Pattern Recognition 47 (2014) 3070–3081

Contents lists available at ScienceDirect

Pattern Recognition

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

Information theoretic clustering using a *k*-nearest neighbors approach

Vidar V. Vikjord [a,*], Robert Jenssen [b]

[a] Microsoft Development Center Norway (MDCN), Tromsø, Norway
[b] Electrical Engineering Group, Department of Physics and Technology, University of Tromsø, Norway

ARTICLE INFO

Article history:
Received 19 March 2013
Received in revised form
17 February 2014
Accepted 18 March 2014
Available online 28 March 2014

Keywords:
Clustering
Scale
Entropy
Divergence
k-nn
Parzen windowing
Information theory

ABSTRACT

We develop a new non-parametric information theoretic clustering algorithm based on implicit estimation of cluster densities using the *k*-nearest neighbors (*k*-nn) approach. Compared to a kernel-based procedure, our hierarchical *k*-nn approach is very robust with respect to the parameter choices, with a key ability to detect clusters of vastly different scales. Of particular importance is the use of two different values of *k*, depending on the evaluation of within-cluster entropy or across-cluster cross-entropy, and the use of an ensemble clustering approach wherein different clustering solutions vote in order to obtain the final clustering. We conduct clustering experiments, and report promising results.

1. Introduction

Clustering is a fundamentally important component in pattern recognition and machine learning. There exists a vast literature on the subject, and there are numerous clustering methods, see e.g. [1,2] for general introductions, or [3–9] for recent specific work in this area. Many methods take a localized approach, in the sense of comparing distances between pairs of data points, such as e.g. single-link hierarchical clustering [10], and variants thereof [11]. The most well-known global approach is *k*-means, which optimizes a compactness criterion in terms of the variance of the clusters [12]. Hence, this method exploits only the second order statistics of the data.

Of key interest to this exposition is the recent development of global clustering cost functions based on *information theory*, such as entropy, divergence or mutual information, as such measures capture the higher order statistical information contained in the data [13]. An example illustrating where the first and second order statistics is not enough to distinguish between functions is shown in Fig. 1. Some of the first attempts in this direction include [14,15], suggesting a deterministic annealing approach, where sample points are associated with cluster representatives according to a maximum entropy probability distribution. A related approach for pairwise clustering was proposed by [16] based on mean-field approximation as minimization of the Kullback–Leibler divergence. Similar in spirit was the algorithm proposed by [17]. Minimization of the expected entropy of the partitions over the observed data was proposed in [18–20], using a Gaussian mixture model. More recently, an approach based on mutual information has received considerable attention. It is called the information bottleneck method [21], and is derived as a generalization to rate distortion theory. It has been widely applied [22–27]. For recent mutual information-based clustering approaches, see [28,29]. In the context of text classification, a clustering cost function based on the Jensen–Shannon divergence was proposed [30–32]. See also [33]. A common trait of the aforementioned information theoretic clustering methods is their parametric nature, in the sense that the form of the desired cluster probability density functions (pdfs) is specified.

Non-parametric information theoretic clustering (ITC), on the other hand, was proposed in [34,35], belonging to the broad family of *information theoretic learning* (ITL) methods (see [36–38] and references therein). The approach taken is to use gradient descent to globally optimize a divergence measure between clusters based on Renyi's second order entropy [39], implicitly using Parzen windowing [40] for non-parametric density estimation. Parzen windowing is also known as kernel density estimation [41], and involves the choice of a bandwidth, or smoothing, parameter. The results obtained were promising, but revealed a sensitivity to the choice of the kernel bandwidth, especially for clusters of different scale in the sense of the spread of the data within clusters. The problem was alleviated to some extent by kernel annealing, but resulted in a very slow procedure with several

* Corresponding author.
E-mail addresses: vidar.vikjord@outlook.com (V.V. Vikjord), robert.jenssen@uit.no (R. Jenssen).

http://dx.doi.org/10.1016/j.patcog.2014.03.018
0031-3203/

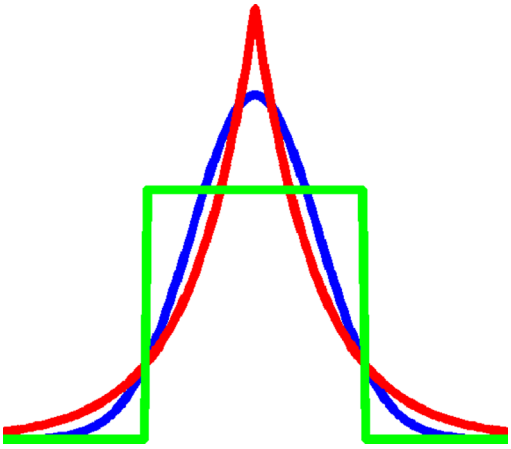

Fig. 1. Three distributions with the same mean and variance (first and second order statistics). Entropy measures are able to discriminate between them, despite the statistical similarities.

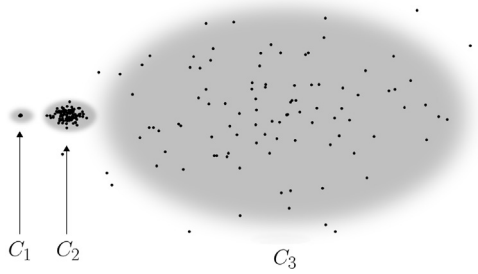

Fig. 2. Example of dataset with three clusters on different scales.

critical hyper-parameters. See Fig. 2 for a toy data example of a dataset comprised of three clusters with very different scale-properties exemplifying a type of dataset which is problematic for a clustering method based on a fixed kernel bandwidth choice throughout the data space.

In this work, we are interested in exploiting another well-known non-parametric density estimation method, namely the k-nearest neighbors (k-nn) approach [42], in the framework of information theoretic clustering [34,35]. The main motivation stems from the fact that k-nn is inherently a method which *adapts* to the local scale of the data, and which is therefore potentially more robust compared to Parzen windowing when clustering datasets wherein the scale of the clusters differs over the data space. Using k-nn for information theoretic measures has been shown to give asymptotically unbiased and mean-square consistent estimates [43,44]. See also [45].

In this paper we choose to focus on a hierarchical approach for optimizing a k-nn information theoretic cost function, as opposed to most other approaches to information theoretic clustering. There are benefits to such an approach, as the hierarchy itself may provide information about the structure of the data.

Our work includes novel contributions:

- We develop a k-nn ITC method, as opposed to a kernel-based approach (which underlies basically all other ITL methods in the framework of [36]). As a key property, we show that our method is able to detect and adapt to clusters of very different scales.
- Contrary to other k-nn approaches, we utilize *two* different values of k, depending on whether a *within-cluster* entropy is estimated, or an *across-cluster* cross-entropy is estimated. The two values of k are *fixed* throughout all experiments. There is no need for parameter tuning with respect to k.

In addition, the algorithm makes use of a form of *ensemble-based clustering*, in the sense that several possible clustering solutions, based on the value of the cost function, *vote* in order to obtain the final clustering result [46]. Very promising results are obtained. A hierarchical implementation of Parzen window-based information theoretic clustering is also implemented, enabling direct comparison with the proposed k-nn approach.

The literature on non-parametric information theoretic clustering with k-nn is quite limited. The work most closely resembling ours is [47], which developed a partitional algorithm optimizing mutual information based on Shannon entropy, using a particular kind of k-nn estimator, obtained by averaging over all k. We have experimented with a similar kind of averaging for the k-nn estimator in our hierarchical algorithm, but did not achieve satisfying results. Other somewhat related approaches are [48,49], each producing hierarchies of clusters.

The remainder of this paper is organized as follows. We begin in Section 2 by deriving the fundamental k-nn non-parametric pdf estimator. In Section 3, we first discuss the information theoretic quantities which will be involved in our final clustering routine and illustrate how these may be estimated. Next, we introduce our clustering algorithm, based on setting up an information theoretic cost function which is optimized using a hierarchical strategy. Results obtained in various clustering experiments are reported in Section 4. These results are contrasted to a similar algorithm using Parzen window estimates. Some discussions are provided in Section 5, and finally, Section 6 concludes the article.

2. Non-parametric density estimation using k-nn

Assume the d-dimensional dataset $\{\mathbf{x}_1,\ldots,\mathbf{x}_N\}$ is generated from a probability density function (pdf) $p(\mathbf{x})$. Making no parametric assumptions about the structure of $p(\mathbf{x})$, a k-nearest neighbors density estimate is given by

$$\hat{p}_{k-\mathrm{nn}}(\boldsymbol{x};k)=\frac{k}{NV_k(\boldsymbol{x})}. \tag{1}$$

Here, $V_k(\boldsymbol{x})$ is the hyper-volume centered on $\mathbf{x}$ whose radius *adapts* to the distance between $\mathbf{x}$ and its k'th nearest neighbor. For a hyper-sphere in d dimensions, the volume is given as

$$V(\boldsymbol{x})=\frac{\pi^{d/2}}{\Gamma(d/2+1)}\|\boldsymbol{x}_k\|_2^d \tag{2}$$

where $\Gamma(\cdot)$ is the gamma function and $\|\boldsymbol{x}_k\|_2$ is the Euclidean norm of the vector from $\mathbf{x}$ to its k'th nearest neighbor in the dataset.

Intuitively, the estimator $\hat{p}_{k-\mathrm{nn}}(\boldsymbol{x};k)$ is a generalization of the histogram. However, instead of a fixed bin size, the bin is now effectively given by $V_k(\boldsymbol{x})$, which varies over space, in order to capture the k'th nearest neighbors of $\mathbf{x}$. In general, there is no widely accepted criterion for selecting the value of k.

