Overcoming Key Weaknesses of Distance-based Neighbourhood Methods using a Data Dependent Dissimilarity Measure

Kai Ming Ting
Federation University
Victoria 3842, Australia
kaiming.ting@federation.edu.au

Ye Zhu, Mark Carman
Monash University
Victoria 3800, Australia
{yale.zhu, mark.carman}@monash.edu

Yue Zhu, Zhi-Hua Zhou
Nanjing University
Nanjing 210023, China
{zhuy,zhouzh}@lamda.nju.edu.cn

ABSTRACT

This paper introduces the first generic version of data dependent dissimilarity and shows that it provides a better closest match than distance measures for three existing algorithms in clustering, anomaly detection and multi-label classification. For each algorithm, we show that by simply replacing the distance measure with the data dependent dissimilarity measure, it overcomes a key weakness of the otherwise unchanged algorithm.

Keywords

Data dependent dissimilarity; distance measure; distance-based neighbourhood; probability-mass-based neighbourhood; k nearest neighbours.

1. INTRODUCTION AND MOTIVATION

Many data mining algorithms rely on a distance measure to provide the closest match between a test instance and instances in a database in order to find its nearest neighbours. The distance calculation is the core process that has been applied to all aspects of data mining tasks, including density estimation, clustering, anomaly detection and classification.

Despite its widespread applications, research in psychology has pointed out since 1970's that distance measures do not possess the key property of dissimilarity as judged by humans [12, 20], i.e., the characteristic where two instances in a dense region are less similar to each other than two instances of the same interpoint distance in a sparse region. Researchers have suggested that this *data dependent dissimilarity* is a better measure than the data independent geometric model based distance measure in psychological tests [12]. For example, two Caucasians will be judged as less similar when compared in Europe (where there are many Caucasians) than in Asia (where there are few Caucasians and many Asians.)

KDD '16, August 13 - 17, 2016, San Francisco, CA, USA

ISBN 978-1-4503-4232-2/16/08...$15.00

DOI: http://dx.doi.org/10.1145/2939672.2939779

We introduce a data dependent dissimilarity which has the above-mentioned characteristic, and we provide concrete evidence that it is a better measure than standard distance measures for three existing algorithms which rely on distance. Using probability mass rather than distance as the means to find the closest match neighbourhood heralds a fundamental change of perspective.

The neighbourhood of an instance has been used for different functions in various data mining tasks. Table 1 shows the key functions and key weaknesses of three existing algorithms relying on distance measure. It is instructive to see that a mere replacement of distance with the data dependent dissimilarity in these algorithms changes the perspective. The corresponding 'new' functions are described in the last column of Table 1. Although the rest of the procedures in each algorithm are unchanged, it overcomes the key weaknesses of these algorithms.

This paper makes the following contributions:

1. A generic data dependent dissimilarity, named mass-based dissimilarity, is proposed to allow for different implementations.

2. Deriving a new neighbourhood function from the data dependent dissimilarity and showing that it can replace an existing neighbourhood density function to overcome a key weakness in density-based clustering, i.e., its inability to find all clusters of varying densities.

3. Through our empirical evaluation, we demonstrate that the data dependent dissimilarity provides a better closest match than distance measure for k-nearest neighbour algorithms in anomaly detection and multi-label classification tasks.

The remainder of the paper is organised as follows. Section 2 introduces the related work. Sections 3 and 4 present the proposed dissimilarity and the new neighbourhood function. Section 5 describes an existing similarity which is a special case of the proposed dissimilarity. Section 6 introduces how the dissimilarity can be applied to three existing algorithms. Section 7 presents the empirical evaluation results. A discussion of related issues and conclusions are provided in the last two sections.

Software download: The source code of mass-based dissimilarity can be obtained at https://sourceforge.net/projects/mass-based-dissimilarity/.

Table 1: Key functions and key weaknesses of algorithms that rely on distance in three tasks and their replacement functions due to mass-based dissimilarity. Details of kNN and MLkNN are provided in Section 6.

Algorithm	Key function	Key weakness	Replacement function
Density-based clustering	Identify core points which have high density using distance-based neighbourhood estimation	Inability to find all clusters of varying densities	Identify core points which have high probability mass using probability mass-based neighbourhood estimation
kNN anomaly detector	Identify anomalies as points with the longest distance to the kth nearest neighbours	Inability to detect local anomalies	Identify anomalies as points having the highest probability mass of the kth lowest probability mass neighbours
Multi-label kNN classifier (MLkNN)	Estimate class-conditioned likelihoods using a frequency estimate based on k nearest neighbours	Poor likelihood estimation in cases where the local neighbourhood covers regions of varied density	Estimate class-conditioned likelihoods using a frequency estimate based on k lowest probability mass neighbours

2. RELATED WORK

Psychologists since 1970's have expressed their concerns on the use of geometric model for dissimilarity measure [12, 20]. The psychological tests they have conducted have clearly shown that the dissimilarity between two instances, as judged by humans, is influenced by the context of measurements and other instances in proximity. It is suggested that a dissimilarity measure which is akin to human's judged dissimilarity is one that interprets two instances in a dense region to be less similar than two instances of equal interpoint distance but located in a less dense region [12].

The first version of data dependent similarity [2], which has the characteristic prescribed by psychologists above, has been shown to provide a more effective match than distance measures in nearest neighbour search for k nearest neighbour classifiers and information retrieval. It is named m_p-dissimilarity and defined in the same form as the ℓ_p-norm, except that the dissimilarity in dimension i is probability mass in a region $P(R_i(x,y))$ rather than distance $|x_i - y_i|$. Region $R_i(x,y)$ is defined as an interval from $min(x_i,y_i)-\delta$ to $max(x_i,y_i)+\delta$, where δ is some small constant value.This implementation requires a search to find all instances in the region, and is not efficient.

This paper presents a general definition of data dependent dissimilarity in which m_p-dissimilarity [2] is a special case. The proposed implementation is efficient and robust for more applications.

A recent definition of mass estimation [7] is the basis of the proposed dissimilarity. We effectively extend from the mass estimation of one point to a dissimilarity measure of two points.

Based on information theory, Lin [14] suggests a probabilistic measure of similarity in ordinal domain as follows:

$$sim(o_a, o_b) = \frac{2 \times \log \sum_{j=\min(a,b)}^{\max(a,b)} P(o_j)}{\log P(o_a) + \log P(o_b)}$$

where o_j is an ordinal random variable.

Like most distance measures, Lin's measure [14] yields a constant maximum value for self similarity; and it has the opposite characteristic of the judged dissimilarity, i.e., two instances are more similar in a dense region than two instances of equal interpoint distance in a sparse region.

Another existing data dependent dissimilarity is shared nearest neighbour (SNN) which has been investigated in clustering only thus far [9, 11]. A detailed discussion of the relationship between SNN and the proposed mass-based dissimilarity is devoted in Section 5.

3. MASS-BASED DISSIMILARITY

Geometric model based measures solely depend on geometric positions to derive their distance measures. Instead, a data dependent dissimilarity mainly depends on data distribution, i.e., the probability mass of the (smallest) region covering the two instances. We use the terms: probability mass or mass or probability, interchangeably hereafter; and name the proposed measure: **mass-based dissimilarity**.

