Information Sciences 584 (2022) 503–518

Contents lists available at ScienceDirect

Information Sciences

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

Adaptive Explicit Kernel Minkowski Weighted K-means

Amir Aradnia [a], Maryam Amir Haeri [b,*], Mohammad Mehdi Ebadzadeh [a]

[a] Department of Computer Engineering, Amirkabir University of Technology, Tehran, Iran
[b] Learning, Data-Analytics and Technology Department, University of Twente, Enschede, The Netherlands

ARTICLE INFO

Article history:
Received 28 December 2020
Received in revised form 13 October 2021
Accepted 19 October 2021
Available online 31 October 2021

Keywords:
Kernel clustering
Minkowski metric
Features map
K-means

ABSTRACT

The K-means algorithm is among the most commonly used data clustering methods. However, the regular K-means can only be applied in the input space, and it is applicable when clusters are linearly separable. The kernel K-means, which extends K-means into the kernel space, is able to capture nonlinear structures and identify arbitrarily shaped clusters. However, kernel methods often operate on the kernel matrix of the data, which scale poorly with the size of the matrix, or suffer from the high clustering cost due to the repetitive calculations of kernel values. Another issue is that algorithms access the data only through evaluation of $K(x_i, x_j)$, which limits many processes that can be done on data through the clustering task. This paper proposes a method to combine the advantages of the linear and nonlinear approaches by using derived corresponding approximate finite-dimensional feature maps based on spectral analysis. Applying approximate finite-dimensional feature maps have been discussed before only in the context of Support Vector Machines (SVM) problems. We suggest using this method in the kernel K-means context as it does not require storing a huge kernel matrix in memory, calculates cluster centers more efficiently, and accesses the data explicitly in the feature space; thus taking advantage of K-means extensions in that space. We demonstrate that our Explicit Kernel Minkowski Weighted K-means (Explicit KMWK-means) method is able to achieve high accuracy in terms of cluster recovery in the new space by applying additional Minkowski exponent and feature weights. The proposed method is evaluated by four benchmark data sets, and its performance is compared with the commonly used kernel clustering approaches. Experiments show the proposed method consistently achieves superior clustering performances while reducing the memory consumption.

1. Introduction

Clustering can be considered as the most important unsupervised learning problem. Clustering methods are used to determine the intrinsic grouping in a set of unlabeled data. It has been shown to be useful in many practical domains such as web search, image segmentation, image compression, gene expression analysis, recommendation systems, and mining text data [1–3]. K-means clustering [4] is one of the most popular conventional clustering algorithms, despite its age. It aims to partition a sample of N observations into K compact clusters in an iterative process. The K-means algorithm only works reasonably well when 1) clusters can be separated by hyper-planes and 2) each data point belongs to the closest cluster center. If one of these conditions does not hold, the standard K-means algorithm will likely not give a good result. Kernel-based

* Corresponding author.
E-mail addresses: a.mahdi@aut.ac.ir (A. Aradnia), m.amirhaeri@utwente.nl (M.A. Haeri), ebadzadeh@aut.ac.ir (M.M. Ebadzadeh).

https://doi.org/10.1016/j.ins.2021.10.048
0020-0255/

clustering methods overcome these limitations by using an appropriate nonlinear mapping to a higher dimensional feature space. Thus, it enables the K-means algorithm to partition data points by the linear separator in the new space that has nonlinear projection back in the original space. Various types of kernel-based methods such as the kernel version of the SOM (Self-organizing map) algorithm [5,6], kernel neural gas [7], one Class SVM (Support Vector Machines) [8] and kernel fuzzy clustering [9,10] have been proposed by researchers. In this paper, we focus on kernel K-means clustering because of its efficiency and simplicity. Furthermore, various studies [11–13] claim that different kernel-based clustering methods show similar result as kernel K-means.

Although kernel-based methods have received considerable attention from the machine learning community in recent years, they still suffer from the following problems in real-world applications. First, the high clustering cost due to either the repeated calculations of kernel values or requiring huge amounts of memory to store the kernel matrix makes it unsuitable for large corpora. Second, algorithms access the data only through the evaluation of $K(x_i, x_j)$. Therefore, many processes on the data points like fighting the concentration phenomenon, handling noise and so on are limited to work in the original space.

The aim of this paper is to develop a clustering method that can group data points with both linear and nonlinear structures while trying to address the two mentioned problems of kernel clustering methods. As the main contribution of this paper, we address both the space complexity of storing kernel matrix and lack of access to data in feature space by proposing a new *Adaptive Explicit Kernel Minkowski Weighted K-means (Explicit KMWK-means)* method. The proposed method combines the advantages of the linear and nonlinear approaches by using corresponding approximate finite-dimensional feature maps based on 1D Fourier analysis [14] for a large family of additive kernels, known as γ-homogeneous kernels. The proposed method first map data to feature space. This feature space is a data-independent approximation of γ-homogeneous kernels. Then, in order to provide a better fit to the cluster structure, the weighted version of Minkowski K-means in low-dimensional feature space is applied. Especially, we analyze the concentration of the Euclidean norm and the impact of distance measure on concentration. We investigate on Minkowski norms and fractional norms, an extension of the Minkowski norms with $p < 1$, as a measure of distance among data in the kernel space. Adaptive Minkowski metric allows to fight against possible concentration in high-dimensional spaces. In addition, the weighting property enables our algorithm to cover spherical and non-spherical (elliptical) structures in feature space and gets the chance of finding more complex clusters in feature space.

The rest of this paper is organized as follows. Section 2 briefly describes K-means and kernel K-means as the preliminary notions. Section 3 introduces Explicit Feature Maps and especially focuses on Homogeneous kernels. Section 4 presents our modified version of the kernel K-means by analyzing the alternative Minkowski distance for arbitrary $p \in \mathbb{R}^+$ instead of Euclidean one in the feature space. In Section 5, we describe the results of experiments for benchmark data sets. Finally, Section 6 concludes the paper.

2. Definitions and related work

In this section, we first outline the K-means algorithm and Then describe the Kernel K-means algorithm, which is proposed to address shortages of K-means. This section ends by reviewing methods that have been proposed to address common issues with Kernel algorithms.

2.1. K-means

The K-means method is designed to partition N D-dimensional samples $X = (x_1, x_2, \ldots, x_N)$. into K clusters $C_1, C_2, \ldots, C_K$ and return centroid vector for each cluster $M = (m_1, m_2, \ldots, m_K)$. The batch mode K-means algorithm would operate by the following iterative procedure:

1. Initialize K cluster center $m_1, m_2, \ldots, m_K$.
2. Assign each sample x_i to its closest center. Namely, compute the indicator matrix δ_{ik}, $(1 \leqslant k \leqslant K)$.

$$\delta_{ik} = \begin{cases} 1 & d(\mathrm{x_i}, \mathrm{m_k}) < d(\mathrm{x_i}, \mathrm{m_j}) \text{ for all } \mathrm{j} \neq k \\ 0 & otherwise \end{cases} \tag{1}$$

3. Update the cluster centers.

$$m_k = \frac{1}{|C_k|} \sum_{i=1}^{N} \delta_{ik} x_i \tag{2}$$

where $|C_k|$ is the number of samples in C_k.

4. Iterate between (2) and (3) until convergence.
5. Return $m_1, m_2, \ldots, m_K$.

Note that, in (1), $d(x_i, m_k)$ is the Euclidean distance given by:

$$d^2(x_i, m_k) = ||x_i - m_k||^2 \tag{3}$$

The preceding procedure actually is an iterative solution to optimization problem that attempts to minimize the objective function as follows:

$$\arg\min_{M,\delta}\sum_{i=1}^{N}\sum_{k=1}^{K}\delta_{ik}||x_i - m_k||^2 \tag{4}$$

In most cases, the distance in use is the squared Euclidean distance. However, the Euclidean distances tend to concentrate when data are high dimensional. This means that all the pairwise distances may converge to the same value. Accordingly, the relevance of Euclidean distance has been questioned in the past, and alternative norms, especially fractional norms (L_p semi-norms with $p < 1$) were suggested to reduce the concentration phenomenon [15,16]. Obviously, by using different metrics, cluster centers do not follow from the equation in (2) anymore. Finding fractional and Minkowski's centers, whose components are minimizers of the summary corresponding distances, are discussed in Section 4.1.3.

