20	64	41	19
Rcv1	64	394	26
Real-sim	128	44	13
Gaussians	4	8	0.5
w-Gaussians	4	7	0.5
Cifar-10	128	594	493
Mnist	64	31	17
A9a	64	9	22
Ijcnn1	128	26	40

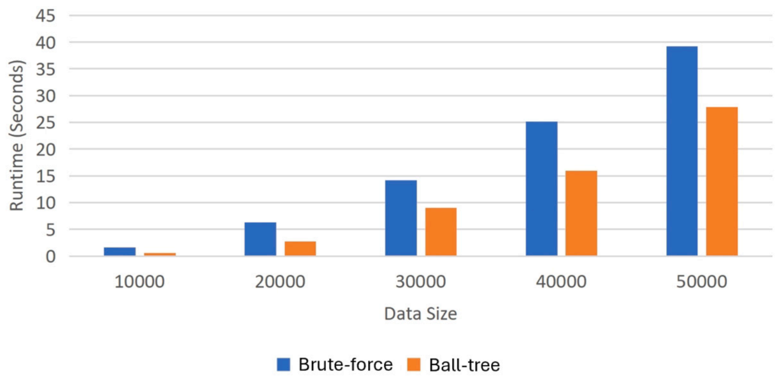


Fig. E.8. Runtime comparison between brute-force search and ball-tree search on the Ijcnn1 dataset.

All datasets used are obtained from https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/, except Gaussians and w-Gaussians which are our creation.

The machine used in the experiments has one Intel E5-2682 v4 @ 2.50GHz 16 cores CPU with 32 GB memory.

Appendix E. Runtime comparison between brute-force search and ball-tree search

Fig. E.8 compares the runtimes of the brute-force and the ball-tree search, using IK, on the Ijcnn1 dataset (with 22 attributes) with up to 50,000 points. It can be seen from the figure that a ball-tree can improve the runtime of the brute-force by 30%.
This shows that the small improvement shown in Table 3 is mainly due to the dataset size used. As the efficacy of an indexing scheme for a large database is outside the scope of this paper, we do not attempt to use large datasets in our experiment. The data size or the actual runtime improvement of an indexing scheme has no impact of the distinguishability of IK in high dimensions.

Appendix F. Kernel density estimation

Given a dataset $D \subset \mathbb{R}^d$ and a kernel κ_σ with bandwidth parameter σ, an ordinary kernel density estimator for any point $x \in \mathbb{R}^d$ is defined as follows:

$$f_\sigma(x|D) = \frac{1}{|D|} \sum_{y \in D} \kappa_\sigma(x, y) \tag{F.1}$$

The Gaussian kernel $\kappa_\sigma(x, y) = \frac{1}{\sqrt[d]{2\pi\sigma^d}} \exp(\frac{-\|x-y\|^2}{2\sigma^2})$ is often used.

IKDE [36] replaces κ_σ with IK.

KDEOS [26] employs an adaptive Gaussian kernel κ_{σ_x}:

$$f_{\sigma_x}(x|D) = \frac{1}{k} \sum_{y \in kNN(x)} \kappa_{\sigma_x}(x, y)$$

where $\sigma_x = \min(mean_{y \in kNN(x)} dist(y, x), \epsilon)$ which returns the minimum of the average distance of the k nearest neighbors of x found in D or a small positive constant ϵ to avoid zero bandwidth.