The k-nn estimator behaves very differently from a kernel density estimator [41]

$$\hat{p}_{\mathrm{kde}}(\boldsymbol{x};\sigma)=\frac{1}{N}\sum_{i=1}^{N} w_\sigma(\mathbf{x}-\mathbf{x}_i), \tag{3}$$

where the so-called Parzen window $w_\sigma(\mathbf{x}-\mathbf{x}_i)$ is a smooth fixed-width histogram bin, where the width is determined by σ. The Parzen window is a density itself, such as the Gaussian function. There exist rule-of-thumb methods for selecting the window size σ [50]. For multivariate data in d dimensions, one example is Silverman's rule of thumb

$$\sigma_{\mathrm{S}}=\sigma_{\mathrm{M}}\{4N^{-1}(2d+1)^{-1}\}^{1/(d+4)} \tag{4}$$

where σ_{M} is the mean value of the standard deviation estimate obtained from the data by maximum likelihood, $\sigma_M=d^{-1}\sum_{i=1}^{d}\hat{\Sigma}_{i,i}$,

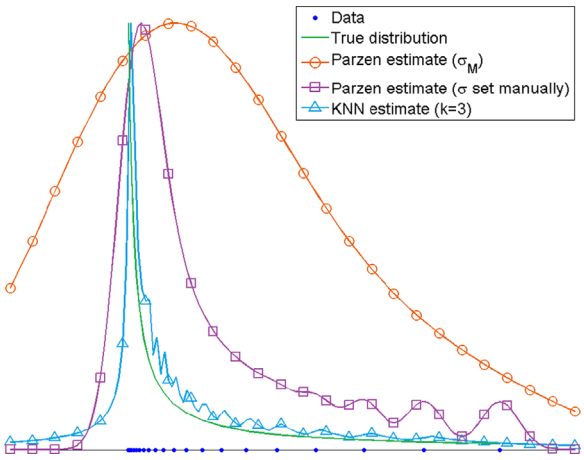

Fig. 3. Example of simple dataset where no single bandwidth parameter is able to capture the shape of the underlying pdf over the entire region. The k-nn estimate is seen to follow the true distribution more closely.

where $\hat{\Sigma}_{i,i}$ is the element in row i and column i of the empirical covariance matrix [51].

In order to illustrate some of the differences between these two non-parametric density estimation methods, consider Fig. 3. The one-dimensional data is shown along the horizontal axis. In the figure, the Parzen window estimate using the mean standard deviation of the data (σ_M) is shown as the (orange) line dotted with circles. Note that the estimate is smooth over the entire data space, but actually much too smooth, especially in the region where the data points are dense. For the (purple) estimate dotted with squares, the Parzen windows width parameter has been reduced. In this case, the estimate over the dense region improves. However, the width is now so small that the density estimate becomes very sensitive to the individual data points in the sparse region. Density estimation using k-nn ($k=3$) adapts the width of the volume capturing the k neighbors and we observe that the (blue) estimate dotted with triangles is able to follow the true density function quite closely, although only piecewise smooth. This is true both in the region densely and sparsely populated by data.

3. *k*-nn information theoretic clustering

In this section, we briefly provide the information theoretic learning background to the clustering approach. We then develop k-nn estimators for the quantities of interest comprising the clustering cost function. We briefly discuss the multi-class case and provide the optimization pseudo code for the clustering algorithm.

3.1. Brief background on information theoretic learning

The key quantity in information theoretic learning, as advocated in [36,52,53], is Renyi's second order entropy, given by

$$H(p) = -\log \mathcal{V}(p), \quad \mathcal{V}(p) = \int p^2(\mathbf{x})\, d\mathbf{x}. \tag{5}$$

One of the key reasons for this is the elegant estimation of $\mathcal{V}(p)$ (and hence $H(p)$) based on kernel density estimation using (3).

Of particular interest for clustering, is *divergence* measures based on Renyi's second order entropy. A divergence measure provides a means to quantify the closeness between two probability density functions $p_1(\mathbf{x})$ and $p_2(\mathbf{x})$. One such example is the so-called Cauchy–Schwarz (CS) divergence defined as $D_{CS}(p_1,p_2) = -\log J_{CS}(p_1,p_2)$, where

$$J_{CS}(p_1,p_2) = \frac{\int p_1(\boldsymbol{x})p_2(\boldsymbol{x})\, d\boldsymbol{x}}{\sqrt{\int p_1^2(\boldsymbol{x})\, d\boldsymbol{x} \int p_2^2(\boldsymbol{x})\, d\boldsymbol{x}}}. \tag{6}$$

Note that this measure is *symmetric*. For $p_1(\mathbf{x}) = p_2(\mathbf{x})$, it satisfies $J_{CS}(p_1,p_2) = 1$, and consequently $D_{CS}(p_1,p_2) = 0$.

The divergence $D_{CS}(p_1,p_2)$ may also be expressed in terms of Renyi entropy, since

$$D_{CS}(p_1,p_2) = -\log \int p_1(\mathbf{x})p_2(\mathbf{x})\, d\mathbf{x} - \tfrac{1}{2}H(p_1) - \tfrac{1}{2}H(p_2), \tag{7}$$

where $-\log \int p_1(\mathbf{x})p_2(\mathbf{x})\, d\mathbf{x}$ represents a cross-entropy measure [36].

Generalization to M different pdfs is possible with the straightforward extension

$$D_{CS}(p_1,\ldots,p_M) = -\log \frac{1}{\kappa_M} \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} D_{CS}(p_i,p_j), \tag{8}$$

where $\kappa_M = \binom{M}{2} = M(M-1)/2$ is a normalizing factor making the divergence measure invariant to the number of clusters.

3.2. Empirical divergence measure using k-nn

In order to utilize the adaptive nature of the k-nn estimator based on a set of data samples, we integrate the estimator into (6) (the multi-cluster extension is straightforward) to obtain an empirical k-nn-based Cauchy–Schwarz divergence measure. Assume in the following that the set $\mathcal{C}_1 = \{\mathbf{x}_i\}$ for $i = 1,\ldots,N_{p_1}$ is drawn from $p_1(\mathbf{x})$ and $\mathcal{C}_2 = \{\mathbf{x}_j\}$ for $j = 1,\ldots,N_{p_2}$ is drawn from $p_2(\mathbf{x})$.

1. *Estimating entropy using k-nn*: Note that $\mathcal{V}(p_1) = E_{p_1(\boldsymbol{x})}[p_1(\boldsymbol{x})]$. Using the sample mean estimator technique, we have $\hat{\mathcal{V}}(p_1) = (1/N_{p_1})\sum_{i=1}^{N_{p_1}} p_1(\boldsymbol{x}_i)$. Furthermore, when inserting the k-nn pdf estimator $\hat{p}_{1_{k-nn}}(\boldsymbol{x};k)$, given by (1) into this expression, we obtain

$$\hat{\mathcal{V}}(p_1) = \frac{k}{N_{p_1}^2} \sum_{i=1}^{N_{p_1}} \frac{1}{V_k^{(p_1)}(\boldsymbol{x}_i)}. \tag{9}$$

   To emphasize that the hyper-volume is computed with respect to the set $\{\mathbf{x}_i\}$ associated with $p_1(\mathbf{x})$ we use the notation $V_k^{(p_1)}(\cdot)$. An estimator for $\mathcal{V}(p_2)$ is obtained in a similar manner.
2. *Estimating cross-entropy using k-nn*: When estimating the cross-entropy, a similar approach as above is used, with some differences. Consider first $\int p_1(\boldsymbol{x})p_2(\boldsymbol{x})\, d\boldsymbol{x} = E_{p_2(\boldsymbol{x})}[p_1(\boldsymbol{x})]$. Using the sample mean approximation, we have

$$\begin{aligned}\int p_1(\boldsymbol{x})p_2(\boldsymbol{x})\, d\boldsymbol{x} &\approx \frac{1}{N_{p_2}} \sum_{j=1}^{N_{p_2}} p_1(\boldsymbol{x}_j) \\ &\approx \frac{1}{N_{p_2}} \sum_{j=1}^{N_{p_2}} \frac{k}{N_{p_1} V_k^{(p_1)}(\boldsymbol{x}_j)} \\ &= \frac{k}{N_{p_2}N_{p_1}} \sum_{j=1}^{N_{p_2}} \frac{1}{V_k^{(p_1)}(\boldsymbol{x}_j)}.\end{aligned}$$

   Here, $V_k^{(p_1)}(\boldsymbol{x}_j)$ denotes the hyper-volume spanned by the distance between the point $\mathbf{x}_j$ associated with $p_2(\mathbf{x})$ and its k'th nearest neighbor in the set $\{\mathbf{x}_i\}$ associated with $p_1(\mathbf{x})$.

Note that in general $E_{p_2(\boldsymbol{x})}[p_1(\boldsymbol{x})] \neq E_{p_1(\boldsymbol{x})}[p_2(\boldsymbol{x})]$ when performing the estimation based on k-nn. Hence, the estimation procedure introduces an asymmetry in the empirical CS divergence measure. For this reason, we modify the CS divergence to enforce symmetry

under k-nn estimation for $J = J_{CS}(p_1, p_2)$ as follows:

$$J = \frac{\frac{1}{2}\left(\int p_1(\mathbf{x}) p_2(\mathbf{x})\, d\mathbf{x} + \int p_2(\mathbf{x}) p_1(\mathbf{x})\, d\mathbf{x}\right)}{\sqrt{\int p_1^2(\boldsymbol{x})\, d\boldsymbol{x} \int p_2^2(\boldsymbol{x})\, d\boldsymbol{x}}}. \tag{10}$$

Empirically, this enables a k-nn estimate of the modified CS divergence as $\hat{J} = \hat{J}_{CS}(\mathcal{C}_1, \mathcal{C}_2)$, where

$$\hat{J} = \frac{\frac{1}{2}\left(\frac{k}{N_{p_2} N_{p_1}} \sum_{j=1}^{N_{p_2}} \frac{1}{V_k^{(p_1)}(\boldsymbol{x}_j)} + \frac{k}{N_{p_1} N_{p_2}} \sum_{i=1}^{N_{p_1}} \frac{1}{V_k^{(p_2)}(\boldsymbol{x}_i)}\right)}{\sqrt{\frac{k}{N_{p_1}^2} \sum_{i=1}^{N_{p_2}} \frac{1}{V_k^{(p_1)}(\boldsymbol{x}_i)} \frac{k}{N_{p_2}^2} \sum_{i=1}^{N_{p_2}} \frac{1}{V_k^{(p_2)}(\boldsymbol{x}_j)}}}. \tag{11}$$

This CS divergence estimator based on k-nn will serve as the basis for our novel non-parametric information theoretic clustering algorithm, as described in the next subsection. The extension to multiple clusters is straightforward following the form of (8).