2.2. Kernel K-means

The K-means algorithm with Euclidean distance generally works on ellipse-shaped clusters. It is not applicable when elliptical regions do not hold. By applying some kind of transformation to the data, mapping them to some new space, the K-means algorithm may achieve better performance than in the original space. Again, suppose we are given N samples of $X = (x_1, x_2, \ldots, x_N)\, x_i \in \mathbb{R}^D$, and mapping function Φ that transforms x_i from original space $\mathbb{R}^D$ to a high dimensional feature space $\mathbb{H}$. Kernel functions are implicitly defined as the dot product of two vectors in the new transformed feature space.

$$K(x_i, x_j) = \Phi(x_i).\Phi(x_j) \tag{5}$$

In the rest of paper, we use Φ_i instead of $\Phi(x_i)$ for ease of description. Essentially, the transformation is defined implicitly, without knowing the concrete form of Φ [17]. Computation of distances in the transformed space is one of the most important issues when K-means is extended to the kernel K-means. The squared Euclidean distance between x_i and x_j in feature space would be as:

$$\begin{aligned} &\underset{Euc}{d^2}(\Phi_i, \Phi_j) = ||\Phi_i - \Phi_j||^2 = \Phi_i^2 - 2\Phi_i.\Phi_j + \Phi_j^2 \\ &= K(x_i, x_i) - 2K(x_i, x_j) + K(x_j, x_j) \end{aligned} \tag{6}$$

The cluster center in transformed space can be calculated as given below:

$$m'_k = \frac{1}{|C_k|}\sum_{i=1}^{N}\delta_{ik}\Phi_i \tag{7}$$

Therefore, the distance of each data point and the cluster center in new space can be computed without knowing the transformation Φ explicitly.

$$\begin{aligned} &d^2(\Phi_i, m'_k) = ||\Phi_i - \tfrac{1}{|C_k|}\sum_{j=1}^{N}\delta'_{jk}.\Phi_j||^2 \\ &= K(x_i, x_i) - \tfrac{2}{|C_k|}\sum_{x_{j\in C_k}}K(x_i, x_j) + \tfrac{1}{|C_k|^2}\sum_{x_{j\in C_k}}\sum_{x_{l\in C_k}}K(x_i, x_l) \end{aligned} \tag{8}$$

where δ' is the indicator matrix which δ'_{jk} indicates whether Φ_i is assigned to C_k or not. We will be moved from K-means to Kernel K-means by applying (8) to the standard K-means.

2.3. Kernel based clustering

As mentioned previously, One of the main problems in kernel-based clustering is the computation and storage of the full $n \times n$ kernel matrix K, which make it unsuitable for large scale data sets. The kernel-based algorithms that are scaled for big data have been extensively studied and applied to various applications. In [18–20] the authors reduce the memory requirement by writing kernel matrix in hard disk and moving only a small portion of the full matrix into memory each time as needed. Some other methods address this important limitation by using the approximate kernel matrix instead of computing the full one [21–23]. Quantization random features with the Gaussian kernel is used to store more features in the same amount of space in [24]. Over the years, a considerable research effort in further improvement of Kernel K-means has been made. In [25,26] authors propose adaptive kernel bandwidths in order to correct the density biases of the kernel K-means algorithm. Kernel fuzzy c-means (KFCM) [27] is another greatly improved edition of kernel K-means for clustering linearly inseparable data sets. To solve the special case of KFCM with fuzzification parameter $m = 1$, specialized versions of probabilistic kernel K-means have been proposed [28]. Multiple kernel learning or kernel-based fusion approaches have also been proposed for real-world applications, where multi-sources data or diverse feature descriptors need to be combined effectively [29–32]. In order to handle the noise in the data and the kernel, a robust multi-kernel learning strategy are proposed in [33,34]. The Kernel K-means algorithm accesses the data only through evaluation of $K(x, y)$. Due to this challenge, any pro-

cesses on the data points are limited to work in the original space or adjusting kernel parameters. Even though the explicit form of the mapping function is useful conceptually, it is not often used in computations. Typically, these feature spaces are infinite-dimensional, yet it is possible to find an efficient finite-dimensional approximation of Φ. There have been few attempts to construct an approximation of mapping function [14,35,36] for classification problems. In this paper, we study alternative approximate finite-dimensional feature maps to reduce the memory complexity of kernel K-means clustering. The choice of approximate explicit feature maps rather than implicit kernel has yet another interesting characteristic: accessing the data points in feature space is a unique opportunity to provide a better fit to cluster structure. The clustering quality can be improved by the weighted version of Minkowski K-means in low-dimensional feature space.

The following notation is used in the rest of the paper. Multidimensional additive kernels have been represented as $K(x_i, x_j)$, henceforth $k(x_i, x_j)$ will be used for the scalar ones. Multidimensional kernel function can be obtained from scalar ones as: $K(x_i, x_j) = \sum_{l=1}^{D} k(x_i^l, x_j^l)$. The scalar feature map is also denoted by $\varphi(x_i)$ and multidimensional ones are given by $\Phi_i = \bigoplus_{l=1}^{D} \varphi_i^l$, it means $\Phi_i = [\varphi_i^1, \dots \varphi_i^D]$.

3. Explicit feature maps

Namely, in kernel learning context, for each positive-definite (PD) kernel $K(x_i, x_j)$ there exists a corresponding mapping function Φ to an arbitrary dimensional space such that $K(x_i, x_j) = \Phi_i.\Phi_j$. Even though the explicit form of the mapping function is useful conceptually, it is not often used in computations. Typically, these feature spaces are infinite-dimensional, yet it is possible to find an efficient finite-dimensional approximation of Φ. In the following, we will introduce a class of kernels commonly used in computer vision, called homogeneous kernels. Then describe deriving corresponding approximate explicit finite-dimensional feature maps proposed by Vedaldi and Zisserman [14]. The main focus of this paper, as in an influential paper [14], is a class of additive kernels such as the Hellinger's, χ^2, intersection, and Jensen-Shannon ones, which are frequently used in computer vision applications. All of the mentioned kernels are mutual in two properties of additivity and homogeneity. Common homogeneous kernels with their properties are listed in Table 1.

Homogeneous Kernels. A kernel $k_h(a, b)$ called a homogeneous of degree γ if

$$\forall t \geqslant 0 : K_h(ta, tb) = t^{\gamma} K_h(a, b) \tag{9}$$

Here $a, b \in \mathbb{R}$. If we set $t = \frac{1}{\sqrt{ab}}$, the homogeneous kernel can be rewritten as:

$$\begin{aligned} k_h(a,b) &= t^{-\gamma} k_h(ta, tb) = (ab)^{\frac{\gamma}{2}} k_h(\sqrt{\tfrac{b}{a}}, \sqrt{\tfrac{a}{b}}) \\ &= (ab)^{\frac{\gamma}{2}} \kappa(\log b - \log a) \end{aligned} \tag{10}$$

The scalar function κ is called signature function which is used to deriving associated feature map of homogeneous kernel functions. It is defined as:

$$\kappa(\lambda) = k_h(e^{\frac{\lambda}{2}}, e^{-\frac{\lambda}{2}}), \ \lambda \in \mathrm{R}, \tag{11}$$

Bochner's theorem by using Fourier transform of signature function, $\kappa(\lambda)$ can address the problem of which mapping function made the given homogeneous kernel. As a result, the feature map φ for γ-homogeneous kernels will be derived as:

$$\varphi_{\omega}(a) = e^{-i\omega \log a} \sqrt{a^{\gamma} \rho(\omega)} \tag{12}$$

where ω can be viewed as the index of vector dimension and $\rho(\omega)$ is the spectrum function and can be obtained from inverse Fourier transform of the signature $\kappa(\lambda)$.

$$\rho(\omega) = \frac{1}{(2\pi)^D} \int_{R^D} e^{-i(\omega,\lambda)} \kappa(\lambda) d\lambda. \tag{13}$$

Table 1
Well-known kernel functions and their corresponding distance, signature and closed form feature maps. This table is adopted from [14].