Let D be a data sample from pdf (probability density function) F; and $H \in \mathcal{H}(D)$ be a hierarchical partitioning model of the space into non-overlapping and non-empty regions. The definitions for the domain of $\mathbb{R}^d$, where d is the number of dimensions, are given as follows.

Definition 1. $R(x,y|H;D)$ is the smallest local region covering x and y wrt H and D is defined as:

$$R(x,y|H;D) = \underset{r \subset H \;\; s.t. \{x,y\} \in r}{\arg\min} \sum_{z \in D} \mathbb{1}(z \in r) \qquad (1)$$

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

Definition 2. Mass-based dissimilarity of x and y wrt D and F is defined as the expected probability of $R(x,y|H;D)$:

$$m(x,y|D,F) = E_{\mathcal{H}(D)}[P_F(R(x,y|H;D))] \qquad (2)$$

where $P_F(\cdot)$ is the probability wrt F; and the expectation is taken over all models in $\mathcal{H}(D)$.

In practice, the mass-based dissimilarity would be estimated from a finite number of models $H_i \in \mathcal{H}(D), i = 1,\ldots,t$ as follows:

$$m_e(x,y|D) = \frac{1}{t}\sum_{i=1}^{t} \tilde{P}(R(x,y|H_i;D)) \qquad (3)$$

where $\tilde{P}(R) = \frac{1}{|D|}\sum_{z \in D} \mathbb{1}(z \in R)$.

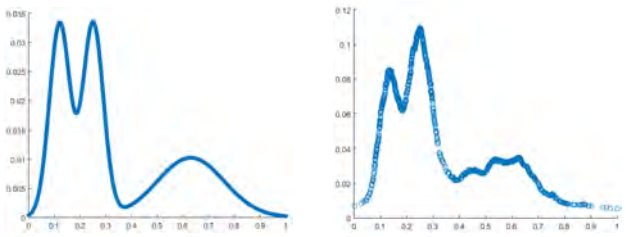
Note that $R(x,y|H;D)$ is the smallest local region covering x and y, it is analogous to the shortest distance between x and y used in the geometric model. Hereafter D is dropped in the notations when the context is clear.

3.1 Self-dissimilarity

One key difference of the mass-based dissimilarity is self-dissimilarity. Unlike self-dissimilarity of other measures (which usually take a constant value equal to the minimum dissimilarity or 0 in [0,1]), $m_e(x,x)$ is not a constant and ranges over [0,1], depending on the data distribution and the partitioning strategy used.

Table 2: m_e versus ℓ_p. $\partial(\cdot,\cdot)$ is a dissimilarity function. $m_e(x,y)=0$ when both x and y are in an empty region.

	$\partial(x,x)$	$\partial(x,y), \forall y \neq x$	Properties
m_e	[0,1]	[0,1]	$\forall x \neq y, m_e(x,x) \leq m_e(x,y)$
			$\exists x \neq y; x \neq z, m_e(x,x) > m_e(z,y)$
ℓ_p	0	(0,1]	$\forall x; y \neq z, \ell_p(x,x) < \ell_p(z,y)$



(a) True density distribution (b) Distribution of $m_e(x,x|D)$

Figure 1: (a) A true density distribution; (b) $m_e(x,x|D)$ based on Random Trees (level=8), where D has 1500 instances randomly sampled from (a)

The differences between m_e and ℓ_p are shown in Table 2. The reasons of the two properties of m_e are given below:

- $\forall x \neq y, m_e(x,x) \leq m_e(x,y)$: because $R(x,x|H) \subseteq R(x,y|H)$.
- $\exists x \neq y; x \neq z, m_e(x,x) > m_e(z,y)$: As the mass distribution of $m_e(x,x)$ is not a constant[1]; if x is in the region that includes the maximum mass point, then its mass value will be larger than the dissimilarity of some points which are close to the fringe or have the minimum mass values.

The distribution of the self-dissimilarity is equivalent to mass distribution [18], and its properties depend on the models $\mathcal{H}$ used:

- Half-space is used to define R. Here $m_e(x,y|D)$ reduces to half-space mass [7] if $x = y$. The mass distribution is always concave within the area bounded by the data, irrespective of the density distribution of the given data set [7].
- Random Trees: Here $m_e(x,y|D)$ reduces to level-h mass estimation [19] if $x = y$. An example is shown in Figure 1(b).

3.2 Model used to define a region

Though there are many methods to implement a model to define regions for mass estimation [19], we employ a method based on completely random trees[2] to implement mass-based dissimilarity in this paper.

We use a recursive partitioning scheme called *iForest* (isolation Forest) [15] to define regions. Though *iForest* was initially designed for anomaly detection [15], it has been shown that it is a special case of mass estimation [19].

[1]Note that mass distribution is not uniform even for a uniformly distributed pdf. See [7, 18] for details.

[2]Note that the random trees are not RandomForests [5] because the trees are completely random, built without class labels and any attribute selection criterion.

The implementation can be divided into two steps. There are two input parameters to this procedure. They are: t - the number of *iTrees* (isolation Trees); and ψ - the subsampling size used to build each *iTree*. The height limit h of each *iTree* is automatically set by $\psi : h = \lceil log_2 \psi \rceil$.

The first step is to build an *iForest* consisting of t *iTrees* as the partitioning structure R. Each *iTree* is built independently using a subset $\mathcal{D} \subset D$, where $|\mathcal{D}| = \psi$. A randomly selected split is employed at each internal node of an *iTree* to partition the sample set at the node into two non-empty subsets until every point is isolated or the maximum tree height h is reached. In the experiments described in Section 7, we perform axis-parallel split at each node of an *iTree* to build *iForest*. The details of the axis-parallel split *iTree* building process can be found in the Appendix.

After an *iForest* is built, all instances in D are traversed through each tree in order to record the mass of each node.

The second step is the evaluation step. Test points x and y are parsed through each *iTree* to calculate the sum of mass of the lowest nodes containing both x and y, i.e., $\sum_i |R(x,y|H_i)|$. Finally, $m_e(x,y)$ is the mean of these mass values over t *iTrees* as defined below:

$$m_e(x,y) = \frac{1}{t}\sum_{i=1}^{t} \frac{|R(x,y|H_i)|}{|D|} \quad (4)$$

4. μ-NEIGHBOURHOOD MASS

We introduce a new function: μ-neighbourhood mass, which denotes the number of points in a region defined by the mass-based dissimilarity, is defined as follows:

$$M_\mu(x) = \#\{y \in D \mid m_e(x,y) \leq \mu\}$$

Like ϵ-neighbourhood density[3], μ-neighbourhood mass produces an estimate based on a region defined by a dissimilarity measure. However, the region is defined by the expected probability mass (instead of distance.) Like ϵ, the parameter μ controls the size of the region: large (small) μ defines a large (small) region.