3.3. Clustering strategy and optimization algorithm

The overall clustering strategy we take is the following:

The aim is to assign data points to clusters such that the CS divergence between the resulting cluster pdfs are maximized. The proposed k-nn procedure will be used in all divergence estimations.

In order to achieve our clustering aim, we take an iterative cluster-and-re-cluster approach creating a hierarchy of clustering assignments. To explain the rationale behind the method, assume first that a subset of the data points have already been assigned to groups (this could be done in various ways). Now consider an unassigned data point. Which of the already existing clusters should this data point be assigned to? Our approach is to assign this data point to the cluster which keeps the overall CS divergence as large as possible, which makes intuitive sense.

Assume furthermore that all the data points have been clustered into many clusters by the aforementioned approach. If one should remove one cluster, in order to reduce the number of clusters by one, for then to re-cluster its members, which cluster should we select? By intuition, once again, our approach is to identify the cluster which, when removed, results in the maximum CS divergence among on the remaining clusters.

Algorithm 1. k-nn information theoretic clustering (pseudocode).

1: **procedure** k-NN CLUSTERING ($K_{\text{init}}, N_{\text{init}}, K_{\text{end}}$)
2: Seed K_{init} clusters of N_{init} points each
3: Iteratively cluster unlabeled points minimizing $\hat{J}_{CS}$
4: $K_{\text{curr}} = K_{\text{init}}$
5: **while** $K_{\text{curr}} > K_{\text{end}}$ **do**
6: $\min_{\mathcal{C}_r} \hat{J}_{CS}(\mathcal{C} \backslash \mathcal{C}_r)$ ▷ Find cluster to remove
7: $\mathcal{C} = \mathcal{C} \backslash \mathcal{C}_r$ ▷ Shatter found cluster
8: $K_{\text{curr}} = K_{\text{curr}} - 1$
9: **for** $i = 1 : N_{\mathcal{C}_r}$ **do** ▷ Reassign points
10: Cluster $\boldsymbol{x}_i$: $\min_{\mathcal{C}_x} \hat{J}_{CS}(\mathcal{C}_1, \ldots, \mathcal{C}_x + \boldsymbol{x}_i, \ldots, \mathcal{C}_{K_{\text{curr}}})$
11: **end for**
12: **end while**
13: **Return** $\mathcal{C}$
14: **end procedure**

Our clustering algorithm starts by initializing a clustering on a subset of the data consisting of K_{init} clusters. This could be done by any number of clustering routines. We choose to seed K_{init} clusters of N_{init} points each by growing clusters from a set of seed points, at each step assigning the nearest point to any clustered point, to one of the initial groups. This creates the clustering $\mathcal{C} = \{\mathcal{C}_1, \ldots, \mathcal{C}_{K_{\text{init}}}\}$. Next, a cluster is removed, and its members are re-assigned into the remaining clusters. This reduces the number of clusters by one.

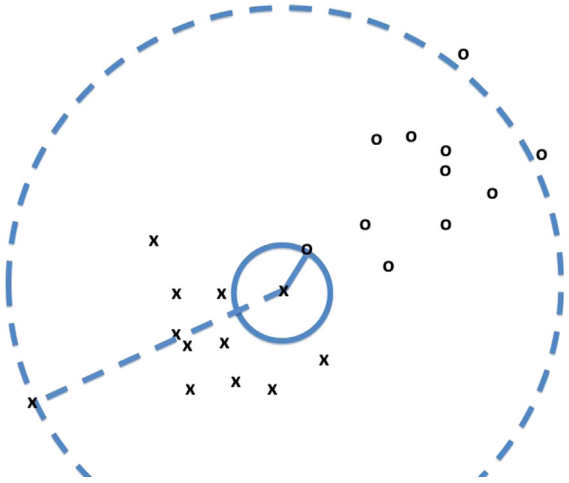

Fig. 4. For the proposed k-nn CS clustering, *two* different values of k are used. When estimating the cross-entropy, $k=1$ is used, illustrated by the small circle. When estimating the entropy within clusters, $k = k_{max}$ is used, where k_{max} is the farthest neighbor. This is illustrated by the stapled circle.

Table 1
Accuracy of one run of the clustering algorithm (no voting) as a function of initial parameters using the Wine dataset. The normalized cost function values are shown on the right. K_{init} governs how many clusters the algorithm starts with. The amount of preclustering adapts the size of each initial cluster (N_{init}) such that this percentage of the dataset is assigned a cluster before the algorithm starts. The gray scale background of the number reflects the accuracy obtained. As is seen, there is high correlation between the accuracy of the clustering and cost function evaluation.

	Accuracy					Divergence				
N_{init}	10%	30%	50%	70%	90%	10%	30%	50%	70%	90%
K_{init}										
3	0.89	0.84	0.90	0.90	0.83	0.63	0.59	0.68	0.68	0.45
5	0.91	0.87	0.94	0.97	0.89	0.74	0.69	0.82	0.93	0.74
8	0.87	0.96	0.97	0.98	0.97	0.72	0.95	1.00	1.00	0.81
10	0.97	0.96	0.97	0.97	0.99	0.92	0.90	0.91	0.91	0.96
15	0.94	0.98	0.97	0.96	0.95	0.83	0.87	0.89	0.92	0.78
20	0.99	0.98	0.97	0.98	0.96	0.97	0.97	0.97	0.96	0.93
25	0.95	0.97	0.96	0.97	0.98	0.79	0.90	0.86	0.84	0.97

The process is repeated iteratively as described in Algorithm 1, until K_{end} clusters are reached. This procedure assumes, K_{end}, is known.

Note that in lines 3 and 10 of Algorithm 1, data points are considered one at a time, for assignment to one of the clusters existing at the current iteration. The approach taken is to find the point closest to an already clustered point, and to assign that data point to the cluster which maximizes the divergence.

The way the proposed algorithm has been presented suggests that there are three parameters which have to be chosen, namely k, K_{init} and N_{init}. In our experience, the latter two parameters are not critical, as long as they are "reasonable"; the algorithm performs good for a wide range of K_{init} and N_{init} as long as K_{init} is not too small (giving the algorithm few re-clustering steps before termination) and N_{init} is not too small (giving the algorithm artificially small initial seeds of clusters). We will come back to this subsequently. However, the value of k is much more critical, in the same manner as σ is critical for Parzen window-based clustering [34]. In the next subsection, we discuss an approach which enables us to circumvent this problem completely.

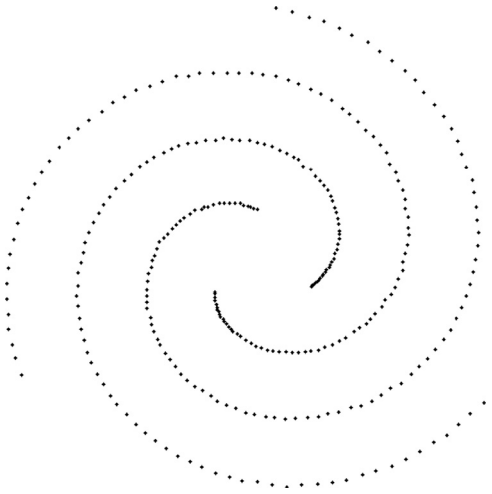

Fig. 5. Three elongated strings spiraling outwards. This dataset exhibits boundaries which are strictly nonlinear.

Table 2
Clustering accuracy. Comparison between k-nn method and Parzen window-based method based on two different choices of the free variable. The agglomerative hierarchical clustering method using the Wards method is included along with a Gaussian mixture model clustering and the standard K-means algorithm. All methods which are randomly initialized were run 50 times on each dataset. For the proposed k-nn method, the result using voting is shown along with the mean and standard deviation of the accuracy for all the runs. The Parzen based method has its mean accuracies shown (the standard deviations are roughly comparable to those seen for the k-nn method). For comparison with the potential capabilities of the GMM and K-means algorithms, the results shown for these methods are chosen using the runs with the highest accuracy according to the prior knowledge.

Dataset	ITC-methods			Other methods		
	k-nn ($\mu \pm std$)	Parzen		Wards	GMM	K-Means
		σ_S	σ_M			
Wine	**0.978**(0.92 ± 0.08)	0.867	0.923	**0.978**	0.933	0.955
Iris	**0.967**(0.94 ± 0.04)	0.920	0.573	0.887	0.596	0.887
WBC	0.955(0.79 ± 0.16)	0.696	0.695	**0.968**	0.963	0.961
Pima	**0.703**(0.66 ± 0.02)	0.660	0.656	0.684	0.657	0.677
Spirals	**1.000**(1.00 ± 0.00)	0.596	0.359	0.372	0.346	0.340

3.4. Two different values of k

Traditionally, when a cost function is estimated using k-nn as an integral part, one faces the problem of selecting k. The only exception, as far as the authors know, is the previously mentioned averaging taken by [47].

In the proposed clustering method, we take a different approach. When implementing the CS clustering using a single value k, we have experienced a certain sensitivity in the results obtained to the value of k, with typically one cluster growing out of bounds. To alleviate this, we propose as a novel aspect to use *two* different values of k, which are *fixed* throughout all experiments. The idea behind this is the observation that our CS cost function consists of two different quantities, which may not necessarily depend on k in the same manner.

On one hand, the k-nn procedure is used to estimate a cross-entropy via $\int p_1(\mathbf{x}) p_2(\mathbf{x})\, d\mathbf{x}$. Obviously, the region in space which is important to focus on when evaluating this integral across clusters is the boundary region between the clusters. In order to get a reliable estimate in the border region, we propose to use $k=1$ for the k-nn estimation of this quantity. That is, for any data point in the set $\{\mathbf{x}_i\}$ associated with $p_1(\mathbf{x})$, hyper-volumes are computed based on the nearest neighbor in the set $\{\mathbf{x}_j\}$ associated with $p_2(\mathbf{x})$. Interestingly, this resembles to some degree the idea used in single-link clustering [10], based on nearest neighbors.