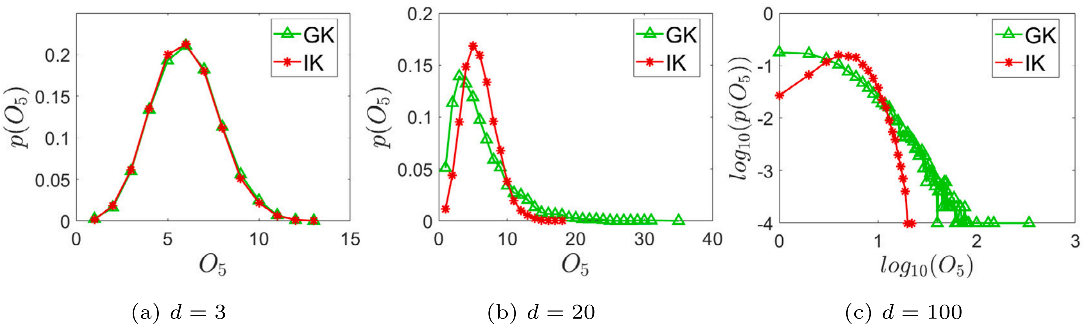


Fig. G.9. The effect of hubness in k-nearest neighbors: GK vs IK. The experiment setting is as used by [4]: a random dataset is drawn uniformly from the unit hypercube $[0,1]^d$. The settings used are: $\psi = 32$ for IK; $\sigma = 5$ for GK.

Here, the bandwidth parameter σ is replaced with k; yet a fixed setting of k allows the bandwidth to be adaptive to the local density of x.

RACE [34] and HBE [35] uses locality-sensitive hashing to provide sketches of the kernel summary (i.e., $\sum_{y \in D} \kappa(x,y)$) in order to avoid the summation over all points in the dataset. RACE [34] performs a kernel summary by using hash functions h_i as follows:

$$\sum_{i=1}^{t} A_i[h_i(x)|D] = \sum_{y \in D} \kappa(x,y)$$

where A_i is a sketch of D, represented as an equi-width-bin histogram, associated with each h_i; and $h_i(x)$ hashes to a bin in the histogram, i.e., $A_i[h_i(x)] \subset D$. The hash functions are from a family $\mathcal{F}$ such that $P_{h \sim \mathcal{F}}[h(x) = h(y)] = \kappa(x,y)$ [63]. The applicable kernels are p-stable LSH kernels [37] for Euclidean and Manhattan distances.

HBE [35] builds an LSH-based KDE via sampling. It can employ commonly used kernels such as Laplacian and Exponential kernels. To estimate x, a uniformly random point y is selected from bin $B_i(x) = \{z \in D | h_i(z) = h_i(x)\}$. The final estimation is given as:

$$f_B(x|D) = \frac{1}{t} \sum_{i=1;\ y \sim B_i(x)}^{t} \frac{|B_i(x)|}{n} \sqrt{\kappa(x,y)}$$

The common ingredient of these LSH-based KDEs is that they employ a random projection [37,34], to implement hash functions to map points in the data space to one-dimensional bins.

Appendix G. Hubness effect

Radovanović et al. [4] attribute the hubness effect to a consequence of high (intrinsic) dimensionality of data, and not factors such as sparsity and skewness of the distribution. However, it is unclear why hubness only occurs in k-nearest neighborhood, and not in ϵ-neighborhood on the same dataset.

In the context of k nearest neighborhood, it has been shown that there are few points which are the nearest neighbors to many points in a dataset of high dimensions [4]. Let $N_k(y)$ be the set of k nearest neighbors of y; and k-occurrences of x, $O_k(x) = |\{y : x \in N_k(y)\}|$, be the number of other points in the given dataset where x is one of their k nearest neighbors. As the number of dimensions increases, the distribution of $O_k(x)$ becomes considerably skewed (i.e., there are many points with zero or small O_k and only a few points have large $O_k(\cdot)$ for many widely used distance measures [4]. The points with large $O_k(\cdot)$ are considered as 'hubs', i.e., the popular nearest neighbors.

Fig. G.9 shows the result of O_5 vs $p(O_5)$ comparing Gaussian kernel and IK, where $p(O_k(x)) = \frac{|\{y \in D | O_k(y) = O_k(x)\}|}{|D|}$. Consistent with the result shown by Radovanović [4] for distance measure, this result shows that Gaussian kernel suffers from the hubness effect as the number of dimensions increases. In contrast, IK is not severely affected by the hubness effect.