Figures 2(a) and 2(b) compare ϵ-neighbourhood with μ-neighbourhood on a dataset having three areas of different densities. Note that the volume of the region defined by μ-neighbourhood mass depends on data distribution—small in dense area and large in sparse area, shown as areas A, B and C in Figure 2(b). Note that the overall shape is not symmetrical. In contrast, ϵ-neighbourhood forms a region which is independent of data distribution with constant volume, in addition to regular and symmetrical shape.

Figure 3(a) shows that a μ-neighbourhood region becomes symmetrical only in the case of a uniform density distribution. Figure 3 shows that the shape depends on the implementation: axis-parallel and non axis-parallel random trees yield diamond and spherical shapes, respectively.

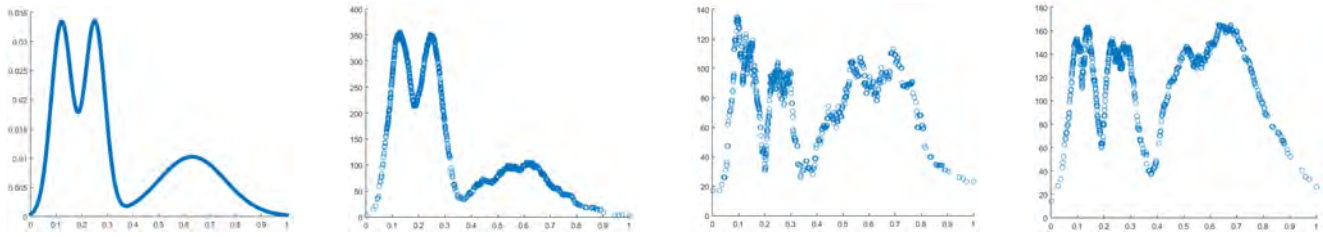
A conceptual comparison between the two neighbourhood functions is given in Table 3. In order to obtain non-zero M_μ, μ must be set higher than $\max_{z \in D} m_e(z,z)$. In other words, μ-neighbourhood mass employs mass distribution (i.e., the distribution of self-dissimilarity) as the reference.

For the purpose of clustering, mass can be used in a similar way as density to identify core points, i.e., instances having high mass values are core points; and those having low mass values are noise.

[3]As used in DBSCAN [10].

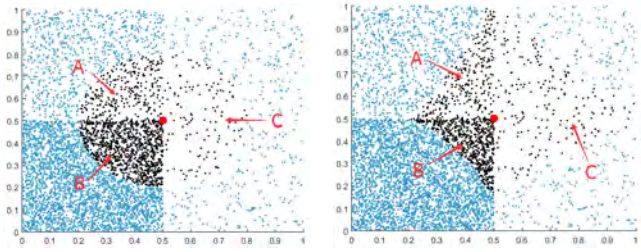
Table 3: μ-neighbourhood mass versus ϵ-neighbourhood density

	μ-neighbourhood mass	ϵ-neighbourhood density
Definition	$M_\mu(x) = \#\{y \in D \mid m_e(x,y) \leq \mu\}$ $M_\mu(x) = 0$ if $m_e(x,x) > \mu$ $M_\mu(x) \geq 0$ if $m_e(x,x) \leq \mu$	$N_\epsilon(x) = \#\{y \in D \mid \ell_p(x,y) \leq \epsilon\}$ $\forall x; 0 < \epsilon \leq 1, N_\epsilon(x) \geq 1$
Region	$m_e(x,x)$ is the reference; not symmetrical; volume unfixed	Distance from x; symmetrical; constant volume



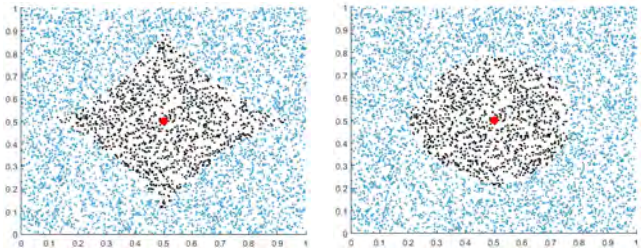
(a) True density distribution (b) ϵ-neighbourhood density (c) μ-neighbourhood mass (d) σ-neighbourhood mass

Figure 4: Density distribution of a "hard distribution" and its estimations using ϵ-neighbourhood density ($\epsilon = 0.03$), μ-neighbourhood mass based on $iForest$ (level=8 and $\mu = 0.2$) and σ-neighbourhood mass based on SNN ($k = 200$ and $\sigma = 0.3$) from a sample of 1500 instances of (a).



(a) ϵ-neighbourhood region (b) μ-neighbourhood region

Figure 2: (a) and (b) show the two regions defined by ϵ-neighbourhood density ($\epsilon = 0.25$) and μ-neighbourhood mass ($\mu = 0.55$), respectively, on a dataset having three areas of different densities, with reference to the red point (0.5,0.5). The blue-coloured dots denotes the points and the dark-coloured dots denotes the points within the region defined by the ϵ- or μ-neighbourhood estimator.



(a) μ-neighbourhood region-A (b) μ-neighbourhood region-B

Figure 3: (a) and (b) show the two regions defined by μ-neighbourhood mass ($\mu = 0.55$) using axis-parallel $iForest$ and non axis-parallel $iForest$, respectively, on a dataset with uniform density distribution with reference to the red point (0.5,0.5).

The next subsection describes a characteristic of μ-neighbourhood mass which makes it a better candidate than ϵ-neighbourhood density to identify core points, especially in a dataset which has clusters of varying densities.

4.1 Characteristic of μ-neighbourhood mass

One interesting characteristic of μ-neighbourhood mass is that valleys of a data distribution can be constrained within a small range of mass values by setting an appropriate μ, where a valley is the area surrounding a local minimum in the distribution. This characteristic is especially important in clustering algorithms which rely on a global threshold to identify core points before grouping the core points into separate clusters. Having all the valleys close to a small mass value, a global threshold slightly larger than this value will identify the majority of the core points of all clusters, irrespective of the densities of the clusters.

In contrast, ϵ-neighbourhood density does not possess this characteristic as it estimates density distribution and its valleys can have huge varying densities. As a result, a clustering algorithm, such as DBSCAN [10] which employs the ϵ-neighbourhood density estimator and relies on a global threshold to identify core points, is unable to detect all clusters of varying densities. This kind of distribution, shown in Figure 4(a), is called hard distribution because it is hard for DBSCAN [10] to identify all the clusters in the distribution.

Possessing the above-mentioned characteristic, the hard distribution (in terms of density) will exhibit as easy distribution (in terms of mass). Examples of ϵ-neighbourhood density and μ-neighbourhood mass are given in Figures 4(b) and 4(c), respectively.

In summary, μ-neighbourhood mass converts all valleys of different densities to become valleys of almost equal low mass by using an appropriate μ, if it exists.

The same applies to peaks, i.e., the difference in mass between peaks at dense and sparse regions can be reduced to a smaller value close to zero with an appropriate μ. However, in the clustering context, it is more important to have valleys reduced to about the same value so that a global threshold can be used to easily identify all clusters.