Kernel	$K(x,y)$	$d(x,y)$	Signature $\kappa(\lambda)$	$\rho(\omega)$	$\varphi_{\omega}(x^i)$
χ^2	$\sum \frac{(x^i-y^i)^2}{(x^i+y^i)}$	$\sum \frac{2x^iy^i}{(x^i+y^i)}$	$\mathrm{sech}(\lambda/2)$	$\mathrm{sech}(\pi\omega)$	$e^{i\omega \log x^i} \sqrt{x^i \mathrm{sech}(\pi\omega)}$
Hellinger	$2\sum(\sqrt{x^i} - \sqrt{y^i})^2$	$\sum \sqrt{x^iy^i}$	1	$\delta(\omega)$	$\sqrt{x^i}$
JS	$\sum(x^i \log_2(\frac{2x^i}{x^i+y^i}) + y^i \log_2(\frac{2y^i}{x^i+y^i}))$	$\frac{1}{2}\sum(x^i \log_2(\frac{x^i+y^i}{x^i}) + y^i \log_2(\frac{x^i+y^i}{y^i}))$	$\frac{e^{\frac{\lambda}{2}}}{2} \log_2(1+e^{-\lambda}) + \frac{e^{\frac{-\lambda}{2}}}{2} \log_2(1+e^{\lambda})$	$\frac{2}{\log 4} \frac{\mathrm{sech}(\pi\omega)}{1+4\omega^2}$	$e^{i\omega \log x^i} \sqrt{\frac{2}{\log 4} \frac{\mathrm{sech}(\pi\omega)}{1+4\omega^2}}$
intersection	$\sum \lvert x^i - y^i \rvert$	$\sum \min(x^i, y^i)$	$e^{\frac{-\lvert\lambda\rvert}{2}}$	$\frac{2}{\log 4} \frac{\mathrm{sech}(\pi\omega)}{1+4\omega^2}$	$e^{i\omega \log x^i} \sqrt{\frac{2}{\pi} \frac{1}{1+4\omega^2}}$

In (12), feature maps are continuous functions, but still, finite approximation feature maps can be generated by sampling the continuous spectrum and rescaling it. Closed-forms of common kernel feature maps are described in Table 1. Moreover, Hein and Bousquet [37] introduced a large class of γ-homogeneous Hilbertian metrics and correspondent kernels, which encompasses all previously described kernels in Table 1. This class of metrics is defined as follows with two parameters α and β to be tuned.

$$d^2_{\alpha|\beta}(a,b) = \frac{2^{\frac{1}{\beta}}(a^\alpha + b^\alpha)^{\frac{1}{\alpha}} - 2^{\frac{1}{\alpha}}(a^\beta + b^\beta)^{\frac{1}{\beta}}}{2^{\frac{1}{\alpha}} - 2^{\frac{1}{\beta}}} \tag{14}$$

Function d in Eq. (14) is a γ-homogeneous metric on $\mathbb{R}^+$ with $\alpha \in [1,\infty]$ and $\beta \in [\frac{1}{2}, \alpha]$ or $\beta \in [-\infty, -1]$. The pointwise limit of d when $\alpha \to \beta$ is determined as:

$$\begin{aligned} &\lim \mathrm{d}^2_{\alpha|\beta}(a,b) = \tfrac{\beta^2 2^{\frac{1}{\beta}}}{\log(2)} \tfrac{\partial}{\partial\beta}\left(\tfrac{a^\beta+b^\beta}{2}\right)^{\frac{1}{\beta}} \\ &= \left(\tfrac{a^\beta+b^\beta}{\log(2)}\right)^{\frac{1}{\beta}}\left[\tfrac{a^\beta}{a^\beta+b^\beta}\log\left(\tfrac{2a^\beta}{a^\beta+b^\beta}\right) + \tfrac{b^\beta}{b^\beta+b^\beta}\log\left(\tfrac{2b^\beta}{a^\beta+b^\beta}\right)\right] \end{aligned} \tag{15}$$

Corresponding PD kernel for the above can be taken by:

$$k(a,b) = \frac{1}{2}(-d^2(a,b) + d^2(a,0) + d^2(b,0)) \tag{16}$$

4. Adaptive Explicit Kernel K-means

In this section, we present the *Adaptive Explicit Kernel K-means* method for homogeneous kernels. Homogeneous kernels have been introduced in the previous section, and a data-independent method [14] for deriving approximate finite-dimensional feature maps was described. This class of kernels is a frequently-used measure for histogram image comparison due to its effectiveness rather than linear kernel (Euclidean distance). Using explicit feature maps alleviates issues around storing huge similarity matrix or repeated calculation of kernel values. Moreover, we use the explicit feature map to take advantage of accessing the data in the feature space. We expect good matching between the K-means model and the real data structure of data in transformed space by choosing a suitable kernel, but we still have this chance to get a better fit by extending from the squared Euclidean to arbitrary weighted Minkowski metric in new space. In addition to adding more flexibility, the adaptive Minkowski metric allows fighting against possible concentration in high-dimensional spaces. Furthermore, weighting property with reflecting within-cluster feature variances enables our algorithm to cover spherical and non-spherical (elliptical) structures. In this way, features with smaller within-cluster variances receive a larger weight and features which more evenly distributed across the cluster get a smaller weight.

4.1. Minkowski metric

The distance we consider here is the Minkowski distance. It can be seen as a generalization of the Euclidean distance, which is defined as below:

$$d_{Mink}(x_i, x_j) = \left(\sum_{l=1}^{D} |x_i^l - x_j^l|^p\right)^{\frac{1}{p}} \tag{17}$$

Note that, when $0 < p < 1$, Eq. (17) does not hold metrics properties; therefore, it is not an actual metric, and the corresponding norm are not are indeed norms. They do not satisfy the triangle inequality. They are usually called prenorms or fractional norms.

4.1.1. Concentration in high-dimensional spaces

Losing the discriminative power of Euclidean distance to indexing points in high-dimensional spaces has been shown in the past. This means as dimension increases, the distance to the nearest point appears to be the same as the farthest one. This phenomenon is known as the concentration of distances. This inability of Euclidean distance to distinguish distances in high dimensions caused alternative distance measures to be suggested. In [15] a theoretical analysis of absolute difference between the farthest point distance d_{max} and the closest point distance d_{min} for Minkowski norms is presented. It points out that for D-dimensional i.i.d random vectors x_i, $(1 \leqslant i \leqslant N)$

$$C \leqslant \lim_{d\to\infty} \mathrm{E}\left(\frac{\max_i ||x_i||_p - \min_i ||x_i||_p}{D^{\frac{1}{p}-\frac{1}{2}}}\right) \leqslant (N-1).C \tag{18}$$

where C is some constant independent of the distribution of the x_i. This means the contrast between closest and farthest neighbor on average grows as $D^{\frac{1}{p}-\frac{1}{2}}$. The authors concluded that l_1 and l_2 norm may be more relevant than l_p when $p \geqslant 3$. In fact, for l_p with $(p \geqslant 3)$, the difference between the farthest and nearest neighbor goes to 0 as dimensionality increases.

These results encouraged researchers to examine fractional distances, l_p distance with $p \in (0, 1)$. In [16] authors extended previous works and proposed using fractional distance metric. It has been shown that fractional distance can provide higher relative contrast and meaningful result under same conditions as in (18).

The obtained results in [15,16] cannot be used in general when the data are not uniformly distributed. Indeed, it is quite rare that data spread through such spaces. Observation of real data shows that high dimensional spaces are mostly empty. In these spaces, it is common that data is spread on a sub-manifold. In [38] data distribution instead of data set has been studied. For that purpose, the relative variance ratio is proposed, which is defined as follows:

$$RV_{F,p} = \frac{\sqrt{Var(||x_i||_p)}}{E(||x_i||_p)} \tag{19}$$

Similarly to the relative contrast (18), the relative variance can be seen as a measure of concentration. Smaller values of $RV_{F,p}$ indicate less concentration. We can see that the shape of distribution F and the value of p might affect the value of the $RV_{F,p}$. Therefore, for adjusting the value of p, the shape of distribution should be considered too. It is completely possible that the higher relative variance is acquired by higher-order norms.