When estimating the within-cluster entropy based on $\int p_1^2(\mathbf{x}) d\mathbf{x}$ $\left(\int p_2^2(\mathbf{x})\, d\mathbf{x}\right)$, one should strive to cover the whole region occupied by a cluster. We propose to achieve this by using $k = k_{max}$ for the k-nn estimation of this quantity, where k_{max} is the farthest neighbor to the data point in question. This resembles in a sense complete-link clustering [11], where analysis is carried out based on the maximum distances between points. See Fig. 4 for an illustration of the selection of two different k's in this context.

These choices we have experienced to give very high accuracy CS divergence clustering results. In fact, in all experiments these two values of k are fixed, which we consider to be a very positive property of the proposed method, compared to a kernel-based approach, which is typically sensitive to the choice of kernel bandwidth. This leaves only the less critical parameters K_{init} and N_{init} to be chosen by the user.

3.5. Ensemble clustering by voting

When reaching the sought number of clusters K_{end}, the final value of the CS clustering cost function may be computed. Since we initiate the clustering randomly, here by seeding K_{init} clusters, the final clustering may be different for repeated executions. This is the case for many clustering algorithms, including e.g. k-means. The solution in such cases is typically to run the algorithm several times, and to pick the final clustering as the one corresponding to the optimal cost function value.

Inspired by the ensemble procedure for clustering advocated by [46], we take a different approach. We execute the proposed CS divergence clustering several times, and let the clustering results of the best CS cost function values *vote* on each datapoint assigned label in order to obtain the final clustering. Voting helps guard against local-minima runs and it frequently happens that the final "voted" result is better than any of the individual results that participate in the vote. For the results presented in this paper where voting has been used, the 10% best clustering runs have been used (that is, if a total of 50 runs are performed, the 5 with the best associated cost is chosen to vote).

4. Experiments

In this section, we report clustering results using the proposed k-nn information theoretic clustering algorithm. Our primary focus is to highlight differences between the k-nn method and a similar approach using Parzen windowing for the estimation of the relevant quantities. Of particular interest is the ability of the k-nn method to handle clusters of very different scales, based on its inherent adaptive nature. We start by evaluating some common benchmark datasets often used in the clustering literature. Here we also include results from using k-means, Gaussian mixture models (GMMs) and Wards method in hierarchical cluster analysis. These are well established algorithms for clustering data. Next, we focus on some image datasets. Finally, and most importantly, we concentrate on datasets exhibiting clusters on vastly different scales.

First, the parameters K_{init} and N_{init} must be selected. In our experience, the k-nn ITC algorithm performs best when getting a chance to remove at least 5 clusters before K_{end} is reached. That way, there is some flexibility to adjust to the dataset over several re-clustering steps. At the same time, each initial cluster should not be made up of very few points, as the estimates of entropy

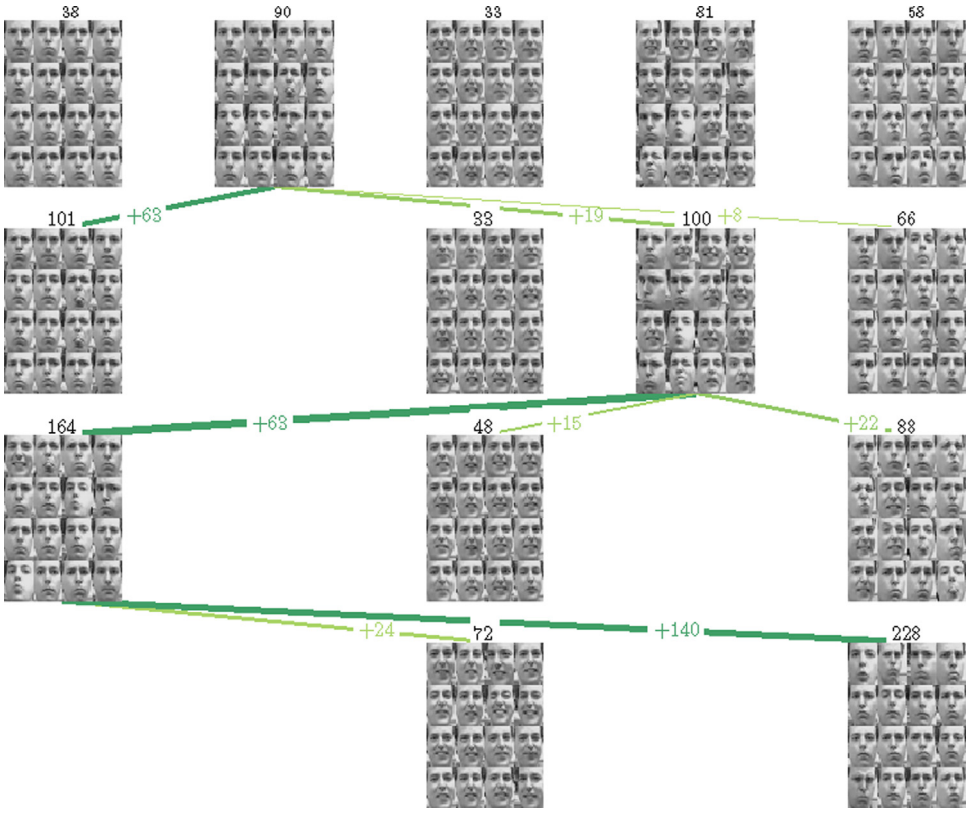

Fig. 6. Clustering hierarchy of faces. Final 4 steps of the algorithm shown where the number of clusters is reduced from 5 down to 2. For the complete end clustering result, see Fig. 7.

and cross-entropy will tend to be more unreliable for very small sample sizes. This creates a trade-off between K_{init} and N_{init} connected to the value of K_{end}. For K_{end} equal to 2 or 3, setting K_{init} to $10-12$ and choosing N_{init} such that approximately 80% of the dataset is included in the initial seeding, has been found to be a stable choice.

An experiment clustering the `Wine` dataset with different values for the starting parameters K_{init} and N_0 is shown in Table 1. The table shows the clustering algorithm to provide robust and comparable results as long as K_{init} is reasonably large such that the clusters may adapt to the dataset.

Note that for a Parzen window-based implementation, the free parameter σ must be selected before proceeding with the clustering. Since clustering is an unsupervised technique, there is no training dataset available to find a suitable value via cross-validation. One therefore must resort to rule-of-thumb methods for selecting this parameter. In this paper, the Parzen windowing is implemented using two different choices for σ: one is σ_S obtained from (4), and the second is σ_M, which appears in the same expression.

As explained previously, we employ the ensemble-based clustering approach which is described above. The algorithm is executed a number of times for each dataset (in our case 50) and the best 5 clustering results according to the CS cost function take place in the vote for the final clustering result.

The datasets are normalized to have all dimensions lie on the interval $[-1, 1]$.

4.1. Benchmark datasets

Datasets obtained from the UCI repository [54] are here considered, including `Wine`, `Iris`, `WBC` (Wisconsin breast cancer) and `Pima`. In addition, a synthetic 2-D dataset (see Fig. 5) consisting of 3 `spirals` is included to illustrate how each method handles highly nonlinear data.

Table 2 shows the results.[1] Using the fixed parameter values, k-nn ITC performs well. Since these datasets are well studied one may compare against alternative methods, showing that k-nn ITC performs at a high level compared to e.g. k-means and the more advanced methods [55,18,34,56]. When using the Parzen window-based ITC approach, results are not satisfying. The reason is that the single kernel rule-of-thumb widths do not capture the structure of the data. This illustrates the problem with clustering algorithms that are based on sensitive parameters which cannot be specified in a robust manner. The proposed k-nn ITC approach, on the other hand, does not exhibit this problem.

4.2. Face images

The previous datasets are widely used in clustering, and one knows in advance that there should be some group structure in the data. In the following, we analyze 300 randomly drawn, out of 1965 (28×20) "Frey faces",[2] previously used in manifold learning [57]. This is a dataset which cannot be easily divided into explicit classes, since the images represent the same person with varying facial expression, tilt and so on. The approach we take is to visualize the levels in the cluster hierarchy all the way down to two clusters. Due to lack of space, we concentrate on the final four levels, shown in Fig. 6. The number of elements in each cluster is indicated, and 16 randomly selected members are shown. The

[1] Note that the parameters used to obtain the results in Table 2 are fixed across all datasets, and not tuned specifically to one dataset such as is done for the Wine dataset in Table 1. This could explain why we see some accuracies in Table 1 being higher compared to the reported Wine accuracy in Table 2.

[2] Obtained from http://cs.nyu.edu/~roweis/data.html.