5. RELATION TO *SNN* SIMILARITY

A measure based on shared nearest neighbours (SNN) in k nearest neighbours has been proposed for clustering [11]:

> "Data points are similar to the extent that they share the same nearest neighbours; in particular, two data points are similar to the extent that their respective k nearest neighbour lists match. In addition, for this similarity measure to be valid, it is required that the tested points themselves belong to the common neighbourhood."

SNN (dis)similarity has been used to replace the distance measure in DBSCAN as a way to overcome its inability to find all clusters of varying densities [9].

Let $s_k(x, y) = SNN(x, y)/k$, where $SNN(x, y)$ is the number of shared nearest neighbours of k nearest neighbours of x and y, which include both x and y; $SNN(x, y) = 0$ if both x and y are not included.

The neighbourhood function based on the SNN similarity can be expressed as:

$$M_\sigma(x) = \#\{y \in D \mid s_k(x, y) \geq \sigma\}$$

Note that $M_\sigma(x)$, like $M_\mu(x)$, cannot be treated as densities as the volume used to compute M_σ for every x is not a constant. In other words, we reveal that SNN clustering algorithm is a mass-based method, and not a density-based method as previously thought [9, 17]. This is despite the fact that SNN employs the same DBSCAN procedure by simply replacing the distance measure with the (inverse) SNN similarity [9, 17].

Using the same notation, let $R(x, y|H)$ be the (implicit) region which covers the shared nearest neighbours of x and y, where H is kNN. $SNN(x, y)$ can be defined as follows:

$$SNN(x, y) = |R(x, y|H)| \tag{5}$$

An example of $M_\sigma(x)$ due to SNN is given in Figure 4(d).

The advantages of using $iForest$ instead of k nearest neighbour to estimate the neighbourhood mass are:

- The SNN similarity matrix is sensitive to the parameter of neighbourhood list size k. In contrast, $iForest$ works well with a default setting.
- SNN clustering [9] has $O(k^2n^2)$ time complexity or $O(n^3)$ when $k = \sqrt{n}$ or larger and $n = |D|$. Yet, our solution (see Section 6.1) has the same $O(n^2)$ time complexity as DBSCAN, except an additional preprocessing to compute the dissimilarity matrix which takes $O(t \log \psi(\psi + n^2))$ or $O(n^2)$ since $\psi \ll n$.

6. APPLICATIONS TO THREE EXISTING ALGORITHMS RELYING ON DISTANCE

Here we show that the mass-based dissimilarity can simply replace the distance measure used in existing algorithms in three tasks: clustering, anomaly detection, and multi-label classification. The pertinent details are described in the following three subsections.

6.1 Density-based clustering

DBSCAN [10] is a natural choice not only because it is a commonly used clustering algorithm, but also it employs ϵ-neighbourhood density estimation. Here we convert DBSCAN to MBSCAN, i.e., from density based to mass based, by simply replacing distance measure with mass-based dissimilarity, leaving the rest of the procedure unchanged. This effectively changes the use of ϵ-neighbourhood density estimation to μ-neighbourhood mass estimation, as described in Section 4. This enables a global threshold to be used to identify all clusters of varying densities in hard distribution as shown in Figure 4.

6.2 k-nearest neighbour anomaly detection

Here we use one of the simplest anomaly detector which is based on k^{th} nearest distance [4]. The anomaly score for a test instance x is defined as the distance between x and its k^{th} nearest neighbour. Anomalies are instances which have the largest scores, i.e., they have the longest distances to their k^{th} nearest neighbours in a given data set.

This distance-based anomaly detector is used to show that a simple replacement of the distance measure with the mass-based dissimilarity can overcome its inability to detect local anomalies and improve its overall detection performance.

The replacement of ℓ_p with m_e effectively converts the entire operation from distance-based to mass-based. Anomalies are redefined as those which have the highest probability mass to their k^{th} lowest probability mass neighbours in a given data set.

The advantage of using mass is that fringes of any clusters with the same structure but different densities will have about the same probability mass. This enables (local) anomalies of dense clusters to be ranked at similar positions as anomalies of sparse clusters. The mass-based version is denoted as M-kNN [4] to contrast it with kNN.

6.3 Multi-label kNN classification

In multi-label classification tasks, each instance is associated with several class labels simultaneously. MLkNN [22] has been proposed to adapt the (single-label) kNN approach to multi-label problems, which makes a prediction via the maximum a posteriori rule. The rule is defined as follows.

In the context of multi-label learning, let $y_{i,j} \in \{1, 0\}$ be the j-th label of the i-th instance, where 1 and 0 indicate the instance's association and dissociation, respectively. The prediction rule of the j-th label for $\boldsymbol{x}_i$ is given by [22]:

$$y_{i,j} = \begin{cases} 1 & \text{if } \mathbb{P}(y_{i,j} = 1|c_{i,j})/\mathbb{P}(y_{i,j} = 0|c_{i,j}) > 1 \\ 0 & \text{otherwise} \end{cases},$$

where $c_{i,j}$ denotes the number of instances in the neighbourhood (i.e., k nearest neighbours) of $\boldsymbol{x}_i$ with the jth-label, $\mathbb{P}(y_{i,j} = 1|c_{i,j})$ is the posterior probability of $y_{i,j} = 1$ given that $\boldsymbol{x}_i$ has $c_{i,j}$ neighbours with the jth-label; and $\mathbb{P}(y_{i,j} = 0|c_{i,j})$ is similarly defined.

Bayes theorem yields that $\mathbb{P}(y_{i,j} = 1|c_{i,j}) \propto \mathbb{P}(y_{i,j} = 1)\mathbb{P}(c_{i,j}|y_{i,j} = 1)$, where $\mathbb{P}(y_{i,j} = 1)$ is the prior probability of $y_{i,j} = 1$ and $\mathbb{P}(c_{i,j}|y_{i,j} = 1)$ is the likelihood that $\boldsymbol{x}_i$ has $c_{i,j}$ neighbours when $y_{i,j} = 1$. Both $\mathbb{P}(y_{i,j} = 1)$ and $\mathbb{P}(c_{i,j}|y_{i,j} = 1)$ can be estimated via frequency counting on the training set. In the same manner, we can obtain $\mathbb{P}(y_{i,j} = 0|c_{i,j})$ via $\mathbb{P}(y_{i,j} = 0)$ and $\mathbb{P}(c_{i,j}|y_{i,j} = 0)$.

[4] Though there is a fundamental change of perspective, we have opted to reuse the same name of the original algorithm to emphasize that the same algorithm is employed with the exception of dissimilarity measure only.

Getting an effective closest match neighbourhood of a test instance is of prime importance in MLkNN, which directly influences the calculation of the likelihood. Thus, the dissimilarity measure employed plays a critical role in this process. In the original MLkNN, Euclidean distance is applied. The use of mass-based dissimilarity has the potential to enhance the nearest neighbour search and leads to a better multi-label classification result. We denote the version employing mass-based dissimilarity, M-MLkNN.