4.1.2. Choosing the optimal value of p

In supervised learning tasks like classification, the value of p could be chosen to maximize model accuracy. However, we do not have that access to the class labels in unsupervised learning. In that case, a sensible way of choosing p could be investigating on use of relative variance or entropy as an objective to be maximized [35,39]. Also, in [40] the relation between the phenomenon of concentration and hubness has been studied. The authors proposed an unsupervised approach for choosing the value of p by measuring hub and anti-hub occurrence as defined in the paper. Although $RV_{F,p}$ and hubness can give a sense of the concentration level, they are not excellent measures all the time. As shown in Fig. (1), a different value of p reflects a distinctive measure of distance in a Euclidean space. It can be seen, the rotation of the coordinate system will also lead to changes in the distance measurement. The only exception is the circular shape, $p = 2$ (Fig. 1d). By adopting $p \to 0$ being exact in one dimension has more value than have balance in two (Fig. 1a). Conversely, by approaching $p \to \infty$, just the maximum difference is matter (Fig. 1f). So by going far from $p = 2$, the distance meaning completely changes; therefore, the value of p should be chosen by considering both being meaningful and reduction of the concentration. However, ground truth annotation is often costly and inaccessible in real-world applications; there are usually limited available class labels. In this case, semi-supervised learning provides a better choice by leveraging unlabeled data by using a small set of labeled data. We discover the optimal value of exponent p, by uncovering only a few percents of data. We employ the entire data set, either labeled or not, to run a series of clustering experiments at various values of p as reported superior results rather than using only labeled data in [41].

4.1.3. Finding centers

After deriving the approximate feature map and an optimal value for p then the kernel clustering algorithm can be started in order to optimize the objective function with considering Minkowski distance. It minimizes the sum of Minkowski dis-

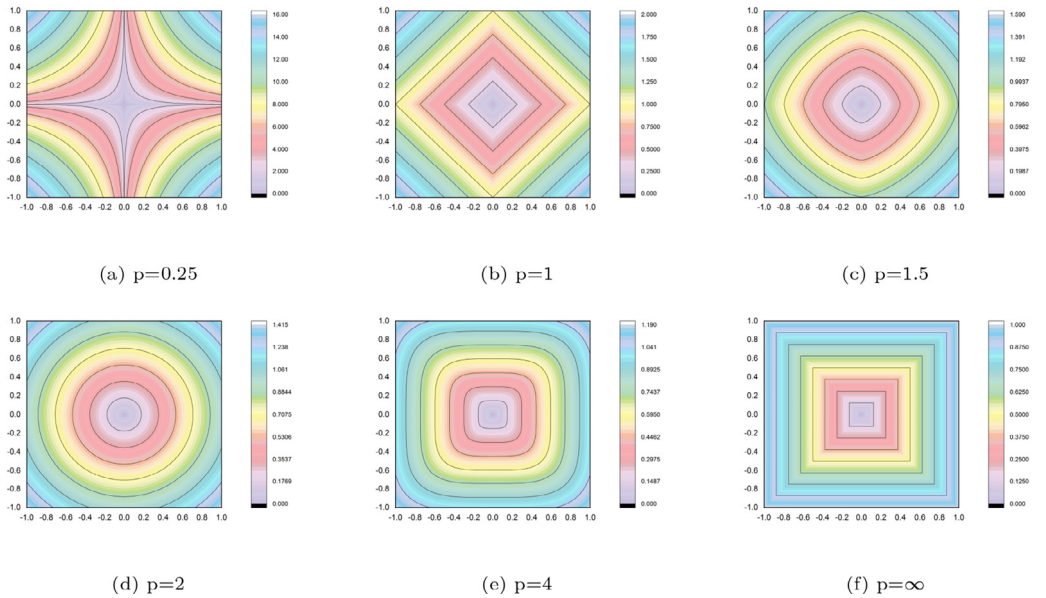

Fig. 1. Isosimilarity contour lines with eight different values of p in the Minkowski distance formula.

tances between instances and the related centers. The Minkowski K-means objective function can be written as below by applying Minkowski distance on (4).

$$\arg\min_{M',\delta}\sum_{i=1}^{N}\sum_{k=1}^{K}\delta_{ik}||\Phi_i - m'_k||^p \tag{20}$$

It should be considered that, it is not possible to use the definition of the center which previously used as Eq. (2) since it does not minimize (20) when δ is given constant. In the other words, For finding Minkowski's centers, we need to find value of vector m'_k for each cluster, minimizes the following function:

$$\arg\min_{m'_k}\sum_{\Phi_i\in C_k}|\Phi_i - m'_k|^p \tag{21}$$

Being minimizer of (21) is required for vector m'_k in order to prove the convergence of Minkowski K-means to a local optimum. In other words, it lowers the cost in each iteration monotonically. In this way, it is proven that the algorithm will be converged in a finite number of iterations since the algorithm iterates a function whose domain is a finite set and the cost is decreased monotonically.

The search space in order to find vector m'_k is too large for an exhaustive search because the algorithm needs accurate solutions. Note that finding vector m'_k is a single-objective problem, and elements on different dimensions are independent. Various evolutionary algorithms can be designed to find the best solutions. However, for $p > 1$ the Eq. (21) is a convex function and more desirable algorithms like steepest descent can be used to find the global minimizer [41].

4.2. Weighted version of explicit kernel K-means

Using the feature weighted version of Minkowski distance enables the algorithm to provide a better fit to the cluster structures than is possible with Minkowski K-means alone. It allows our algorithm to find both spherical and non-spherical (elliptical) structural clusters in feature space; accordingly, gives a much better fit to arbitrary shaped clusters in original spaces. Authors in [41] have extended weighted K-means variants work [42–44] through transforming feature weights to be feature scale. This means Minkowski exponent is assigned to feature weights too. The objective function for the weighted version of Minkowski K-means in Eq. (20) can be written as the following equation:

$$\arg\min_{M',\delta,W}\sum_{i=1}^{N}\sum_{k=1}^{K}\sum_{l=1}^{D'}\delta_{ik}(w_{kl})^p||\Phi_i^l - m_k'^l||^p \tag{22}$$

The weight w_{kl} reflects the relevance of feature l at the cluster k. They are assigned base on inverse proportion of dispersion of a feature within a certain cluster. Thus, a feature with small dispersion within a specific cluster would have a higher weight, and vice versa. More precisely, w_{kl} is updated on each iteration as Eq. (23), where $V_{kl} = \sum_{i\in c_k}|\varphi(x_{iv}) - m_{kv}|^p$

$$w_{kv} = \frac{1}{\sum_{u\in V}\left(\frac{V_{kl}}{V_{ku}}\right)^{\frac{1}{p-1}}} \tag{23}$$

4.3. Overview of Explicit KMWK-means

To illustrate how our algorithm works, we present a toy example using Tic-Tac-Toe Endgame dataset by applying Explicit KMWK-mean. The dataset contains the entire variety of potential board configurations at the end of tic-tac-toe games. Tic-Tac-Toe Endgame Data Set has 958 records with 9 attributes corresponding to one tic-tac-toe square. Each attribute can take three different values: "*x*" for player *X*, "*o*" for player *O* or "*b*" for blank. The goal is to see whether machine learners can correctly identify the end of Tic-Tac-Toe games. Therefore we cluster the data into two groups: positive and negative. In order to use K-means algorithm for a categorical dataset, we convert them to numeric features. The value "*o*" is converted into 1, The value "*x*" is converted into 2 and The value "*b*" is converted into 3. The results of the experiments reveal a superior classification accuracy in terms of their F-measures when group the dataset by using the chi-square kernel (Fig. 2a).