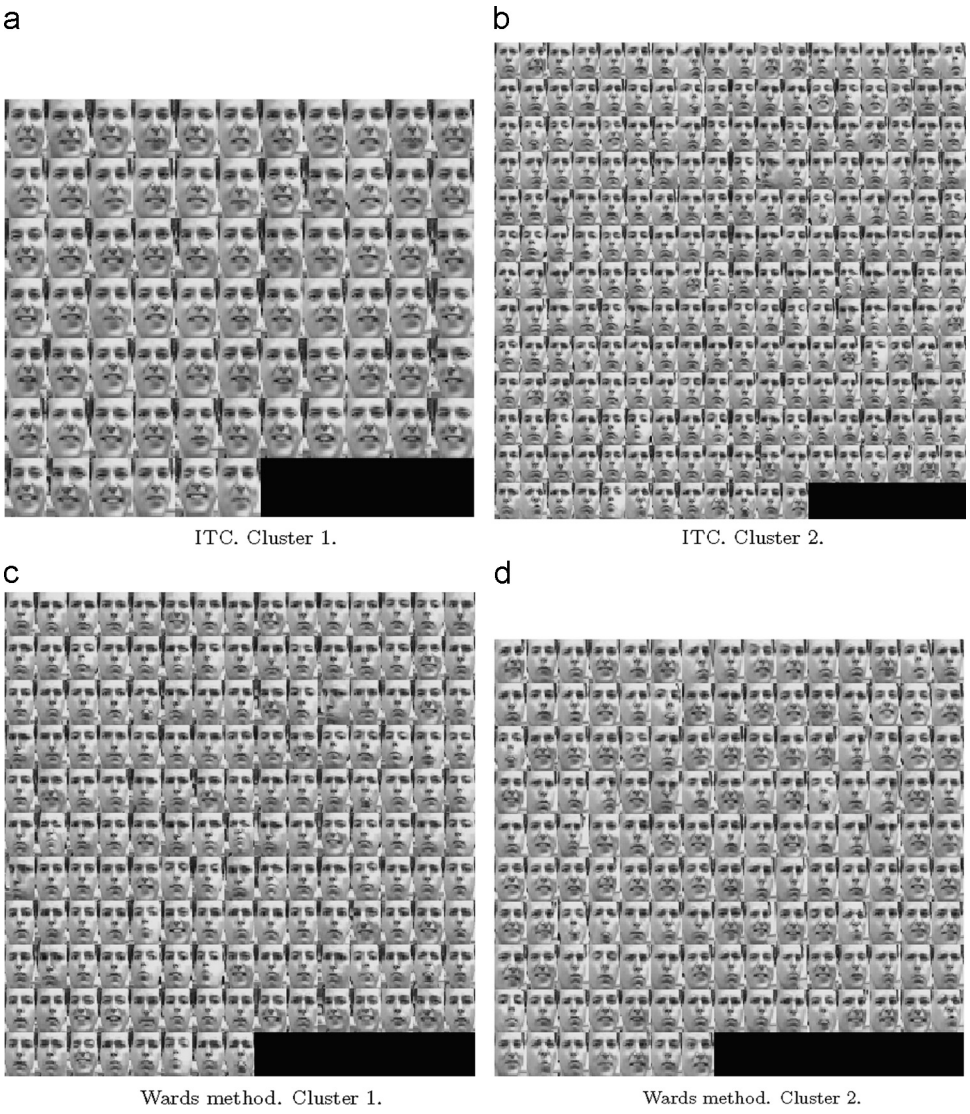

Fig. 7. Clustering of 300 randomly drawn Frey-faces using the proposed ITC method and the Wards method. For the ITC method, cluster 1 appears to always be smiling, while cluster 2 does not. Using the Wards method, the clusters are of more equal size with no clear pattern difference discerned between to the two. (a) ITC. Cluster 1. (b) ITC. Cluster 2. (c) Wards method. Cluster 1. (d) Wards method. Cluster 2.

cluster to be re-clustered distributes its members to the remaining clusters, and the number of points assigned to the other clusters is shown explicitly and represented by the thickness of the lines.

The hierarchy itself actually conveys information about the structure of the data. Consider for example the cluster with 33 members in the top row of Fig. 6. At this point in the process, this cluster consists predominantly of smiling faces. This is a stable cluster, in the sense that it is not chosen for re-clustering in any of the iterations shown in the figure. It is however fed smiling faces from the other clusters being shattered. Consider the third row from the top. At this point, the leftmost cluster is primarily of non-smiling faces with a tendency to look straight ahead or slightly to the left. The rightmost cluster consists mostly of non-smiling faces looking to the right or straight ahead. In the final two cluster grouping, one cluster represents only smiling faces, and the other contains the remaining predominantly non-smiling faces. We have also clustered this dataset using the Wards method in hierarchical clustering. Using this method did not uncover the same difference between the two end clusters. Instead, the two clusters appear to be of more similar size and somewhat separated based on the tilt of the head. The complete clustering results are shown in Fig. 7.

Note that for very high dimensional data, like the images examined here of $28 \times 20 = 560$ dimensions, the calculations for the hypervolumes involved in (2) can break down when working in finite precision. To circumvent this problem, we observe that the hypervolume is monotonically increasing as a function of distance for all dimensions. In terms of optimizing $\hat{J}_{CS}$ at each step in Algorithm 1, we can then simply drop the hypervolume calculations and solve the equivalent problem using only the distances.

4.3. Cloud screening in remote sensing

This example focuses on the issue of detecting clouds in optical remote sensing imagery. The clouds in effect correspond to missing data, since the surface of the earth is invisible to the imaging sensors. It is therefore important to be able to detect the clouds with high precision. The images are taken from the MERIS instrument on board the Environmental Satellite (ENVISAT). Six physically inspired features obtained from the MERIS bands were used [58]. The new k-nn ITC algorithm is used to find two clusters in the original image, taken over Spain (BR-2003-07-14) and shown in Fig. 8(a). In theory, the entire image could have been

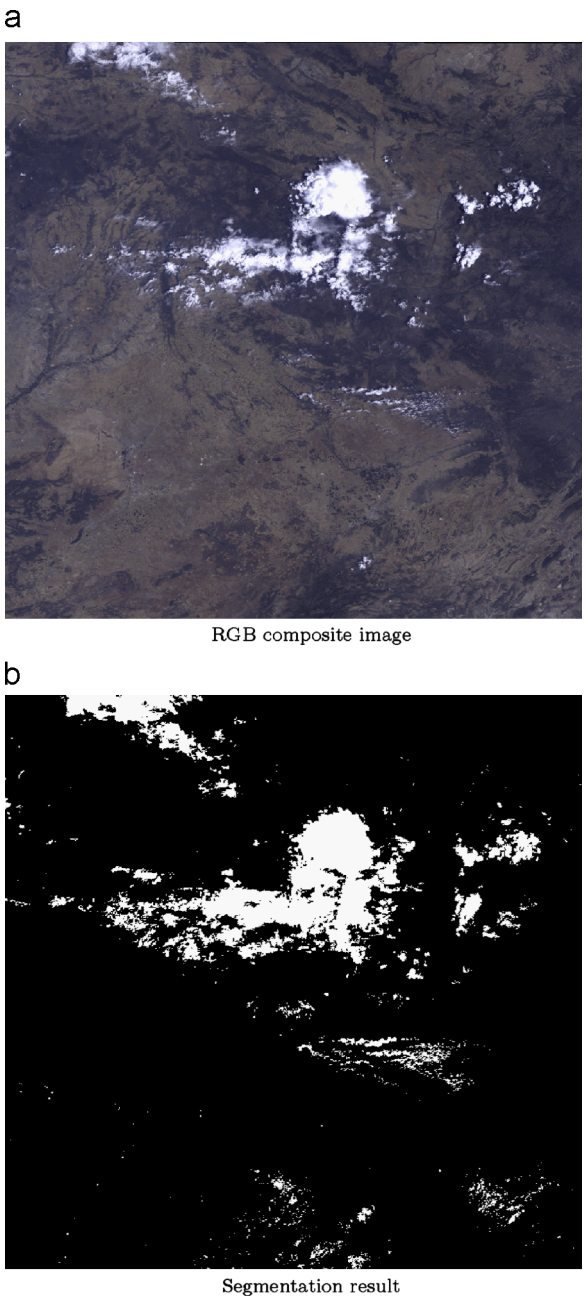

Fig. 8. Cloud screening in remote sensing using the proposed k-nn clustering technique. (a) RGB composite image. (b) Segmentation result.

clustered directly, but since it has a resolution of $\approx(1000 \times 1000)$, a direct approach becomes computationally demanding. Instead, the known labels of the image are used to draw 150 pixels of clouds and 150 pixels of land cover at random, similar to the set-up in [59]. This reduced dataset is then clustered, and the remaining points are assigned to the groups using a 1-nn classifier.

The result of using this procedure is shown in Fig. 8(b), corresponding to a success rate of 0.989. This is comparable to the best result reported in [59] obtaining an accuracy of 0.994. In that article, several other clustering attempts were also made, all obtaining worse results compared to the k-nn ITC. For instance, kernel k-means clustering gave an accuracy of 0.962. Lastly, it should be noted that the best result reported in [59] was obtained by tuning the bandwidth parameter over 5 orders of magnitude to find the best one (the process was therefore not entirely

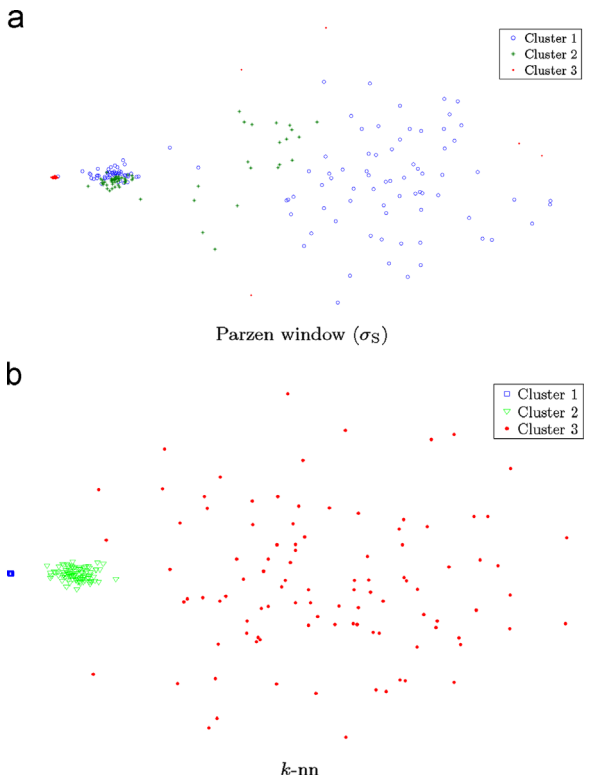

Fig. 9. Information theoretic clustering on dataset of 3 artificial clusters on vastly different scales. The Parzen window clustering in (a) is seen to struggle to distinguish the 3 clusters, while the proposed k-nn clustering in (b) gives good results. Using a Gaussian mixture model yields results similar to the k-nn method. (For interpretation of the references to color in this figure caption, the reader is referred to the web version of this paper.)

unsupervised). The proposed k-nn algorithm on the other hand requires no parameter tuning (besides the less critical seed parameters).