Appendix H. IK with three implementations in retrieval and anomaly detection tasks

Here we compare IK with three current known implementations, i.e., Voronoi Diagram [8], Isolation Forest [7] and Isolating hyperspheres [20], in retrieval and anomaly detection tasks.

Table H.12 shows the retrieval precision of 5 nearest neighbors of exact NN searches using IK with three implementations. It shows that IK with Voronoi Diagram outperforms the two other implementations in general. This is because Isolating Hyperspheres cover the parts of the space which has large data size. Those uncovered points (having zero feature values) are mapped to the origin in the Hilbert space; therefore they are indistinguishable. Isolation Forest also loses information by using only a subset of the original attributes for space partitioning.

Table H.13 shows the anomaly detection results of kernel density estimator using IK with three implementations. It shows that IK with Isolating Hyperspheres has better detection accuracy than that with Voronoi Diagram on all datasets, except on the Gaussians

Table H.12
Retrieval result in terms of Precision@5 using IK: Comparing the three implementations of IK.

Dataset	Voronoi Diagram	Isolating Hyperspheres	Isolation Forest
Url	0.95	0.92	**0.97**
News20	**0.88**	0.78	0.69
Rcv1	**0.94**	0.92	0.92
Realsim	**0.90**	0.82	0.77
Gaussians	**1.00**	**1.00**	**1.00**
w-Gaussians	**1.00**	0.98	**1.00**
Cifar10	0.27	0.24	**0.30**
Mnist	**0.93**	0.88	**0.93**
A9a	**0.79**	**0.79**	**0.79**
Ijcnn1	**0.96**	0.95	**0.96**
average	0.86	0.83	0.83

Table H.13
Anomaly detection results (in terms of AUC) of kernel density estimator using IK: comparing the three implementations of IK. The Mnist_230 dataset has the normal hand-writings of digits 2 & 3 which can be easily differentiated from those anomalous digit 0. The Mnist_479 dataset has the normal hand-writings of digits 4 & 7 which are hard to differentiate from the anomalous digit 9.

Dataset	Voronoi Diagram	Isolating Hyperspheres	Isolation Forest
Url	**0.73**	0.72	0.50
News20	0.67	**0.70**	0.52
Rcv1	0.78	**0.84**	0.63
Realsim	0.86	**0.92**	0.58
Gaussians	**1.00**	**1.00**	**1.00**
w-Gaussians	0.93	**0.97**	0.91
Cifar10	0.63	**0.70**	0.60
Mnist_479	0.69	**0.82**	0.63
Mnist_230	0.97	**0.99**	0.83
A9a	0.59	**0.63**	0.59
Ijcnn1	0.72	0.74	**0.75**
average	0.78	**0.82**	0.68

dataset where they have the same accuracy. IK with Isolation Forest performs worse than the other implementations in all datasets, except on Ijcnn1. In particular, it has very poor accuracy on the four datasets with the highest number of dimensions. It has been known previously that Isolation Forest, as an anomaly detector, has poor detection accuracy in high dimensional datasets [64]. This is because it utilizes a subset of the dimensions only to create isolation trees, and they are not sufficient to detect anomalies when the number of relevant dimensions is low. The issues apply here.