In the study of the Fourier series, periodic functions $k(x, y)$ can be written as the sum of simple waves mathematically represented by sines and cosines. However, function k is resolved into an infinite sum of sines and cosines; an approximation to whatever accuracy is desired can also be employed to obtain a finite low-dimensional discrete feature map. In Section 3, we have described a procedure to derive this approximation by using a single scalar signature function for homogeneous kernels. Consider the following example of two points $a = (1, 2)$ and $b = (2, 3)$. By employing Fourier series approach and using 5 features for approximating each dimension, the Squared Euclidean Distance between approximated feature maps of a and b for χ^2 distances takes the value $\Phi(a)' * \Phi(b) = 4.1675$ while the exact distance between a and b is $d_{\chi^2}(a, b) = 4.1667$. Fig. 2b shows that the homogeneous kernel map has the same classification accuracy as exact kernel one by using 5 features for approximating each dimension. Also, Fig. 2c shows a comparison between the exact and approx-

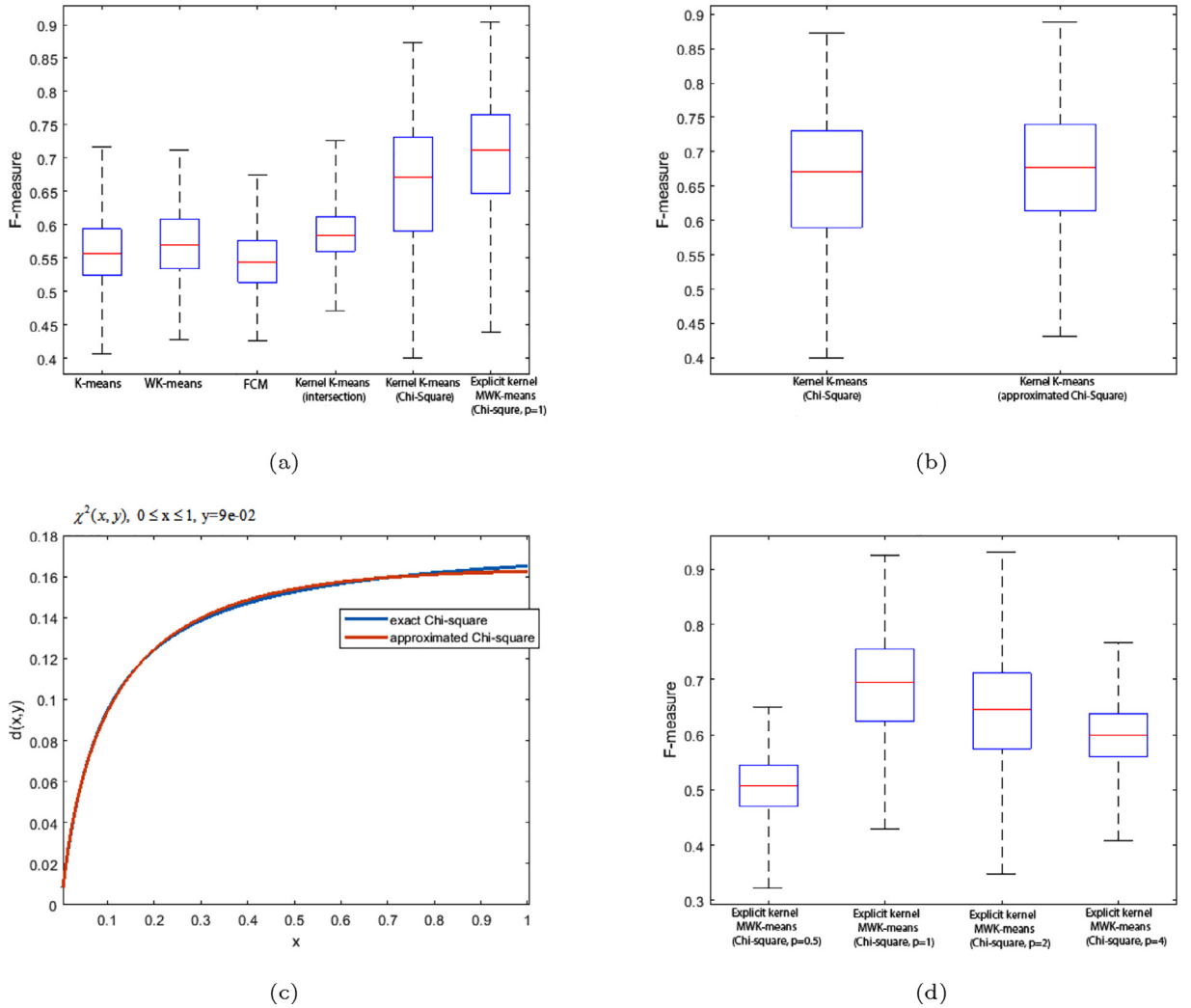

Fig. 2. (a) Various versions of K-means. (b) Exact and approximated Chi-square on Tic-Tac-Toe dataset. (c) Comparison between exact and approximated Chi-square when $0 \leqslant x \leqslant 1$ and $y = 9e-02$ (d) Semi-supervised learning of the exponent p.

imated χ^2 distances for ranges of value of $y \in [0, 1]$ By employing the Fourier series approach and using three features for approximating each dimension.

By accessing the data in the feature space explicitly, we can take advantage of more generality, and it is possible to adapt them to specific problems. Using the feature weighted version of Minkowski distance enables the algorithm to provide a better fit. The value of exponent p have been chosen by uncovering labels on 15% when it maximizes model accuracy (Fig. 2d). The results in Fig. 2a show that the Explicit kernel MWK-means algorithm outperforms the others both in terms of maximum and average values achieved.

4.4. Accelerate Minkowski K-means clustering

The Minkowski's center computation would significantly decelerate the running time. We already know that at $p = 2$ the center is the mean, and at $p = 1$ it is given by the component-wise median. Otherwise, it requires an iterative, steepest descent process or an evolutionary computation that can be considered in computation costs. A significant computational speed-up can be achieved by appropriate initialization in order to lessen the number of iterations and consequently decrease the time of searching for Minkowski centers as a process of minimization. We use an output vector containing cluster centers of K-means with p equal 2 or 1 as an initialization. This initialization approach results in an impressive reduction in the time required for clustering because all of these Minkowski's distances share the same properties, and data would be half-clustered. In [45], the effect of the initialization of K-Means has been studied. The authors found when the clusters overlap, considering various strategies for initialization would not matter much on the results.

4.5. Complexity

We estimate the time and space complexity of the suggested methods in this section. All the analyses and calculations are conducted as a unit cost for each operation and storage space. In the estimation, the terms that we used are outlined as follows: Here, n is the number of data objects, d is the number of dimensions of each object, d' is the number of dimensions for approximated feature map $\Phi(x)$; K is the number of clusters, and t is the number of iterations. For the Explicit Kernel MWK-means algorithm, the computation time consists of two parts: the computation time for feature map approximation and the computation time for Minkowski metric weighted K-Means on mapped data.

The parameters of the approximation for deriving feature map are selected automatically and do not require special regularization. Therefore, this calculation of the feature map requires $O(nd)$. The K-means algorithm is known to have a time complexity of $O(Ktn)$. Adding weight to the features will cause a rise in the scaling ratio and the number of iterations. However, the computational bottleneck in practice is the computation of Minkowski's centers. It involves an iterative steepest descent process that has a linear rate of convergence to find cluster centers in every iteration or an evolutionary computation which worsens the computation costs. Nonetheless, as described in the previous subsection, a significant computational speed-up can be achieved by appropriate initialization in order to lessen the number of iterations by using an output vector containing cluster centers of K-means with p equal 2 or 1 as an initialization.

The Explicit Kernel MWK-means algorithm reduces the overall memory consumption. While the memory required for kernel K-means algorithm is quadratic in the input size for the storage of the kernel matrix, the proposed algorithm have $O(nd')$ memory requirements which often $d' \ll n$.

5. Experimental results

In this section, We present the results of performance experiments of our Explicit Kernel MWK-means. We have conducted our experiments on four benchmark data sets: USPS dataset, MNIST dataset, caltech101, and MSRC-V1. Some relevant statistics of them are shown in Table 2.

Two standard metrics were used to measure the performance of the image clustering that is, Normalized Mutual Information (NMI) and Purity.

5.1. Data descriptions

MNIST Dataset introduced by Yann Lecun and Corinna Cortes. The dataset includes a total of 70,000 samples of digits 0–9. Each sample consists of 28 by 28 pixels which are within the range $[0, 255]$.

USPS dataset includes 11,000 0–9 digits instances. The dataset is known to be very complicated, with a recorded human error rate of 2.5%. The images are 16 by 16 gray-scale pixels.

Caltech101 dataset comprises 9144 images of items belonging to 101 classes and one background class. The number of images in each class differs. The size of each image is approximately 300×200 pixels. We have selected the commonly used 7 groups, i.e. Face, Motorbike, Dolla-Bill, Garfield, Snoopy, Stop-sign and Windsor-chair for our experiments.

MSRC-V1 dataset is from Microsoft Research in Cambridge. This data set is commonly used for scene recognition. We adopt Lee and Grauman's approach [46] to refine and to get 7 classes, including tree, building, airplane, cow, face, car and bicycle, where each class has 30 images.