4.4. Robustness to scale

Our final set of clustering experiments analyze the performance of k-nn ITC when the data consists of clusters of vastly different scales. Handling such cases was the main motivation of this work in the first place.

First, we revisit the artificial dataset shown in Fig. 2. The dataset consists of a very tight cluster to the left, in the sense that the data points are very close together, one spread-out cluster to the right, and a third cluster in between. Obviously, the scales of the clusters are vastly different, where scale here refers to the spread of the data points. Fig. 9(a) shows the result of clustering this dataset using the Parzen window approach. The width of the window function is fixed throughout the data space using Silverman's rule. However, the data lives on very different scales, and the bandwidth is not able to capture the overall structure of the data. In fact, only the tight cluster is well represented. The k-nn clustering, on the contrary, is perfectly able to cluster the dataset into three clusters, and the result obviously makes a lot of sense, as shown in Fig. 9(b). Of key importance is the fact that there is no tuning of the number of neighbors. As before, we use the two fixed values of k, depending on which quantity that is estimated. This enables the method in essence to adapt its distance measure to the local structure of the data.

As further illustrations of k-nn ITC's robustness to scale, we consider two different color images. Consider first Fig. 9(a) displaying an RGB image consisting of two parts, where one part

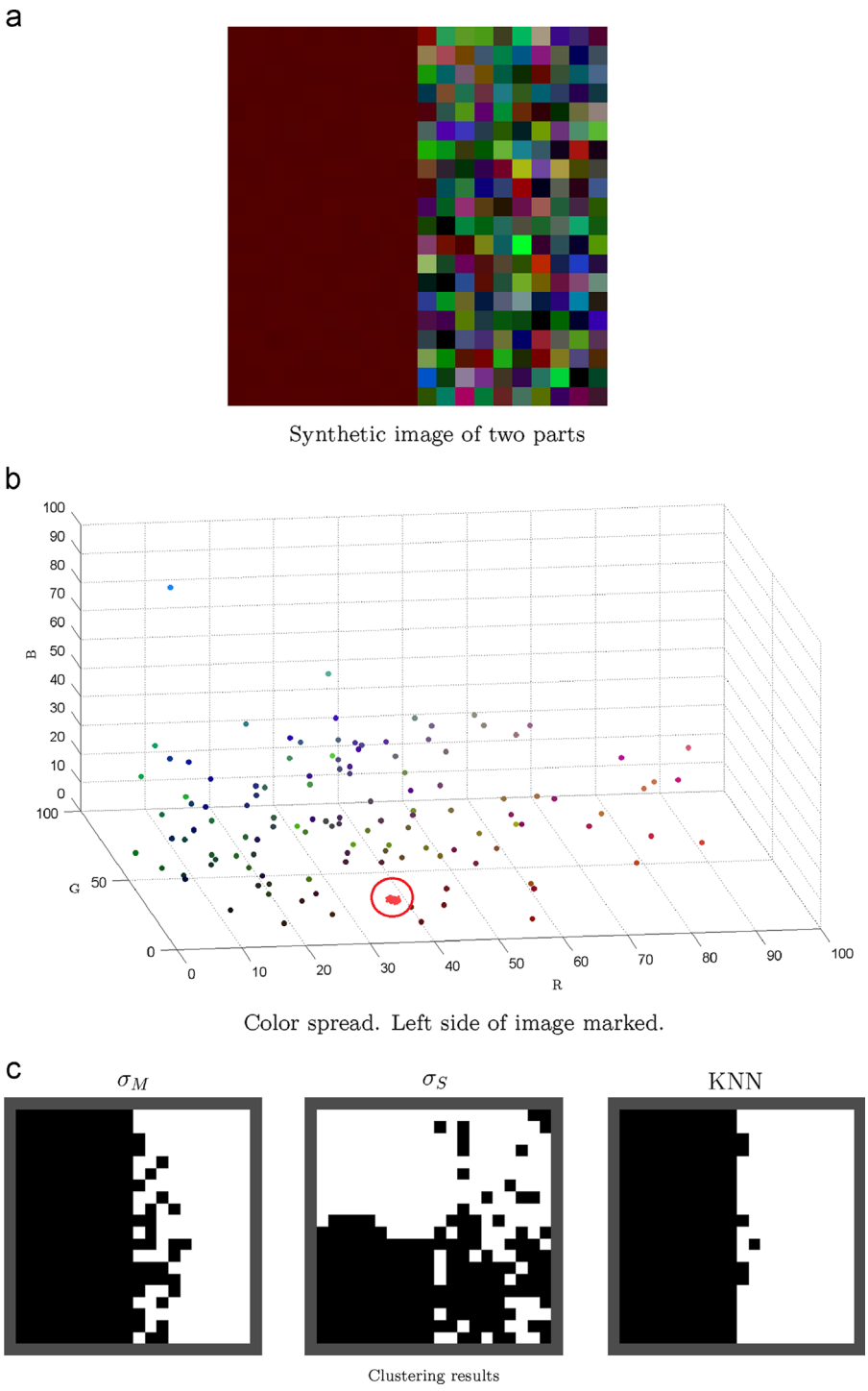

Fig. 10. Clustering experiment on synthetic image. The Parzen window method is allowed clustering attempt with both heuristic bandwidth choices. Both are unable to cluster the image satisfactory, while our proposed *k*-nn approach is seen to perform much better. Using a Gaussian mixture model yields results similar to the *k*-nn method. (a) Synthetic image of two parts. (b) Color spread. Left side of image marked. (c) Clustering results.

is monochromatic, while the colors of the other part are varying considerably (best viewed in colors). Clustering this image proves difficult for a Parzen window-based method, as the colors of one of the clusters are spread out, while the other cluster is very compact, i.e. the clusters live on different scales, as shown in (b) (tight cluster encircled for clarity). For each pixel, the three color channels as well as the pixels' position in the image are used as features for the clustering algorithms. This dataset is clustered using the Parzen window approach (using widths σ_M and σ_S) and *k*-nn. The clustering results are reported in (c). The result is by far the best for the *k*-nn clustering with only 6 pixels (out of 400) being assigned to the wrong cluster (Fig. 10).

It should be noted that clustering with the Gaussian mixture model yields results comparable to the proposed *k*-nn method on these two examples.

Consider next Fig. 11. In (a), the image displays a monochromatic gray object lying on a textured rug, while (b) represents a down sampled version of the same image. The colors of the textured region are varying greatly compared to the colors of the object. This can be seen in (c), where the RGB value of each pixel of the downsized image

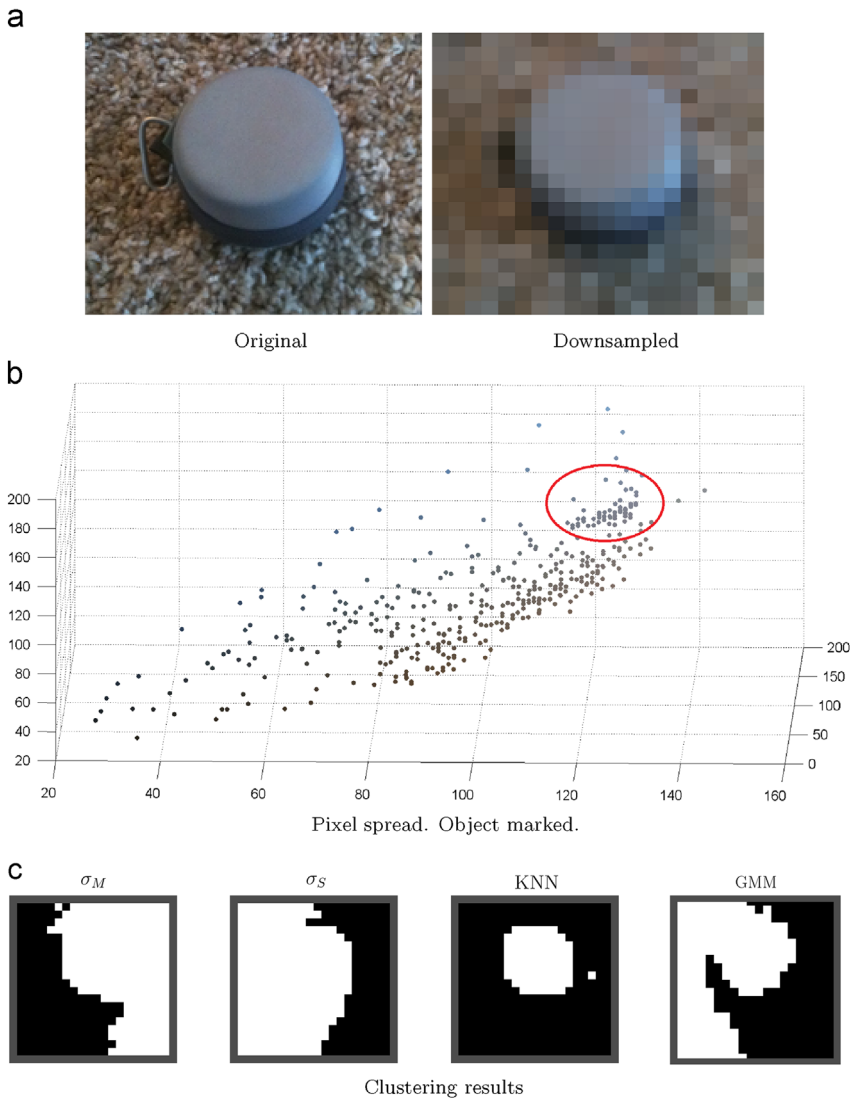

Fig. 11. Clustering experiment on image of a monochromatic object with a rough texture background. The Parzen window method is allowed clustering attempts with both heuristic bandwidth choices. Both bandwidth choices and the GMM method are unable to cluster the image satisfactory, while our proposed *k*-nn approach is seen to perform much better. (a) Original. (b) Downsampled. (C) Pixel spread. Object marked. (d) Clustering results.

constitutes a point in a three-dimensional space. The location of the gray object is encircled for clarity, illustrating that the object constitutes a tight cluster compared to the textured surface. Again, the dataset is clustered using the Parzen window approach (σ_M and σ_S) and *k*-nn ITC. The proposed *k*-nn method clearly provides the most appealing result, and again, it is the key ability to adapt to the local structure of the data which makes the difference.