References

[1] D.L. Donoho, High-Dimensional Data Analysis: The Curses and Blessings of Dimensionality, Invited Lecture at Mathematical Challenges of the 21st Century, AMS National Meeting, 2000.
[2] M. Talagrand, A new look at independence, Ann. Probab. 24 (1) (1996) 1–34.
[3] K.S. Beyer, J. Goldstein, R. Ramakrishnan, U. Shaft, When is “nearest neighbor” meaningful?, in: Proceedings of the 7th International Conference on Database Theory, Springer-Verlag, London, UK, 1999, pp. 217–235.
[4] M. Radovanović, A. Nanopoulos, M. Ivanović, Hubs in space: popular nearest neighbors in high-dimensional data, J. Mach. Learn. Res. 11 (86) (2010) 2487–2531.
[5] K.P. Bennett, U. Fayyad, D. Geiger, Density-based indexing for approximate nearest-neighbor queries, in: Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1999, pp. 233–243.
[6] V. Pestov, On the geometry of similarity search: dimensionality curse and concentration of measure, Inf. Process. Lett. 73 (1) (2000) 47–51.
[7] K.M. Ting, Y. Zhu, Z.-H. Zhou, Isolation kernel and its effect on SVM, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2018, pp. 2329–2337.
[8] X. Qin, K.M. Ting, Y. Zhu, V.C.S. Lee, Nearest-neighbour-induced isolation similarity and its impact on density-based clustering, in: Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence, 2019, pp. 4755–4762.
[9] V.D. Mil’man, New proof of the theorem of A. Dvoretzky on intersections of convex bodies, Funct. Anal. Appl. 5 (4) (1972) 288–295.
[10] A. Hinneburg, C.C. Aggarwal, D.A. Keim, What is the nearest neighbor in high dimensional spaces?, in: Proceedings of the 26th International Conference on Very Large Data Bases, 2000, pp. 506–515.
[11] C.C. Aggarwal, A. Hinneburg, D.A. Keim, On the surprising behavior of distance metrics in high dimensional space, in: Proceedings of the 8th International Conference on Database Theory, 2001, pp. 420–434.
[12] A. Kabán, On the distance concentration awareness of certain data reduction techniques, Pattern Recognit. 44 (2) (2011) 265–277.
[13] I.T. Jolliffe, Principal Component Analysis, Springer Series in Statistics, Springer-Verlag, New York, 2002.
[14] L. van der Maaten, G.E. Hinton, Visualizing high-dimensional data using t-SNE, J. Mach. Learn. Res. 9 (2) (2008) 2579–2605.
[15] H.-P. Kriegel, M. Schubert, A. Zimek, Angle-based outlier detection in high-dimensional data, in: Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Association for Computing Machinery, 2008, pp. 444–452.