5.2. Setting of the experiments

To figure out which value of the exponent p has the best performance, a semi-supervised manner was employed. The value of p was derived from uncovering the class of labels on a 15% data sample though clustering was conducted over the whole dataset. After running a series of clustering experiments at different values of p, the p that produced the higher clustering accuracy was picked. Each of the experiments is repeated 30 times, and the average NMI and purity are reported. The Welch's two-sample t-test is used to identify significantly altered clustering performances.

Images clustering by using raw pixels as features is unlikely to work effectively. The standard practice is to use visual descriptors such as HOG and SIFT. Both HOG and SIFT use histograms of pixel intensity gradients in their descriptors. The class of homogeneous kernels are popular due to its effectiveness. Hog can describe the shape and edge information of an

Table 2
Description of the data sets.

	# instances	# features	# classes
MNIST Digits	70,000	784	10
USPS Digits	11,000	256	10
Caltech7	441	60,000	7
MSRC-V1	210	68160	7

object, and SIFT features are invariant to image scale, rotation, noise and illumination changes. For MNIST and USPS databases, the HOG feature is used. We choose 4×4 grid cells, 2×2 cells in one block and 1 cell spacing as the parameter in HOG calculation. For the two other datasets, Caltech7 and MSRC-V1, the key points are extracted from each image then represented each keypoint as a 128-dimensional SIFT descriptor. A randomly chosen subset of SIFT features was clustered in order to form a visual vocabulary for either of the datasets. Each SIFT descriptor was then quantified into a visual word by considering the nearest cluster center. A 500-dimensional vector representation for each image is obtained.

5.3. Comparison results

Table 3 clearly shows the match between the value of p learned in semi-supervised and fully supervised settings. The only exception is on MNIST with Using χ^2 kernel that there is no match between learned and optimal value of exponent p. However, Table 4 shows it has still better performance than common kernel K-means with the exponent $p = 2$.

We compared the clustering performance of our method (Explicit kernel MWK-means) with their respective equally-weighted version and constant Euclidean one. In addition, we also compared the results of our method with the exact kernel K-means clustering. Table 4–7, demonstrate the clustering results in terms of NMI and purity on all data sets. It can be seen that, compared with common exact kernel K-means counterparts, our proposed Explicit kernel MWK-means improves the clustering performance on all the data sets. Welch's two sample-test was applied for statistical significance verification between proposed Explicit kernel MWK-means and other kernel K-means methods. When average clustering scores were compared between proposed Explicit kernel MWK-means and other kernel K-means methods, the t-test revealed a statistically significant difference for the difference in means.

The experiments on four real-world datasets are conducted to demonstrate the effectiveness of the Explicit kernel MWK-means method rather than the respective equally-weighted version, constant Euclidean one and the exact kernel K-means clustering. In subsequent, the following baseline methods are compared with our proposed Explicit kernel MWK-means method. We Choose the best clustering performance of Explicit kernel MWK-means, which uses one the χ^2, intersection or Jensen-Shannon kernel.

LBP, GIST, CENTRIST, DoG-SIFT, and HOG descriptors are extracted and the Gaussian Kernel by self-tuning [47] parameter σ applied for single view approaches. The four other methods exploit all of views (five similarity matrices).

- Feature Concatenation: Features of multiple views are concatenated, and then spectral clustering algorithm is conducted to the combined view representation.
- Kernel Addition: Building a similarity matrix for each view and combines information of multi-view data by averaging the sum of kernel matrices of all views, then inputting to a spectral clustering algorithm.
- Multi-View Graph Learning (MVGL) [48]: Learning the initial graph of each view, Then it learns the consensus proximity matrix with k connected components by combining the proximity matrices.
- Multi-view Spectral Clustering (MVSC) [49]: Learning a bipartite graph for each view, combining them using local manifold integration to fuse heterogeneous features.

As shown in Table 8, experimental results are reported in the form of the average NMI and Purity achieved by running 30, as well as the standard deviation. The proposed Explicit kernel MWK-means can achieve better clustering overall performance on four benchmark datasets than other competed single and multi-view clustering methods. The bold numbers highlight the best results.

Table 3
The effect of p on clustering accuracy results with the semi-supervised learning by revealing the class of labels on a 15% of data.

	Exponent p		NMI	
	learned	Optimal	learned	Optimal
MNIST (intersection)	1.83	1.87	0.7862	0.7981
MNIST (JS)	1.7	1.7	0.7692	0.7692
MNIST(χ^2)	1.7	3.1	0.7772	0.7862
USPS (intersection)	1.66	1.5	0.7212	0.7420
USPS (JS)	1.4	1.4	0.7353	0.7353
USPS (χ^2)	1.4	1.2	0.7291	0.7510
Caltech7 (intersection)	1.1	1	0.6558	0.6661
Caltech7 (JS)	1.1	0.9	0.6203	0.6367
Caltech7 (χ^2)	0.9	1.2	0.6547	0.6706
MSRC-V1 (intersection)	1.22	1	0.7228	0.7438
MSRC-V1 (JS)	1.2	1.1	0.7358	0.7559
MSRC-V1 (χ^2)	1.3	1.1	0.7565	0.7646

Table 4
MNIST Digits: Adaptive Explicit kernel MWK-means after using HOG descriptor. The best value for each measure for each of the clustering algorithms is shown in bold.

		intersection				chi2				JS			
		NMI		Purity		NMI		Purity		NMI		Purity	
	dm.	mean ± std	p-value	mean + std	p-value	mean + std	p-value	mean + std	p-value	mean + std	p-value	mean + std	p-value
Exact Kernel	–	0.7101 ± 0.024	⩽e−16	0.7388 ± 0.040	⩽1e−7	0.7109 ± 0.029	⩽1e−11	0.7047 ± 0.044	⩽1e−9	0.7121 ± 0.031	⩽1e−7	0.7007 ± 0.041	⩽1e−11
Explicit Kernel K-means	3	0.7138 ± 0.023	⩽1e−15	0.7388 ± 0.040	⩽1e−7	0.7186 ± 0.028	⩽1e−9	0.7107 ± 0.045	⩽1e−8	0.7148 ± 0.03	⩽1e−7	0.7027 ± 0.042	⩽1e−6
Explicit kernel MK-means	3	0.7362 ± 020	⩽1e−10	0.7380 ± 0.038	⩽1e−8	0.7192 ± 0.029	⩽1e−9	0.6868 ± 0.043	⩽1e−12	0.7272 ± 0.035	⩽1e−4	0.7127 ± 0.045	⩽1e−8
Explicit kernel MWK-means	3	**0.7862 ± 0.026**		**0.8048 ± 0.039**		**0.7772 ± 0.031**		**0.7909 ± 0.047**		**0.7692 ± 0.038**		**0.7935 ± 0.045**	

Table 5
USPS Digits: Adaptive Explicit kernel MWK-means after using HOG descriptor. The best value for each measure for each of the clustering algorithms is shown in bold.

		intersection				chi2				JS			
		NMI		Purity		NMI		Purity		NMI		Purity	
	dm.	mean ± std	p-value	mean + std	p-value	mean + std	p-value	mean + std	p-value	mean + std	p-value	mean + std	p-value
Exact Kernel	–	0.6567 ± 0.040	⩽1e−6	0.6981 ± 0.052	⩽1e−4	0.6695 ± 0.046	⩽1e−5	0.6932 ± 0.057	⩽1e−5	0.6506 ± 0.034	⩽1e−12	0.6976 ± 0.045	⩽1e−6
Explicit Kernel K-means	3	0.6591 ± 0.048	⩽1e−5	0.6986 ± 0.056	⩽1e−4	0.6631 ± 0.045	⩽1e−6	0.7006 ± 0.057	⩽1e−5	0.6621 ± 0.036	⩽1e−9	0.6923 ± 0.044	⩽1e−7
Explicit kernel MK-means	3	0.6896 ± 0.44	⩽1e−2	0.7087 ± 0.055	⩽1e−3	0.6953 ± 0.049	⩽1e−2	0.7107 ± 0.059	⩽1e−3	0.6991 ± 0.037	⩽1e−3	0.7029 ± 0.045	⩽1e−5
Explicit kernel MWK-means	3	**0.7212 ± 0.045**		**0.7578 ± 0.055**		**0.7291 ± 0.043**		**0.7658 ± 0.050**		**0.7353 ± 0.039**		**0.7643 ± 0.047**	