For completeness we also included the results from a Gaussian mixture model method in the context of robustness to scale. This method is known to adapt to different scales and will for example be able to cluster the dataset shown in Fig. 2 satisfactory. The GMM method is however not able to handle nonlinear cluster structures very well. This is apparent e.g. from the clustering result on the three spirals dataset. Note also that this method fails to detect the object shown in Fig. 11.

4.5. Empirical analysis of runtime

We conclude this section with a brief discussion on the complexity and runtime of the proposed algorithm.

There is, as expected, a price to pay in terms of computations, when developing a clustering algorithm capable of capturing the higher-order statistics of the data, compared to e.g. a simple method like *k*-means. First of all, in our current implementation, the data is stored in the form of a matrix, requiring memory usage of the order $\mathcal{O}(N^2)$. Computation-wise, the main bottleneck is the computation and sorting of the pairwise distances in order to do nearest neighbor search. Assuming the sorting of distances from one datapoint to the $N-1$ others can be done in $\mathcal{O}(N \log N)$ time, the distance sorting for all N datapoints will run with a complexity of $\mathcal{O}(N^2 \log N)$.

The choice of K_{init} will also influence the runtime of the algorithm. This is because the parameter directly dictates how many times the outer loop has to run to reach K_{end}. In addition, at each step, the algorithm needs to consider all K_{curr} clusters to assign datapoints and to evaluate which cluster to shatter next (see Algorithm 1). In order to keep the runtime low, it is therefore advisable to keep K_{init} as low as possible, while still letting the algorithm adjust to the data.

An empirical analysis of the algorithms runtime is visualized in Fig. 12. Here, the time required to cluster a synthetic dataset of 3 equally sized clusters was investigated as the total number of datapoints increased. In the figure we observe that the runtime appears to follow the input size close to $\mathcal{O}(N^2 \log N)$. As is evident from the figure, a runtime complexity of this order proves

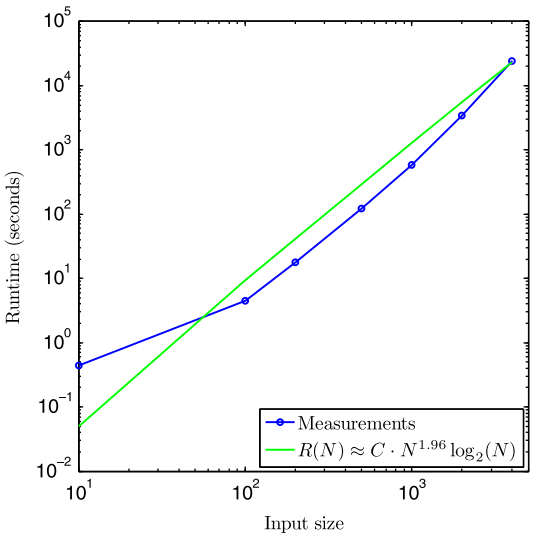

Fig. 12. Empirical investigation of the clustering algorithms runtime. The experiment was performed with a computer running on a 2.5 GHz Core i7 CPU with 8 GB of memory. K_{init} was fixed to 10 and N_{init} set such that 80% of the datapoints were preclustered. The constant factor C was here found to be approximately $\frac{2}{10^4}$.

problematic when encountering datasets of sizes above a couple of thousand points.

5. Discussion

We have presented our novel k-nn ITC algorithm, and have illustrated great potential, where we, without having to tune the parameter k, obtain very good results in general and importantly for the case of clusters of different scales, because of k-nn's adaptive nature.

A hierarchical approach has been chosen. There are several reasons for this. For example, the hierarchy of clusters may itself provide valuable information, as illustrated on the "Frey faces". However, another reason is that k-nn estimators are not smoothly varying, such that it is a challenge to optimize the CS cost function using gradient descent approaches.

We have seen that the runtime complexity of the clustering method makes it costly to apply on datasets of considerable size. However, it is worth noting that the optimization algorithm proposed can easily be run in parallel and distributes over several nodes. At lines 6 and 10 in Algorithm 1, the cost function considerations regarding each of the current cluster is calculated. Each of these calculations is independent from each other and can be distributed and calculated in parallel. Performing this parallelization will result in a factor K_{curr} calculation speed increase at each step in the algorithm.

Not being able to run on large datasets will limit the ability to accurately estimate pdfs in higher dimensions, which in turn degrades the information theoretic estimates used in the clustering cost function. However, as was first noted in [35], and later in [34], this cost function has proven to obtain positive results even in high dimensions with relatively few points of data. Consider for instance the clustering of face images presented. Here the algorithm is able to discern two meaningful clusters from a dataset of 300 points in 560 dimensions.

Another challenge is the utilization of the value of the cost function during iterations. As described above, the value of the CS cost function, when reaching the sought-for number of clusters K_{end}, is used to implement the ensemble clustering approach, which gives very satisfying results. In traditional hierarchical clustering, such as single-link [10], a so-called dendrogram is often produced, which may help determining the number of clusters in the dataset. For the proposed k-nn ITC hierarchical algorithm, we have experienced that it is not straightforward to compare the cost function at different levels in the hierarchy. In practice, for all but artificial datasets, we have not been able to generate a robust type of dendrogram which could enable the method to determine the most likely number of clusters. This is a challenge for our future research efforts.

Finally, we would like to mention that one could of course implement the voting scheme also for the alternative clustering methods (k-means, GMMs etc.) that we have displayed in this paper for completeness. In some cases, the voting may enhance the performance of these methods. However, if critical hyperparameters (e.g. the window width for the Parzen window-based method) do not represent the data well in the first place, or if the method fails to detect clusters of different scales, the voting scheme will have very limited effect.

6. Conclusion

In this paper, we have proposed a novel k-nn approach to information theoretic clustering, thus being able to capture the statistical properties of the data beyond second order moments. The main motivation was to develop a method capable of adapting to the local structure of the data, and hence being able to handle datasets consisting of clusters on very different scales. The proposed method performs well in that respect, as illustrated on several datasets. A huge benefit of the k-nn ITC method is that it operates without having to fine-tune critical bandwidth parameters. A novel aspect of our method enables this: the utilization of two different values of k, depending on whether estimating entropy ($k = k_{max}$), or cross-entropy ($k = 1$). We have also with success implemented an ensemble clustering approach, wherein promising individual clustering results according to the CS cost function vote in order to obtain the final result.

Challenges for future research include the development of a robust dendrogram, to help identify the number of clusters in the dataset, and the development of gradient descent-based optimization procedures.

Conflict of interest

None declared.

Acknowledgments

The authors would like to thank the anonymous reviewers for their insightful comments and suggestions which helped greatly in improving the paper. Dr. Luis Gomez Chova from the University of Valencia also has our gratitude for kindly providing the ENVISAT/MERIS dataset used in the cloud screening experiment.

References

[1] A.K. Jain, Data clustering: 50 years beyond k-means, Pattern Recognit. Lett. 31 (8) (2010) 651–666.
[2] R. Xu, D.C.W. II, Clustering, Wiley and Sons, Hoboken, NJ, USA, 2008.
[3] Y.G. Lu, Y. Wan, Pha: a fast potential-based hierarchical agglomerative clustering method, Pattern Recognit. 46 (5) (2013) 1227–1239.
[4] F. Xie, A. Bovik, Automatic segmentation of dermoscopy images using self-forming neural networks seeded by genetic algorithm, Pattern Recognit. 46 (3) (2013) 1012–1019.
[5] A.H. Kashan, B. Rezaee, S. Karimiyan, An efficient approach for unsupervised fuzzy clustering based on grouping evolution strategies, Pattern Recognit. 46 (6) (2013) 1240–1254.
[6] D.M. Vargas, F.J.G. Funes, A.J.R. Silva, A fuzzy clustering algorithm with spatial robust estimation constraint for noisy color image segmentation, Pattern Recognit. Lett. 34 (4) (2013) 400–413.