7. EMPIRICAL EVALUATION

7.1 Clustering

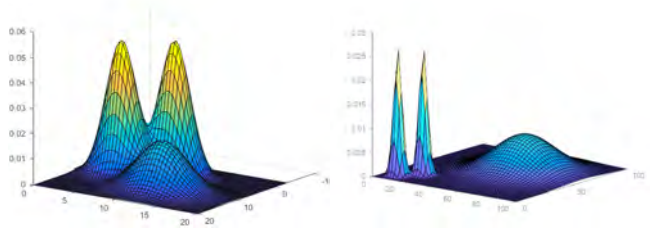
We evaluated the mass-based version of DBSCAN, named MBSCAN, and compared it with DBSCAN, SNN [9] and OPTICS [1]. Note that the only difference among DBSCAN, SNN and MBSCAN is the dissimilarity matrix, which is pre-processed and serves as input to these algorithms. We used $iForest$ with the default setting (i.e., $\psi = 256$ and $t = 100$) [15] to generate the mass-based dissimilarity matrix as the input for MBSCAN.

Th evaluation is conducted on 2 synthetic and 8 real-world datasets from UCI Machine Learning Repository [13]. Table 4 presents the properties of the datasets.

Table 4: Properties of clustering datasets

Dataset	Size	Dimensions	Clusters
Libras	360	90	15
WDBC	569	30	2
Thyroid	215	5	3
Segment	2310	19	7
Wine	178	13	3
Seeds	210	7	3
Pendig	10992	16	10
Iris	150	4	3
S1	900	2	3
S2	1500	2	3

S1 and S2 are synthetic datasets. S1 is a "hard distribution" which contains 3 Gaussian clusters $N(mean, std)$ with means located at $(x^{(1)}, x^{(2)})$=(3.3, 9.3), (8, 5), (12, 12), and standard deviations $std = 3, 3, 8$ in each dimension; and each cluster has 300 instances. S2 is an "easy distribution" which has 3 Gaussian clusters $N(mean, std)$ with means located at $(x^{(1)}, x^{(2)}) = (10, 10), (20, 20), (60, 60)$, and $std = 2, 2, 11$ in each dimension; and each cluster has 500 instances. The density plots of S1 and S2 are shown in Figure 5.



(a) S1: "hard distribution" (b) S2: "easy distribution"

Figure 5: Density distributions of S1 and S2.

We recorded the best F-measure[5] of a clustering algorithm on a dataset. Because $iForest$ is a randomised method, we reported the average result over 10 trials.

For each clustering algorithm, the search range of either ϵ, μ or σ was from the minimum to the maximum value of pairwise dissimilarity in the given dataset. The search range of $MinPts$ in DBSCAN, SNN and MBSCAN was in the range $\{2, 3, \ldots, 10\}$. The parameter k in SNN was set to the square root of the data size as suggested by some researchers [16]. For OPTICS, we searched $MinPts$ to produce the required hierarchical plots, and then searched threshold ξ[6] in the range $\{0.01, 0.02, \ldots, 0.99\}$ to extract clusters from each plot.

Figure 6 shows the best F-measure of DBSCAN, OPTICS, SNN and MBSCAN. The counts in the last column of the table reveal that MBSCAN and SNN performed the best in 5 and 3 datasets, respectively.

Table 5 shows the performance ratio of OPTICS, SNN, and MBSCAN with reference to DBSCAN. The geometric mean reveals that MBSCAN enhances DBSCAN the most by 27%; and SNN enhances DBSCAN by 21%.

Although MBSCAN and SNN have similar F-measures in many datasets, MBSCAN is significantly better than SNN in two datasets: S1 and WDBC. For example, MBSCAN enhances DBSCAN by more than 80% compared with less than 50% achieved by SNN on S1. Because S1 is a hard distribution for DBSCAN, this shows that mass-based dissimilarity provides a much better solution than shared nearest neighbour similarity in finding clusters of varying densities. Except for S1, WDBC and Thyroid, other datasets appear to have a lesser degree of hard distribution since their enhancements over DBSCAN are less than 40%.

The post-hoc Nemenyi test[7] reveals that both MBSCAN and SNN are significantly better than both DBSCAN and OPTICS. Figure 7 shows the average rank of each algorithm and its critical difference.

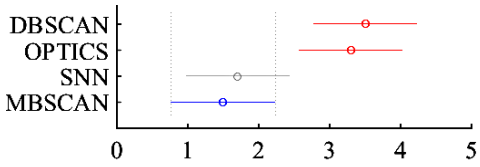
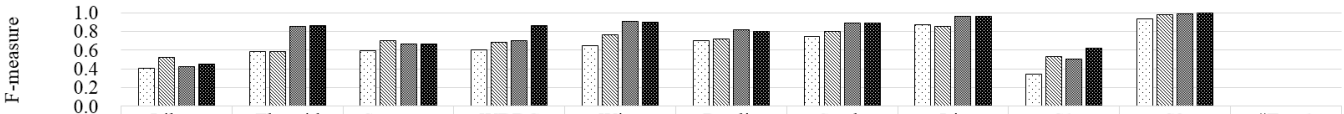


Figure 7: Critical difference (CD) diagram of the post-hoc Nemenyi test ($\alpha = 0.05$). The difference between two algorithms is significant if their CDs do not overlap.

[5]Given a clustering result, we calculate the precision score p_i and the recall score r_i for each cluster based on the confusion matrix, and then the overall F-measure is the average over all clusters: F-measure= $\frac{1}{m}\sum_{i=1}^{m}\frac{2P_iR_i}{P_i+R_i}$.

[6]Parameter ξ is used to identify downward and upward areas of a hierarchical plot in order to extract clusters. This hierarchical extraction method was proposed in the original OPTICS paper [1].

[7]This test (after the Friedman test) [8] is conducted to examine whether the performance difference between any two algorithms is significant. Firstly, the algorithms were ranked on each dataset according to their F-measures, where the best one is rank 1. Then, the post-hoc Nemenyi test is used to calculate the critical difference value (CD) for each algorithm.



	Libras	Thyroid	Segment	WDBC	Wine	Pendig	Seeds	Iris	S1	S2	#Top 1
DBSCAN	0.40	0.58	0.59	0.60	0.65	0.70	0.75	0.87	0.34	0.94	0/10
OPTICS	0.52	0.59	0.70	0.68	0.76	0.72	0.80	0.85	0.53	0.98	2/10
SNN	0.42	0.85	0.66	0.70	0.91	0.82	0.89	0.96	0.50	0.99	3/10
MBSCAN	0.45	0.86	0.67	0.86	0.90	0.80	0.889	0.963	0.62	0.993	5/10

Figure 6: Best F-measures of DBSCAN, OPTIC, SNN and MBSCAN on 10 datasets. The best performer on each dataset is underlined.

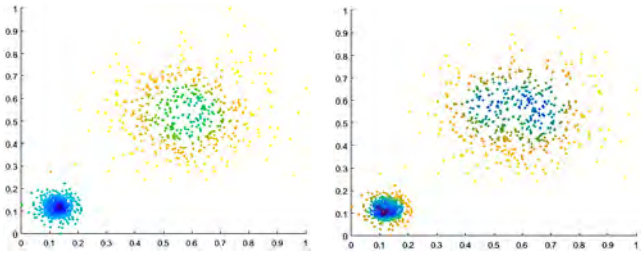
Table 5: Performance ratio of OPTICS, SNN, and MBSCAN with reference to DBSCAN on 10 datasets.