Table 6
Caltech7: Adaptive Explicit kernel MWK-means after using SIFT descriptor. The best value for each measure for each of the clustering algorithms is shown in bold.

		intersection				chi2				JS			
		NMI		Purity		NMI		Purity		NMI		Purity	
	dm.	mean ± std	p-value	mean + std	p-value	mean + std	p-value	mean + std	p-value	mean + std	p-value	mean + std	p-value
Exact Kernel	-	0.6064 ± 0.018	⩽1e−11	0.6581 ± 0.028	⩽1e−19	0.5918 ± 0.014	⩽1e−14	0.6849 ± 0.023	⩽1e−11	0.5649 ± 0.019	⩽1e−14	0.6710 ± 0.025	⩽1e−8
Explicit Kernel K-means	3	0.5961 ± 0.019	⩽1e−14	0.6941 ± 0.027	⩽1e−12	0.5981 ± 0.014	⩽1e−13	0.6858 ± 0.021	⩽1e−12	0.5919 ± 0.017	⩽1e−7	0.6735 ± 0.023	⩽1e−8
Explicit kernel MK-means	3	0.6158 ± 0.021	⩽1e−8	0.6987 ± 0.025	⩽1e−11	0.6147 ± 0.025	⩽1e−7	0.6926 ± 0.028	⩽1e−8	0.5803 ± 0.018	⩽1e−11	0.6635 ± 0.022	⩽1e−11
Explicit kernel MWK-means	3	**0.6558 ± 0.024**		**0.7581 ± 0.028**		**0.6547 ± 0.026**		**0.7419 ± 0.029**		**0.6203 ± 0.021**		**0.7159 ± 0.025**	

Table 7
MSRC-v1: Adaptive Explicit kernel MWK-means after using SIFT descriptor. The best value for each measure for each of the clustering algorithms is shown in bold.

		intersection				chi2				JS			
		NMI		Purity		NMI		Purity		NMI		Purity	
	dm.	mean ± std	p-value	mean + std	p-value	mean + std	p-value	mean + std	p-value	mean + std	p-value	mean + std	p-value
Exact Kernel		0.6509 ± 0.024	⩽1e−15	0.7201 ± 0.029	⩽1e−16	0.6889 ± 0.028	⩽1e−10	0.7301 ± 0.031	⩽1e−15	0.6859 ± 0.034	⩽1e−6	0.7407 ± 0.039	⩽1e−11
Explicit Kernel K-means	3	0.6504 ± 0.022	⩽1e−15	0.7221 ± 0.023	⩽1e−17	0.6927 ± 0.030	⩽1e−9	0.7343 ± 0.034	⩽1e−13	0.6893 ± 0.035	⩽1e−6	0.7436 ± 0.040	⩽1e−11
Explicit kernel MK-means	3	0.6728 ± 0.027	⩽1e−9	0.7256 ± 0.032	⩽1e−14	0.7058 ± 0.032	⩽1e−6	0.7390 ± 0.034	⩽1e−13	0.7065 ± 0.033	⩽1e−4	0.7541 ± 0.037	⩽1e−9
Explicit kernel MWK-means	3	**0.7228 ± 0.026**		**0.8105 ± 0.030**		**0.7556 ± 0.033**		**0.8269 ± 0.036**		**0.7358 ± 0.030**		**0.8254 ± 0.034**	

Table 8
Average clustering scores and standard deviation over 30 runs by different clustering methods. The best value for each measure for each of the clustering algorithms is shown in bold.

Metric	Methods	MNIST Digits	USPS Digits	Caltech7	MSRC-v1
NMI	LBP	0.5868 ± 0.021	0.4932 ± 0.008	0.4118 ± 0.009	0.4719 ± 0.007
	GIST	0.6214 ± 0.011	0.5511 ± 0.003	0.5236 ± 0.030	0.6390 ± 0.004
	CENTRIST	0.6011 ± 0.013	0.5285 ± 0.003	0.5550 ± 0.005	0.6027 ± 0.002
	DoG-SIFT	0.5511 ± 0.015	0.6120 ± 0.050	0.2794 ± 0.046	0.2904 ± 0.006
	HOG	0.6721 ± 0.005	0.6227 ± 0.005	0.5461 ± 0.014	0.5209 ± 0.003
	Feature Concat.	0.6973 ± 0.021	0.6481 ± 0.022	0.3810 ± 0.028	0.2968 ± 0.0012
	Kernel Addition	0.7456 ± 0.012	0.7085 ± 0.052	0.2973 ± 0.031	0.3019 ± 0.003
	MVGL	**0.8922 ± 0.000**	**0.7706 ± 0.000**	0.5588 ± 0.000	0.6580 ± 0.000
	MVSC	0.7808 ± 0.000	0.7409 ± 0.000	0.5810 ± 0.000	0.6696 ± 0.000
	Explicit kernel MWK-means	0.7862 ± 0.026	0.7353 ± 0.039	**0.6558 ± 0.024**	**0.7556 ± 0.033**
Purity	LBP	0.6845 ± 0.035	0.5632 ± 0.035	0.5737 ± 0.022	0.6106 ± 0.019
	GIST	0.6890 ± 0.015	0.6153 ± 0.017	0.7220 ± 0.041	0.7208 ± 0.022
	CENTRIST	0.7072 ± 0.045	0.5773 ± 0.025	0.6831 ± 0.017	0.7354 ± 0.017
	DoG-SIFT	0.6246 ± 0.030	0.6834 ± 0.011	0.7211 ± 0.022	0.4468 ± 0.020
	HOG	0.7268 ± 0.024	0.7243 ± 0.015	0.7008 ± 0.004	0.6392 ± 0.004
	Feature Concat.	0.7529 ± 0.016	0.7135 ± 0.025	0.6856 ± 0.018	0.5065 ± 0.015
	Kernel Addition	0.7872 ± 0.022	0.7470 ± 0.024	0.6221 ± 0.021	0.5218 ± 0.005
	MVGL	**0.8105 ± 0.000**	0.7302 ± 0.000	0.7797 ± 0.000	0.7205 ± 0.000
	MVSC	0.7935 ± 0.000	**0.7826 ± 0.000**	0.7335 ± 0.000	0.7958 ± 0.000
	Explicit kernel MWK-means	08048 ± 0.039	0.7643 ± 0.047	**0.7581 ± 0.028**	**0.8269 ± 0.036**

6. Conclusions

In this paper, we proposed a kernel K-means method based on explicit feature maps with further matching in the feature space. Using adaptive Minkowski metric and feature weighting in the feature space enables our algorithm to get high clustering quality and show strong robustness to noise feature and concentration phenomena. Our method reduces the memory requirements as it does not require storing a huge kernel matrix in memory. Experimental results demonstrate that our proposed method consistently achieves superiors clustering performances in terms of two standard metrics, Normalized Mutual Information (NMI) and Purity, evaluated on four real benchmark data sets.

CRediT authorship contribution statement

Amir Aradnia: Conceptualization, Software, Methodology, Validation, Writing - original draft. **Maryam Amir Haeri:** Conceptualization, Investigation, Validation, Formal analysis, Writing - review & editing, Supervision. **Mohammad Mehdi Ebadzadeh:** Writing - review & editing, Supervision.

Declaration of Competing Interest

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

References

[1] S. Daruru, N.M. Marin, M. Walker, J. Ghosh, Pervasive parallelism in data mining: dataflow solution to co-clustering large and sparse netflix data, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2009, pp. 1115–1124.
[2] D. Jiang, C. Tang, A. Zhang, Cluster analysis for gene expression data: a survey, IEEE Transactions on Knowledge & Data Engineering (11) (2004) 1370–1386.
[3] R. Dubes, A.K. Jain, Clustering methodologies in exploratory data analysis, in: Advances in Computers, vol. 19, Elsevier, 1980, pp. 113–228..
[4] J. MacQueen, et al., Some methods for classification and analysis of multivariate observations, in: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, vol. 1, Oakland, CA, USA, 1967, pp. 281–297..
[5] D. MacDonald, C. Fyfe, The kernel self-organising map, in: KES'2000. Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies. Proceedings (Cat. No. 00TH8516), vol. 1, IEEE, 2000, pp. 317–320..
[6] R. Inokuchi, S. Miyamoto, Lvq clustering and som using a kernel function, 2004 IEEE International Conference on Fuzzy Systems (IEEE Cat No. 04CH37542), vol. 3, IEEE, 2004, pp. 1497–1500.
[7] A.K. Qin, P.N. Suganthan, Kernel neural gas algorithms with application to cluster analysis, in: Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004, vol. 4, IEEE, 2004, pp. 617–620..
[8] F. Camastra, A. Verri, A novel kernel method for clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (5) (2005) 801–805.
[9] D.-Q. Zhang, S.-C. Chen, A novel kernelized fuzzy c-means algorithm with application in medical image segmentation, Artificial Intelligence in Medicine 32 (1) (2004) 37–50.
[10] D.Q. Zhang, S.C. Chen, Kernel-based fuzzy and possibilistic c-means clustering, in: Proceedings of the International Conference Artificial Neural Network, vol. 122, 2003, pp. 122–125..