[7] F. Carvalho, Y. Lechevallier, F. Melo, Partitioning hard clustering algorithms based on multiple dissimilarity matrices, Pattern Recognit. 45 (1) (2012) 447–464.
[8] M. Herbin, N. Bonnet, P. Vautrot, A clustering method based on the estimation of the probability density function and on the skeleton by influence zones application to image processing, Pattern Recognit. Lett. 17 (11) (1996) 1141–1150.
[9] R. Amorim, B. Mirkin, M. Metric, Feature weighting and anomalous cluster initializing in *k*-means clustering, Pattern Recognit. 45 (3) (2012) 1061–1075.
[10] P.H.A. Sneath, R.R. Sokal, Numerical Taxonomy, Freeman, London, 1973.
[11] J.H. Ward, Hierarchical grouping to optimize an objective function, J. Am. Stat. Assoc. 58 (1963) 236–244.
[12] J. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, Berkeley, 1967, pp. 281–297.
[13] T.M. Cover, J.A. Thomas, Elements of Information Theory, John Wiley & Sons, New York, 1991.
[14] K. Rose, E. Gurewitz, G.C. Fox, A deterministic annealing approach to clustering, Pattern Recognit. Lett. 11 (11) (1990) 589–594.
[15] K. Rose, D. Miller, A. Gersho, Entropy-constrained tree-structured vector quantizer design by the minimum cross entropy principle, in: Proceedings of IEEE Data Compression Conference, Snowbird, Utah, March 29–31, 1994, pp. 12–21.
[16] T. Hofmann, J.M. Buhmann, Pairwise data clustering by deterministic annealing, IEEE Trans. Pattern Anal. Mach. Intell. 19 (1) (1997) 1–14.
[17] S. Shimoji, S. Lee, Data clustering with entropical scheduling, in: Proceedings of IEEE International Conference on Neural Networks, vol. 4, Orlando, USA, June 26–July 2, 1994, pp. 2423–2428.
[18] S.J. Roberts, R. Everson, I. Rezek, Minimum entropy data partitioning, in: Proceedings of IEE International Conference on Artificial Neural Networks, vol. 2, London, UK, September 7–10, 1999, pp. 844–849.
[19] S.J. Roberts, R. Everson, I. Rezek, Maximum certainty data partitioning, Pattern Recognit. 33 (2000) 833–839.
[20] S.J. Roberts, C. Holmes, D. Denison, Minimum entropy data partitioning using reversible jump Markov Chain Monte Carlo, IEEE Trans. Pattern Anal. Mach. Intell. 23 (8) (2001) 909–914.
[21] N. Tishby, F.C. Pereira, W. Bialek, The information bottleneck method, in: Proceedings of Annual Allerton Conference on Communication, Control and Computing, Monticello, USA, September 22–24, 1999, pp. 368–377.
[22] N. Tishby, N. Slonim, Data clustering by Markovian relaxation and the information bottleneck method, in: Advances in Neural Information Processing Systems, vol. 13, MIT Press, Cambridge, 2001, pp. 640–646.
[23] N. Slonim, N. Tishby, Agglomerative information bottleneck, in: Advances in Neural Information Processing Systems, vol. 12, MIT Press, Cambridge, 2000, pp. 617–623.
[24] S. Still, W. Bialek, L. Bottou, Geometric clustering using the information bottleneck method. in NIPS, 2003.
[25] D. Gondek, T. Hofmann, Conditional information bottleneck clustering, in: Workshop on Clustering Large Data Sets, IEEE International Conference on Data Mining, Melbourne, USA, November 19–22, 2003.
[26] N. Friedman, O. Mosenzon, N. Slonim, N. Tishby, Multivariate information bottleneck, in: Proceedings of Conference on Uncertainty in Artificial Intelligence, Seattle, USA, August 2–5, 2001, pp. 151–161.
[27] R. El-Yaniv, O. Souroujon, Iterative double clustering for unsupervised and semi-supervised learning, in: Advances in Neural Information Processing Systems, vol. 14, MIT Press, Cambridge, 2001, pp. 1025–1032.
[28] A.L.N. Fred, A.K. Jain, Robust data clustering, in: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Madison, USA, June 16–22, 2003, pp. 128–136.
[29] A. Kraskov, H. Stogbauer, R.G. Andrzejak, P. Grassberger, Hierarchical clustering based on mutual information, Bioinformatics, http://arxiv.org/abs/q-bio/0311039.
[30] I.S. Dhillon, S. Maella, R. Kumar, A divisive information-theoretic feature clustering algorithm for text classification, J. Mach. Learn. Res. 3 (2003) 1265–1287.
[31] I.S. Dhillon, S. Maella, R. Kumar, Information-theoretic co-clustering, in: Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington DC, USA, August 24–27, 2003, pp. 89–98.
[32] I.S. Dhillon, S. Maella, R. Kumar, Enhanced word clustering for hierarchical text classification, in: Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Canada, July 23–26, 2002, pp. 191–200.
[33] A. Banerjee, S. Merugu, I. Dhillon, J. Ghosh, Clustering with bregman divergences, in: Proceedings of SIAM International Conference on Data Mining, Lake Buena Vista, USA, April 22–24, 2004, pp. 234–245.
[34] R. Jenssen, D. Erdogmus, K.E. Hild, J.C. Principe, T. Eltoft, Information cut for clustering using a gradient descent approach, Pattern Recognit. 40 (2007) 796–806.
[35] E. Gokcay, J. Principe, A new clustering evaluation function using Renyi's information potential, in: Proceedings of International Conference on Acoustics, Speech and Signal Processing, Istanbul, Turkey, June 6–9, 2000, pp. 3490–3493.
[36] J.C. Principe, Information theoretic learning: Renyi's entropy and kernel perspectives, Information Science, 2010.
[37] R. Jenssen, Entropy relevant dimensions in kernel feature space, IEEE Signal Process. Mag. (2013), 30(4) 30–39, http://dx.doi.org/10.1109/MSP.2013.2249692.
[38] R. Jenssen, Kernel entropy component analysis, IEEE Trans. Pattern Anal. Mach. Intell. 33 (5) (2010) 847–860.
[39] A. Renyi, On measures of entropy and information, Selected Papers of Alfred Renyi, Akademiai Kiado, Budapest, vol. 2, 1976, pp. 565–580.
[40] E. Parzen, On the estimation of a probability density function and the mode, Ann. Math. Stat. 32 (1962) 1065–1076.
[41] M.P. Wand, M.C. Jones, Kernel Smoothing, Chapman and Hall, London, 1995.
[42] Q. Wang, S.R. Kulkarni, S. Verdú, Divergence estimation for multi-dimensional densities via *k*-nearest neighbor distances, IEEE Trans. Inf. Theory 55 (5) (2009) 2392–2405.
[43] Q. Wang, S.R. Kulkarni, S. Verdú, A nearest-neighbor approach to estimating divergence between continuous random vectors, in: 2006 IEEE International Symposium on Information Theory, IEEE 2006, pp. 242–246.
[44] F. Pérez-Cruz, Estimation of information theoretic measures for continuous random variables, in: NIPS, 2008, pp. 1257–1264. 〈http://books.nips.cc/papers/files/nips21/NIPS2008_0755.pdf〉.
[45] J. Barranquero, P. Gonzáilez, J. Díez, J.J. Coz, On the study of nearest neighbor algorithms for prevalence estimation in binary problems, Pattern Recognit. 46 (2) (2013) 472–482.
[46] A.L.N. Fred, A.K. Jain, Combining multiple clustering using evidence accumulation, IEEE Trans. Pattern Anal. Mach. Intell. 27 (6) (2005) 835–850.
[47] L. Faivishevsky, J. Goldberger, Nonparametric information theoretic clustering algorithm, in: Proceedings of International Conference on Machine Learning, June 21–24, Haifa, Israel, 2010, pp. 351–358.
[48] J.M. Santos, J.M. de Sa, L.A. Alexandre, LEGCLUST—a clustering algorithm based on layered entropic subgraph, IEEE Trans. Pattern Anal. Mach. Learn. 30 (1) (2008) 62–75.
[49] H. Li, K. Zhang, T. Jiang, Minimum entropy clustering and applications to gene expression analysis, in: Proceedings of IEEE Computational Systems Bioinformatics Conference, Stanford, USA, August 16–19, 2004, pp. 142–151.
[50] D.W. Scott, Multivariate Density Estimation, John Wiley & Sons, New York, 1992.
[51] B.W. Silverman, Density Estimation for Statistics and Data Analysis, Chapman and Hall, London, 1986.
[52] D. Erdogmus, J.C. Principe, From linear adaptive filtering to nonlinear information processing, IEEE Signal Process. Mag. 23 (6) (2006) 14–33.
[53] R. Jenssen, Information theoretic learning and kernel methods, in: F. Emmert-Streib, M. Dehmer (Eds.), F. Emmert-Streib and M. Dehmer, Information Theory and Statistical Learning. Springer, 2008. pp. 209-230 (chapter 9).
[54] R. Murphy, D. Ada, UCI Repository of Machine Learning databases, Technical Report, Deparment of Computer Science, University of California, Irvine, 1994.
[55] M. Girolami, Mercer kernel-based clustering in feature space, IEEE Trans. Neural Netw. 13 (3) (2002) 780–784.
[56] G. McLachlan, K. Basford, Mixture Models: Inference and Applications to Clustering, Marcel Dekker, New York, 1988.
[57] S. Roweis, L. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (2000) 2323–2326.
[58] L. Gomez-Chova, G. Camps-Valls, J. Calpe-Maravilla, L. Guanter, J. Moreno, Cloud-screening algorithm for ENVISAT/MERIS multispectral images, IEEE Trans. Geosci. Remote Sens. 45 (12) (2007) 4105–4118.
[59] L. Gomez-Chova, R. Jenssen, G. Camps-Valls, Kernel entropy component analysis in remote sensing image clustering, IEEE Geosci. Remote Sens. Lett. 9 (2) (2012) 312–316.

Vidar V. Vikjord received his Master of Science (MSc) in Data Analysis and Sensor Technology in 2012 at the University of Tromsø (UiT), Norway. He is currently employed at Microsoft Development Center Norway (MDCN), where he is working on large scale data analysis and machine learning within Office 365.

Robert Jenssen received the degree of Dr. Scient. (Ph.D.) in Electrical Engineering in 2005 from the University of Tromsø (UiT), Norway, where he is currently an associate professor at the Department of Physics and Technology. Jenssen is also an adjunct professor at the Norwegian Center for Telemedicine and Integrated Care, and is currently a guest researcher at the Technical University of Denmark (DTU Compute), at the Cognitive Systems Section, 2012/2013. Jenssen was a visiting guest researcher at the University of Florida, 2002/2003, and March/April 2004, and at the Technical University of Berlin, 2008/2009. In his research, he has focused on developing an information theoretic approach to machine learning based on Renyi entropy, with strong connections to Mercer kernel methods and to spectral clustering and dimensionality reduction methods. Jenssen received "Honorable Mention for the 2003 Pattern Recognition Journal Best Paper Award", the "2005 IEEE ICASSP Outstanding Student Paper Award" and the "2007 UiT Young Investigator Award". His paper "Kernel Entropy Component Analysis" was the Featured Paper of the May 2010 issue of IEEE Transactions on Pattern Analysis and Machine Intelligence, and the paper "Kernel Entropy Component Analysis for Remote Sensing Image Clustering", co-authored by Jenssen, was the Editor's Choice Paper of the March 2012 issue of the IEEE Geoscience and Remote Sensing Letters. Jenssen served on the IEEE Signal Processing Society's Machine Learning for Signal Processing Technical Committee 2006–2009, and is currently an Associate Editor of Pattern Recognition.