	Libras	Thyroid	Segment	WDBC	Wine	Pendig	Seeds	Iris	S1	S2	Geomean
OPTICS	1.30	1.02	1.19	1.13	1.17	1.03	1.07	.98	1.56	1.04	1.14
SNN	1.05	1.47	1.12	1.17	1.40	1.17	1.19	1.10	1.47	1.06	1.21
MBSCAN	1.13	1.48	1.14	1.43	1.38	1.14	1.19	1.11	1.82	1.06	1.27

7.2 Anomaly detection

In this section, we evaluate M-kNN in two experiments.

We first examined the ability of kNN and M-kNN to detect local anomalies on a synthetic dataset. The dataset contains one sparse cluster and one dense cluster, and each cluster has 500 instances. The instances at the fringes of the dense cluster are anomalies relative to this cluster only because they have higher densities than most instances in the sparse cluster; that is why they are called local anomalies.



(a) kNN (b) M-kNN

Figure 8: The ability to detect local anomalies in the dense cluster. Contour of the scores of k-nearest neigbour anomaly detectors using ℓ_p and m_e, with $k = 100$ on a synthetic dataset. The lighter the colour, the higher the anomaly score.

Figure 8 shows the contour of anomaly scores on the dataset. As expected, kNN is unable to detect local anomalies. Yet, M-kNN is able to detect all instances at the fringes of the dense cluster as anomalies. Note that the fringes of both dense and sparse clusters have similar high scores, allowing all of them to be identified as anomalies.

In the second experiment, we compared the performance on 6 real-world benchmark datasets[8]. The data size, dimensions and percentage of anomalies are shown in Table 6. A state-of-the-art local anomaly detector named Local Outlier Factor (LOF) [6] is also used in the comparison.

[8]Velocity is from http://openscience.us/repo/defect/ck/ and others are from UCI Machine Learning Repository [13].

Table 6: Properties of benchmark datasets

Dataset	Size	Dimensions	% Anomaly
Velocity	229	20	34.06%
Mfeat	410	649	2.44%
BloodDonation	604	4	5.63%
Diabetes	768	8	34.9%
annThyroid	7200	6	7.42%
p53Mutant	10387	5408	0.51%

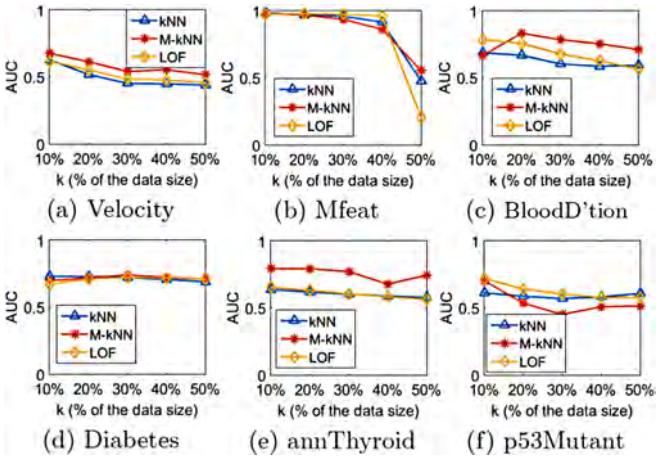


Figure 9: AUC of kNN, M-kNN and LOF.

We search k from 10% to 50% of the data size and present the result in terms of AUC (Area Under ROC Curve.)

M-kNN was run over 10 trials using different random seeds in computing the dissimilarity matrix for each dataset. kNN yields a deterministic result for a given data set; thus, it is run once only for each data set.

Comparing their best AUC values in Figure 9, M-kNN performs equivalent to or better than kNN on every dataset. M-kNN always outperforms kNN on Velocity and annThyroid with any k. On BloodDonation, M-kNN with a high k value performs better than kNN. Compared with LOF, M-KNN performs equally or better than kNN on 5 out of 6 datasets in terms of the highest AUC.

Table 8: M-MLkNN versus MLkNN on hamming loss, ranking loss, coverage, one error and average precision. Each result is an average over 10 trials of 10-fold cross-validation and its standard error. For the first four performance measures, the lower the better; for the last one, the higher the better.

		HammingLoss	RankingLoss	Coverage	OneError	AveragePrecision
Birds	MLkNN	0.051 ± 0.001	0.298 ± 0.004	3.507 ± 0.084	0.713 ± 0.017	0.392 ± 0.010
	M-MLkNN	**0.046** ± 0.001	**0.167** ± 0.007	**2.008** ± 0.067	**0.473** ± 0.006	**0.600** ± 0.005
CAL500	MLkNN	0.139 ± 0.001	0.187 ± 0.002	132.467 ± 0.489	0.131 ± 0.005	0.489 ± 0.002
	M-MLkNN	0.139 ± 0.001	0.187 ± 0.001	131.731 ± 0.538	0.125 ± 0.005	0.489 ± 0.001
Emotions	MLkNN	0.269 ± 0.007	0.278 ± 0.011	2.358 ± 0.071	0.419 ± 0.013	0.692 ± 0.011
	M-MLkNN	**0.208** ± 0.004	**0.180** ± 0.005	**1.857** ± 0.025	**0.309** ± 0.008	**0.776** ± 0.006
Enron	MLkNN	0.053 ± 0.000	0.099 ± 0.001	13.850 ± 0.095	0.340 ± 0.013	0.604 ± 0.005
	M-MLkNN	**0.052** ± 0.000	**0.094** ± 0.001	**13.477** ± 0.103	**0.288** ± 0.008	**0.640** ± 0.004
Scene	MLkNN	0.175 ± 0.002	0.198 ± 0.003	1.065 ± 0.023	0.338 ± 0.008	0.774 ± 0.003
	M-MLkNN	**0.168** ± 0.002	**0.175** ± 0.003	**0.978** ± 0.021	**0.310** ± 0.008	**0.794** ± 0.004

Table 7: Properties of multi-label datasets. W is the average number of labels per instance.

Dataset	Size	Dimensions	Labels	W
Birds	645	260	19	1.014
CAL500	502	68	174	26.044
Emotions	593	72	6	1.869
Enron	1702	1001	53	3.378
Scene	2407	294	6	1.074

7.3 Multi-label classification

In this section, we compare the original distance-based MLkNN[22] with the mass-based version M-MLkNN. The best k is searched in the range: 1,3,5,10,15 via 5-fold cross-validation on the training set.

We evaluate their performance in terms of Hamming loss, ranking loss, coverage, one error and average precision on five datasets[9], commonly used for multi-label classifier evaluations. Table 7 shows the properties of these datasets.

The result presented in Table 8 shows that M-MLkNN is significantly better (by two standard errors) than MLkNN in four out of the five datasets in terms of all five performance measures. The improvement of M-MLkNN over MLkNN is noteworthy (range between 30% and 50%) on the Birds dataset in terms of the last four measures.