[16] D. Francois, V. Wertz, M. Verleysen, The concentration of fractional distances, IEEE Trans. Knowl. Data Eng. 19 (7) (2007) 873–886.
[17] S. Aryal, K.M. Ting, T. Washio, G. Haffari, Data-dependent dissimilarity measure: an effective alternative to geometric distance measures, Knowl. Inf. Syst. 53 (2) (2017) 479–506.
[18] S. Kpotufe, K-NN regression adapts to local intrinsic dimension, in: Proceedings of the 24th International Conference on Neural Information Processing Systems, 2011, pp. 729–737.
[19] F.T. Liu, K.M. Ting, Z.-H. Zhou, Isolation forest, in: Proceedings of the IEEE International Conference on Data Mining, 2008, pp. 413–422.
[20] K.M. Ting, B.-C. Xu, T. Washio, Z.-H. Zhou, Isolation distributional kernel: a new tool for kernel based anomaly detection, in: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2020, pp. 198–206.
[21] B.-C. Xu, K.M. Ting, Z.-H. Zhou, Isolation set-kernel and its application to multi-instance learning, in: Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2019, pp. 941–949.
[22] K.M. Ting, J.R. Wells, T. Washio, Isolation kernel: the X factor in efficient and effective large scale online kernel learning, Data Min. Knowl. Discov. 35 (6) (2021) 2282–2312.
[23] M. Gromov, V. Milman, A topological application of the isoperimetric inequality, Am. J. Math. 105 (4) (1983) 843–854.
[24] L. Devroye, L. Györfi, G. Lugosi, H. Walk, On the measure of Voronoi cells, J. Appl. Probab. 54 (2) (2017) 394–408.
[25] S.M. Omohundro, Five balltree construction algorithms, International Computer Science Institute Berkeley, 1989.
[26] E. Schubert, A. Zimek, H.-P. Kriegel, Generalized outlier detection with flexible kernel density estimates, in: Proceedings of the SIAM Conference on Data Mining, 2014, pp. 542–550.
[27] R.A. Jarvis, E.A. Patrick, Clustering using a similarity measure based on shared nearest neighbors, IEEE Trans. Comput. 100 (11) (1973) 1025–1034.
[28] M.E. Houle, H.-P. Kriegel, P. Kröger, E. Schubert, A. Zimek, Can shared-neighbor distances defeat the curse of dimensionality?, in: Proceedings of the International Conference on Scientific and Statistical Database Management, 2010, pp. 482–500.
[29] L. Zelnik-Manor, P. Perona, Self-tuning spectral clustering, in: Advances in Neural Information Processing Systems, 2005, pp. 1601–1608.
[30] U. Shaft, R. Ramakrishnan, Theory of nearest neighbors indexability, ACM Trans. Database Syst. 31 (3) (2006) 814–838.
[31] A.M. Kibriya, E. Frank, An empirical comparison of exact nearest neighbour algorithms, in: Proceedings of European Conference on Principles of Data Mining and Knowledge Discovery, 2007, pp. 140–151.
[32] D.W. Scott, Multivariate Density Estimation: Theory, Practice, and Visualization, 2nd edition, Wiley, 2015.
[33] K. Muandet, K. Fukumizu, B. Sriperumbudur, B. Schölkopf, Kernel mean embedding of distributions: a review and beyond, Found. Trends Mach. Learn. 10 (1–2) (2017) 1–141.
[34] B. Coleman, A. Shrivastava, Sub-linear race sketches for approximate kernel density estimation on streaming data, in: Proceedings of the World Wide Web Conference, 2020, pp. 1739–1749.
[35] A. Backurs, P. Indyk, T. Wagner, Space and time efficient kernel density estimation in high dimensions, in: Advances in Neural Information Processing Systems, vol. 32, 2019, pp. 15799–15808.
[36] K.M. Ting, T. Washio, J.R. Wells, H. Zhang, Isolation kernel density estimation, in: Proceedings of IEEE International Conference on Data Mining, 2021, pp. 619–628.
[37] M. Datar, N. Immorlica, P. Indyk, V.S. Mirrokni, Locality-sensitive hashing scheme based on p-stable distributions, in: Proceedings of the 20th Annual Symposium on Computational Geometry, 2004, pp. 253–262.
[38] F. Wickelmaier, An Introduction to MDS, Sound Quality Research Unit, Aalborg University, 2003.
[39] G.C. Linderman, M. Rachh, J.G. Hoskins, S. Steinerberger, Y. Kluger, Fast interpolation-based t-SNE for improved visualization of single-cell RNA-seq data, Nat. Methods 16 (3) (2019) 243–245.
[40] A. Häkkinen, J. Koiranen, J. Casado, K. Kaipio, O. Lehtonen, E. Petrucci, J. Hynninen, S. Hietanen, O. Carpén, L. Pasquini, et al., qSNE: quadratic rate t-SNE optimizer with automatic parameter tuning for large datasets, Bioinformatics 36 (20) (2020) 5086–5092.