[11] H. Zha, X. He, C. Ding, M. Gu, H.D. Simon, Spectral relaxation for k-means clustering, in: Advances in Neural Information Processing Systems, 2002, pp. 1057–1064..
[12] C. Ding, X. He, H.D. Simon, On the equivalence of nonnegative matrix factorization and spectral clustering, in: Proceedings of the 2005 SIAM International Conference on Data Mining, SIAM, 2005, pp. 606–610.
[13] I.S. Dhillon, Y. Guan, B. Kulis, A Unified View of Kernel k-means, Spectral Clustering and Graph Cuts, Citeseer, 2004.
[14] A. Vedaldi, A. Zisserman, Efficient additive kernels via explicit feature maps, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (3) (2012) 480–492.
[15] A. Hinneburg, C.C. Aggarwal, D.A. Keim, What is the nearest neighbor in high dimensional spaces?, in: 26th Internat. Conference on Very Large Databases, 2000, pp. 506–515..
[16] C.C. Aggarwal, A. Hinneburg, D.A. Keim, On the surprising behavior of distance metrics in high dimensional space, in: International Conference on Database Theory, Springer, 2001, pp. 420–434.
[17] J. Shawe-Taylor, N. Cristianini, Support vector machines, An Introduction to Support Vector Machines and Other Kernel-based Learning Methods (2000) 93–112.
[18] R. Zhang, A.I. Rudnicky, A large scale clustering scheme for kernel k-means, in: Object Recognition Supported by User Interaction for Service Robots, vol. 4, IEEE, 2002, pp. 289–292..
[19] N. Tsapanos, A. Tefas, N. Nikolaidis, I. Pitas, A distributed framework for trimmed kernel k-means clustering, Pattern Recognition 48 (8) (2015) 2685–2698.
[20] N. Tsapanos, A. Tefas, N. Nikolaidis, I. Pitas, Efficient mapreduce kernel k-means for big data clustering, in: Proceedings of the 9th Hellenic Conference on Artificial Intelligence, 2016, pp. 1–5.
[21] R. Chitta, R. Jin, T.C. Havens, A.K. Jain, Approximate kernel k-means: Solution to large scale kernel clustering, in: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2011, pp. 895–903.
[22] L. He, H. Zhang, Kernel k-means sampling for nyström approximation, IEEE Transactions on Image Processing 27 (5) (2018) 2108–2120.
[23] L. Chen, S. Zhou, J. Ma, M. Xu, Fast kernel k-means clustering using incomplete cholesky factorization, Applied Mathematics and Computation 402 (2021) 126037.
[24] J. Zhang, A. May, T. Dao, C. Ré, Low-precision random fourier features for memory-constrained kernel approximation, in: The 22nd International Conference on Artificial Intelligence and Statistics, PMLR, 2019, pp. 1264–1274.
[25] D. Marin, M. Tang, I.B. Ayed, Y. Boykov, Kernel clustering: density biases and solutions, IEEE Transactions on Pattern Analysis and Machine Intelligence 41 (1) (2017) 136–147.
[26] J. Zhao, H. Zhang, J.A. Zhang, Gaussian kernel adaptive filters with adaptive kernel bandwidth, Signal Processing 166 (2020) 107270.
[27] H.-C. Huang, Y.-Y. Chuang, C.-S. Chen, Multiple kernel fuzzy clustering, IEEE Transactions on Fuzzy Systems 20 (1) (2011) 120–134.
[28] B. Liu, T. Zhang, Y. Li, Z. Liu, Z. Zhang, Kernel probabilistic k-means clustering, Sensors 21 (5) (2021) 1892.
[29] Y. Lu, L. Wang, J. Lu, J. Yang, C. Shen, Multiple kernel clustering based on centered kernel alignment, Pattern Recognition 47 (11) (2014) 3656–3664.
[30] J. Liu, F. Cao, X.-Z. Gao, L. Yu, J. Liang, A cluster-weighted kernel k-means method for multi-view clustering, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 4860–4867..
[31] X. Liu, E. Zhu, J. Liu, T. Hospedales, Y. Wang, M. Wang, Simplemkkm: Simple multiple kernel k-means, arXiv preprint arXiv:2005.04975..
[32] C.M. Wilson, K. Li, X. Yu, P.-F. Kuan, X. Wang, Multiple-kernel learning for genomic data mining and prediction, BMC Bioinformatics 20 (1) (2019) 1–7.
[33] L. Guo, X. Zhang, Z. Liu, X. Xue, Q. Wang, S. Zheng, Robust subspace clustering based on automatic weighted multiple kernel learning, Information Sciences 573 (2021) 453–474.
[34] X. Zhang, Z. Ren, H. Sun, K. Bai, X. Feng, Z. Liu, Multiple kernel low-rank representation-based robust multi-view subspace clustering, Information Sciences 551 (2021) 324–340.
[35] J.-W. Xu, P.P. Pokharel, K.-H. Jeong, J.C. Principe, An explicit construction of a reproducing gaussian kernel hilbert space, in: 2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings, vol. 5, IEEE, 2006, pp. V-V..
[36] R. Wang, Y. Xu, Functional reproducing kernel hilbert spaces for non-point-evaluation functional data, Applied and Computational Harmonic Analysis 46 (3) (2019) 569–623.
[37] M. Hein, O. Bousquet, Hilbertian metrics and positive definite kernels on probability measures..
[38] D. Francois, V. Wertz, M. Verleysen, The concentration of fractional distances, IEEE Transactions on Knowledge and Data Engineering 19 (7) (2007) 873–886.
[39] F. Salehi, M.R. Keyvanpour, A. Sharifi, Smkfc-er: Semi-supervised multiple kernel fuzzy clustering based on entropy and relative entropy, Information Sciences 547 (2021) 667–688.
[40] A. Flexer, D. Schnitzer, Choosing lp norms in high-dimensional spaces based on hub analysis, Neurocomputing 169 (2015) 281–287.
[41] R.C. De Amorim, B. Mirkin, Minkowski metric, feature weighting and anomalous cluster initializing in k-means clustering, Pattern Recognition 45 (2012) 1061–1075.
[42] E.Y. Chan, W.K. Ching, M.K. Ng, J.Z. Huang, An optimization algorithm for clustering using weighted dissimilarity measures, Pattern Recognition 37 (5) (2004) 943–952.
[43] J.Z. Huang, M.K. Ng, H. Rong, Z. Li, Automated variable weighting in k-means type clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (5) (2005) 657–668.
[44] J.Z. Huang, J. Xu, M. Ng, Y. Ye, Weighting method for feature selection in k-means, Computational Methods of Feature Selection (2008) 193–209.
[45] P. Fränti, S. Sieranoja, How much can k-means be improved by using better initialization and repeats?, Pattern Recognition 93 (2019) 95–112
[46] Y.J. Lee, K. Grauman, Foreground focus: Unsupervised learning from partially matching images, International Journal of Computer Vision 85 (2) (2009) 143–166.
[47] L. Zelnik, Manor and pietro perona. self* tuning spectral cluste* ring, in: Neural Information Processing Systems, 2004..
[48] Y. Wang, X. Li, R. Ruiz, S. Sui, An iterated greedy heuristic for mixed no-wait flowshop problems, IEEE Transactions on Cybernetics 48 (5) (2017) 1553–1566.
[49] Y. Li, F. Nie, H. Huang, J. Huang, Large-scale multi-view spectral clustering via bipartite graph, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 29, 2015..