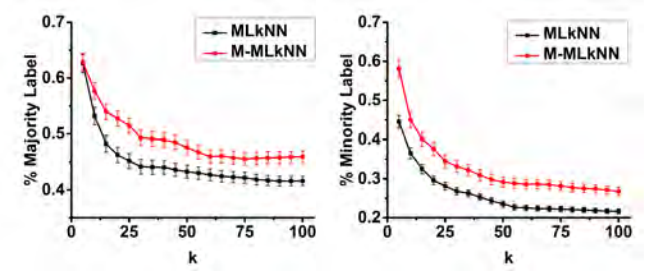
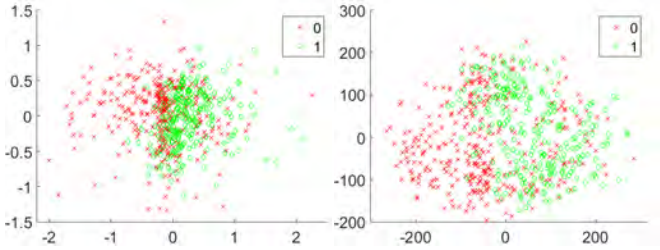


Figure 10: The proportion of k nearest neighbours having a particular label on the Emotions dataset for different k. The left and right figures show the proportions for the majority and minority labels (i.e., the labels which have the largest and smallest number of instances), respectively, on the dataset.

The improvement is due to the fact that mass-based dissimilarity provides a better closest match neighbourhood, where instances are more likely to hold the same label and the proportion of the majority label is larger. Figure 10 provides an insight into two of the labels individually. Each of the majority and minority labels (in the dataset) has a higher proportion in the k nearest neighbours found using mass-based dissimilarity than that using the distance measure. In fact, this occurs for every label on this dataset.

Visualisation provides another perspective of the advantage, i.e., how far apart instances are from one another according to each of the two dissimilarity measures. Here we performed multidimensional scaling (MDS)[10]. The MDS plot based on ℓ_2 is shown in Figure 11(a); and the one based on m_e is shown in Figure 11(b). It shows that instances of different labels are easier to separate using m_e.



(a) MDS using ℓ_2 (b) MDS using m_e

Figure 11: MDS plots using distance matrix and mass-based dissimilarity matrix on Emotions. Green and red points represent the positive and negative instances of the majority label, respectively.

7.4 Evaluation in runtime

The only difference between the ℓ_p and m_e versions of the algorithms in the above three subsections is the computation time for dissimilarity matrix. After the matrix is computed and served as input, the algorithm has the same runtime regardless of the dissimilarity used to compute the matrix.

Using ℓ_p to compute the dissimilarity matrix has $O(dn^2)$ time complexity. m_e builds $iForest$ and computes the dissimilarity matrix based on $iForest$, which yields $O(t\psi log\psi + n^2 tlog\psi)$ time complexity. For large datasets, $\psi \ll n$, the time cost is $O(n^2)$. The time complexity of SNN similarity is $O(n^3)$ in the worst case when $k = \sqrt{n}$ or larger. Table 9

[9]Source: http://mulan.sourceforge.net/datasets-mlc.html.

[10]Multidimensional scaling is a technique for visualising the information contained in a dissimilarity matrix [21]. An MDS algorithm aims to place each data point in a p-dimensional space ($p = 2$ is used here), while preserving as much as possible pairwise dissimilarities between them.

gives the time and space complexities of dissimilarity matrix calculation based on ℓ_p, m_e and SNN.

Table 9: Time and space complexities of dissimilarity matrix calculation based on ℓ_p, m_e and SNN.

Measure	Time complexity	Space complexity
ℓ_p	$O(dn^2)$	$O(dn)$
m_e	$O(t\psi log\psi + n^2 t log\psi)$	$O(t\psi log\psi + dn)$
SNN	$O(k^2n^2 + dn^2)$	$O(dn + kn)$

Table 10: Runtime of the dissimilarity matrix calculation for the three dissimilarities (in seconds).

Data set (Data size) (Dimension)	Segment (2310) (19)	annThyroid (7200) (6)	Pendig (10992) (16)	p53Mut (10387) (5408)
ℓ_p	5	42	110	8182
m_e	31	259	600	548
SNN	26	243	573	9141

Table 10 shows the runtime of the dissimilarity matrix calculation for the three dissimilarities on four real-world datasets. In small dimensional datasets, m_e has almost the same runtime as SNN, but takes longer to compute than ℓ_p due to the use of $iForest$. However, in large and high dimensional datasets such as p53Mutant, m_e is much faster than both ℓ_p and SNN because it is independent of the data dimensionality, i.e., each split node of a tree chooses one attribute randomly up to the certain height limit only.

8. DISCUSSION

Concepts. Dissimilarity measures are assumed to be a metric or a pseudo-metric as a necessary criterion for all data mining tasks. This work shows for the first time that this assumption is incorrect in three tasks.

We show that distance measures are the source of key weaknesses in three existing algorithms, highlighted in Table 1. Having recognised the source issue and created an effective alternative to distance measure, the solution becomes simple—merely replacing the distance measure with the data dependent dissimilarity; the otherwise unchanged algorithm can now overcome its weakness.

The result of not recognising the source issue can often lead to a solution which is more complicated than necessary and may not resolve the issue completely. An example is the inability of density-based clustering to find all clusters of varying densities. This issue is well-known and many suggestions have focused on density-based solutions [1, 9]. The fact that the ϵ-neighbourhood density estimator employed relies on distance measure, which is the source of the weakness, has been overlooked.

It is interesting to note that one of the existing solutions, i.e., SNN clustering has been incorrectly designated as density-based [9, 17] thus far. Our analysis in Section 5 has revealed that when replacing SNN (dis)similarity with the distance measure in ϵ-neighbourhood density estimator, the result is a mass estimator, not a density estimator.

Viewed from the conceptual perspective, simply changing the distance measure to a data dependent measure converts a density-based algorithm to a mass-based algorithm.

μ-neighbourhood mass can be viewed as a more general version of ϵ-neighbourhood density. However, the shape of ϵ-neighbourhood density region is fixed while the shape of μ-neighbourhood mass region depends on data distribution. We provide an example in Figure 3 that μ-neighbourhood mass has a regular shape region like ϵ-neighbourhood estimator in uniform density distribution only.

Implementations. An efficient implementation of m_p dissimilarity [2] employs a fixed-width one-dimensional histogram [3]. It partitions each attribute space into a fixed-number of cells independently, and the dissimilarity of any two instances is the total number of instances in the cells between the two instances in each attribute. In the experiments we have done in clustering and anomaly detection tasks, we found that using m_p with this implementation performs better than using distance on most datasets. However, their overall results are worse than those using m_e with the $iForest$ implementation.

The use of $iForest$ can be viewed as estimating probability from multiple variable-size multi-dimensional histograms.

Parameter ψ in iForest, used in the μ-neighbourhood estimator, is a smoothing parameter similar to k in a k-nearest neighbour density estimator. High ψ yields large trees which are sensitive to local variations in data distribution—similar effect of setting small k.