[41] R.J. Durrant, A. Kabán, When is 'nearest neighbour' meaningful: a converse theorem and implications, J. Complex. 25 (4) (2009) 385–397.
[42] A. Hendrikse, R. Veldhuis, L. Spreeuwers, Likelihood-ratio-based verification in high-dimensional spaces, IEEE Trans. Pattern Anal. Mach. Intell. 36 (1) (2013) 127–139.
[43] F. Bach, Breaking the curse of dimensionality with convex neural networks, J. Mach. Learn. Res. 18 (2017) 1–53.
[44] A. O'Brien, P. Szul, O. Luo, A. George, R. Dunne, D. Bauer, Breaking the curse of dimensionality for machine learning on genomic data, in: Proceedings of the Workshop on Advances in Bioinformatics and Artificial Intelligence, 2017.
[45] S. Sarkar, A.K. Ghosh, On perfect clustering of high dimension, low sample size data, IEEE Trans. Pattern Anal. Mach. Intell. 42 (9) (2019) 2257–2272.
[46] V. Cabannes, L. Pillaud-Vivien, F. Bach, A. Rudi, Overcoming the curse of dimensionality with Laplacian regularization in semi-supervised learning, in: Advances in Neural Information Processing Systems, 2021.
[47] H. Njah, S. Jamoussi, W. Mahdi, Breaking the curse of dimensionality: hierarchical Bayesian network model for multi-view clustering, Ann. Math. Artif. Intell. 16 (2021) 1013–1033.
[48] V. Pestov, An axiomatic approach to intrinsic dimension of a dataset, Neural Netw. 21 (2) (2008) 204–213.
[49] F. Angiulli, On the behavior of intrinsically high-dimensional spaces: distances, direct and reverse nearest neighbors, and hubness, J. Mach. Learn. Res. 18 (170) (2018) 1–60.
[50] L. Amsaleg, O. Chelly, M.E. Houle, K. Kawarabayashi, M. Radovanovic, W. Treeratanajaru, Intrinsic dimensionality estimation within tight localities, in: Proceedings of the SIAM International Conference on Data Mining, 2019, pp. 181–189.
[51] B. Kulis, K. Grauman, Kernelized locality-sensitive hashing, IEEE Trans. Pattern Anal. Mach. Intell. 34 (6) (2012) 1092–1104.
[52] M. Aumüller, E. Bernhardsson, A. Faithfull, ANN-Benchmarks: a benchmarking tool for approximate nearest neighbor algorithms, Inf. Sci. 87 (2020) 101374.
[53] B. Schölkopf, J.C. Platt, J.C. Shawe-Taylor, A.J. Smola, R.C. Williamson, Estimating the support of a high-dimensional distribution, Neural Comput. 13 (7) (2001) 1443–1471.
[54] K. Muandet, B. Schölkopf, One-class support measure machines for group anomaly detection, in: Proceedings of the Twenty-Ninth Conference on Uncertainty in Artificial Intelligence, 2013, pp. 449–458.
[55] K.M. Ting, B.-C. Xu, T. Washio, Z.-H. Zhou, Isolation distributional kernel: a new tool for point and group anomaly detections, IEEE Trans. Knowl. Data Eng. 35 (03) (2023) 2697–2710.
[56] C.K.I. Williams, M. Seeger, Using the Nyström method to speed up kernel machines, in: T.K. Leen, T.G. Dietterich, V. Tresp (Eds.), Advances in Neural Information Processing Systems, vol. 13, MIT Press, 2001, pp. 682–688.
[57] A. Rahimi, B. Recht, Random features for large-scale kernel machines, in: Advances in Neural Information Processing Systems, 2007, pp. 1177–1184.
[58] X.Y. Felix, A.T. Suresh, K.M. Choromanski, D.N. Holtmann-Rice, S. Kumar, Orthogonal random features, in: Advances in Neural Information Processing Systems, 2016, pp. 1975–1983.
[59] Y. Zhu, K.M. Ting, Improving the effectiveness and efficiency of stochastic neighbour embedding with isolation kernel, J. Artif. Intell. Res. 71 (2021) 667–695.
[60] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: machine learning in python, J. Mach. Learn. Res. 12 (2011) 2825–2830.
[61] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, C.-J. Lin, LIBLINEAR: a library for large linear classification, J. Mach. Learn. Res. (2008) 1871–1874.
[62] C.-C. Chang, C.-J. Lin, LIBSVM: a library for support vector machines, ACM Trans. Intell. Syst. Technol. 2 (27) (2011) 1–27.

[63] M.S. Charikar, Similarity estimation techniques from rounding algorithms, in: Proceedings of the 34th ACM Symposium on Theory of Computing, 2002, pp. 380–388.

[64] T.R. Bandaragoda, K.M. Ting, D. Albrecht, F.T. Liu, Y. Zhu, J.R. Wells, Isolation-based anomaly detection using nearest neighbour ensembles, Comput. Intell. 34 (4) (2018) 968–998.