Since the default setting of $iForest$ ($\psi = 256$ and $t = 100$) can be used to provide good performance on many datasets, the implementation of mass-based dissimilarity based on $iForest$ does not create additional limitations, i.e., each existing algorithm, which has been transformed with the mass-based dissimilarity, has the same time complexity and the same number of parameters as in the original algorithm.

The generic formulation of mass-based dissimilarity allows different implementations, including different variants of $iForest$; and m_p dissimilarity [2] and SNN (dis)similarity are its special cases—all of them possess the characteristic of judged dissimilarity as prescribed by psychologists [12].

It is possible to use SNN in the contexts of kNN anomaly detection and MLkNN. However, its use has two issues. First, there are two k parameters as kNN is employed separately in the dissimilarity matrix calculation and the decision making process. Second, the high time complexity shown in Table 9 makes it prohibitive in large datasets.

Note that a mass-based neighbourhood function can be implemented using distance measure, as in the case of SNN. But, it is not only an indirect way to estimate mass but also an expensive one, as mentioned in Section 5.

Our implementation of data-dependent dissimilarity using trees opens up new research directions that worth investigating. kNN-based methods are traditionally regarded as distance-based methods; they become tree-based methods, though still employ dissimilarity measures, as we have shown here. The distinction between tree-based and distance-based methods are not clear-cut any more. Tree-based methods are usually thought to be less amenable than distance-based methods in dealing with high-dimensional datasets, especially when they consists of mostly relevant attributes. This research raises the question whether this is still true when data-dependent dissimilarity is used.

9. CONCLUDING REMARKS

We introduce a generic mass-based dissimilarity which is readily applied to existing algorithms in different tasks. The data dependent dissimilarity implemented with $iForest$ overcomes key weaknesses of three existing algorithms that rely on distance, and effectively improves their task-specific

performance on density-based clustering, kNN anomaly detection and multi-label classification.

These existing algorithms are transformed by simply replacing the distance measure with the mass-based dissimilarity, leaving the rest of the procedures unchanged.

As the transformation heralds a fundamental change of perspective in finding the closest match neighbourhood, the converted algorithms are more aptly called lowest probability mass neighbour algorithms than nearest neighbour algorithms, since the lowest mass represents the least dissimilar.

In the future, we will further explore other implementations of data dependent dissimilarity and investigate their influence in different data mining tasks.

10. ACKNOWLEDGMENTS

This material is based upon work partially supported by the Air Force Office of Scientific Research, Asian Office of Aerospace Research and Development (AOARD) under award number: #FA2386-13-1-4043 and 111 Project (B14020) (Kai Ming Ting); and NSFC (61333014) and the Collaborative Innovation Center of Novel Software Technology and Industrialization (Zhi-Hua Zhou). The anonymous reviewers have provided many helpful suggestions to improve this paper.

11. REFERENCES

[1] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. OPTICS: ordering points to identify the clustering structure. In *ACM Sigmod Record*, volume 28, pages 49–60. ACM, 1999.

[2] S. Aryal, K. M. Ting, G. Haffari, and T. Washio. m_p-dissimilarity: A data dependent dissimilarity measure. In *Proceedings of the IEEE International Conference on Data Mining*, pages 707–712, 2014.

[3] S. Aryal, K. M. Ting, G. Haffari, and T. Washio. Beyond tf-idf and cosine distance in documents dissimilarity measure. In *Proceedings of the 11th Asia Information Retrieval Societies Conference*, pages 400–406. Springer, 2015.

[4] S. D. Bay and M. Schwabacher. Mining distance-based outliers in near linear time with randomization and a simple pruning rule. In *Proceedings of the Ninth ACM International Conference on Knowledge Discovery and Data Mining*, pages 29–38. ACM, 2003.

[5] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[6] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. LOF: identifying density-based local outliers. In *ACM sigmod record*, volume 29, pages 93–104. ACM, 2000.

[7] B. Chen, K. M. Ting, T. Washio, and G. Haffari. Half-space mass: a maximally robust and efficient data depth method. *Machine Learning*, 100(2-3):677–699, 2015.

[8] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

[9] L. Ertöz, M. Steinbach, and V. Kumar. Finding clusters of different sizes, shapes, and densities in noisy, high dimensional data. In *Proceedings of the SIAM Data Mining Conference*, pages 47–58, 2003.

[10] M. Ester, H.-P. Kriegel, J. S, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231, 1996.

[11] R. A. Jarvis and E. A. Patrick. Clustering using a similarity measure based on shared near neighbors. *IEEE Transactions on Computers*, 100(11):1025–1034, 1973.

[12] C. L. Krumhansl. Concerning the applicability of geometric models to similarity data: The interrelationship between similarity and spatial density. *Psychological Review*, 85(5):445–463, 1978.

[13] M. Lichman. UCI machine learning repository, 2013.

[14] D. Lin. An information-theoretic definition of similarity. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 296–304, San Francisco, CA, USA, 1998. Morgan Kaufmann.

[15] F. T. Liu, K. M. Ting, and Z.-H. Zhou. Isolation forest. In *Proceedings of the IEEE International Conference on Data Mining*, pages 413–422, 2008.

[16] B. W. Silverman. *Density estimation for statistics and data analysis*, volume 26. CRC press, 1986.

[17] P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing, Boston, MA, USA, 2005.

[18] K. M. Ting, G.-T. Zhou, F. T. Liu, and J. S. C. Tan. Mass estimation and its applications. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 989–998, New York, NY, USA, 2010. ACM.

[19] K. M. Ting, G.-T. Zhou, F. T. Liu, and S. C. Tan. Mass estimation. *Machine Learning*, 90(1):127–160, 2013.

[20] A. Tversky. Features of similarity. *Psychological Review*, 84(4):327–352, 1977.

[21] F. Wickelmaier. An introduction to MDS. *Sound Quality Research Unit, Aalborg University*, 2003.

[22] M.-L. Zhang and Z.-H. Zhou. ML-KNN: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038–2048, 2007.

Appendix: Algorithm for random trees

iForest consists of t *iTrees*, each built independently using a subset $\mathcal{D}$, sampled without replacement from D, where $|\mathcal{D}| = \psi$. The maximum tree height $h = \lceil log_2 \psi \rceil$. Note that the parameter e in *iTree* is initialised to 0 at the beginning of the tree building process.

Algorithm 1 $iTree(X, e, h)$

Input: X - input data; e - current height; h - height limit.
Output: an *iTree*.

if $e \geqslant h$ OR $|X| \leqslant 1$ **then**
  return $exNode\{Size \leftarrow |X|\}$;
else
  Randomly select an attribute q;
  Randomly select a split point p between min and max values of attribute q in X;
  $X_l \leftarrow filter(X, q < p)$, $X_r \leftarrow filter(X, q \geqslant p)$;
  return $inNode\{ Left \leftarrow iTree(X_l, e+1, h),$
    $Right \leftarrow iTree(X_r, e+1, h),$
    $SplitAttr \leftarrow q, SplitValue \leftarrow p\}$;
end if