Kernel Functions Derived from Fuzzy Clustering and Their Applications

March 2011

Jeongsik Hwang

Kernel Functions Derived from Fuzzy Clustering and Their Applications

Graduate School of Systems and Information Engineering

University of Tsukuba

March 2011

Jeongsik Hwang

Acknowledgments

It would not have been possible to write this thesis without the help and support of the kind people around me, to only some of whom it is possible to give particular mention here.

First and foremost I offer my sincerest gratitude to my supervisor, Prof. Sadaaki Miyamoto who has showed me the appropriate attitude as a researcher and provided me with abundant opportunities. My graduate student life would be totally different without his guidance and encouragement. I cannot even imagine what it would be like without him, Prof. Sadaaki Miyamoto .

I also would like to thank members of the soft computing group in university of Tsukuba. Their biggest contribution is to remind me of the importance of cooperation between members.

My parents and brother have given me their unequivocal support throughout, as always, for which my mere expression of thanks likewise does not suffice. I am indebted to my

father for his care and love. I have no suitable word that can fully describe my mother everlasting love to me. In particular, I would like to thank my wife, Junga Lee. Without my wife's encouragement, I would not have finished the degree.

Finally, I am grateful to my thesis committee members, Dr. Yasunori Endo, Dr. Mika Sato, Prof. Takehisa Onisawa and Dr. Hajime Nobuhara.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

As computational means have more and more expanded, the automation of data collection has generated tremendous amount of datasets. Since such datasets are massive and complex, it is difficult for us to find solutions to some problems. Hence, data analysis to obtain knowledge out of datasets is required. Machine learning can often be successfully applied to satisfy these requirements.

Usually, learning means acquiring knowledge about a previously unknown system or little known concept. All objects in a dataset used by machine learning algorithms are represented using the same set of features. The features may be continuous, categorical or discrete. If datasets are given with known labels corresponding correct outputs, then the learning algorithm is called supervised, where element $z = (x, y)$ of a dataset is a pair of

an input $x \in \mathcal{X}$ and an output $y \in \mathcal{Y}$, where $\mathcal{X}$ is referred to as the input space, and $\mathcal{Y}$ is referred to as the output space of class labels. In contrast to supervised learning, objects are unlabeled in unsupervised learning. That is, we do not have output $\mathcal{Y}$.

It is important to understand the difference between supervised and unsupervised classification. In supervised classification, we are firstly provided with a collection of labeled patterns of the descriptions of classes. And then, the problem is to label a newly unlabeled pattern. In the case of unsupervised classification, the problem is to group unlabeled objects into meaningful clusters based on similarity between objects. Clustering can be considered the most important unsupervised classification problem.

The method of hard *c*-means (HCM) [4, 28] is the most popular clustering algorithm. The hard *c*-means are to divide a set of objects into *c* clusters using a simple procedure. An object *x* is assigned to a cluster exclusively. The assigned cluster has the nearest cluster center to an object *x*. There are many extensions of hard *c*-means.

The method of fuzzy *c*-means (FCM) [4, 23, 24, 25] has been remarked by many researchers. It is an extension of hard *c*-means using the concept of fuzzy sets theory. That is, fuzzy *c*-means is a method of clustering which allows an object to belong to two or more different clusters.

The methods of entropy-based fuzzy *c*-means (eFCM) have been proposed in order to regularize the objective function of hard *c*-means [43, 45, 46, 50]. An idea of regularization has been found in the formulation of ill-posed problem [5].

In possibilistic clustering [39, 41], the constraint $\sum u_{ki} = 1$ of fuzzy *c*-means is not

imposed on $U = (u_{ki})$, the membership of object x_k to cluster i. That is, no practical constraint for U. Thus, we know that possibilistic clustering is more robust to noise and outliers. FCM has been applied to many real-world problems, but there are not many theoretical studies on FCM. A typical study of the latter aspect has been done by Miyamoto and Mukaidono [46]; they introduced fuzzy classification functions and showed their importance in observing properties of solutions of FCM and related methods. We should further investigate theoretical aspects of FCM, eFCM, and related methods.

Vector quantization (VQ) is a well-known prototype based clustering method. A p-dimensional vector is mapped into one of a representative vector [2, 13, 17, 57]. Vector quantization and self-organizing maps (SOM) [53] are, strictly, not clustering algorithms in themselves, but application of them to clustering is similar to the hard c-means algorithm and straightforward. Thus, we will call it VQ clustering.

In real-world situations, methods of clustering to deal with datasets having nonlinear classification boundaries are more appropriate. It has been known that conventional clustering methods cannot separate two clusters shown in Fig. 1.1 and Fig. 1.2. For this reason a kernel method to classify nonlinear datasets is now studied by many researchers in recent years.

In general, objects can be considered as points within a p-dimensional space where each dimension corresponds to one of the features which are used to describe an object. In kernel methods, the absolute position of the features within feature space is not significant. The basic idea of kernel methods is as follows. Objects of a dataset in input space $\mathcal{X}$ are mapped

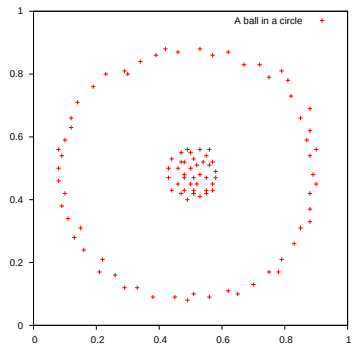

Figure 1.1: A ball in a circle.

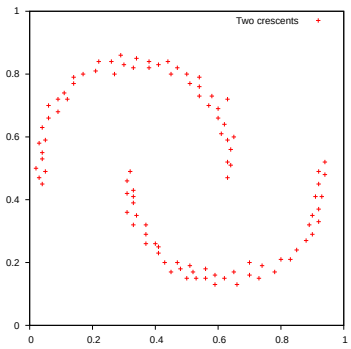

Figure 1.2: Two crescents.

by a mapping function into the feature space, which is of possibly infinite dimension. Then, a linear separation in the transformed feature space corresponds to a nonlinear separation in the original input space. We thus are capable of analyzing objects in the feature space $\mathcal{H}$ instead of the input data space $\mathcal{X}$. In this thesis we assume that $\mathcal{H}$ is a *reproducing kernel Hilbert space* (RKHS) with kernel $K = \langle f, f \rangle$, where $\langle \cdot, \cdot \rangle$ is the inner product in $\mathcal{H}$, and note that kernels can be defined on more general input spaces $\mathcal{X}$ [29, 33, 34].

Kernel methods are a class of algorithms for data analysis. Because of their simplicity, kernel function and related RKHS play an increasingly significant role in data analysis and their applications. This was initiated with the introduction of support vector machine (SVM) [10, 11, 35, 55], and continued with the development of many other kernel-based clustering algorithms [8, 12, 29, 48, 49].

An SVM is an algorithm that makes crucial use of a kernel function for classification. Such an SVM performs better than other classification algorithms for datasets having non-linear cluster boundaries. As long as the kernel function is legitimate, an SVM will operate

correctly even if the practitioner does not know exactly what features of the training data are being used in the transformed feature space. However, it is not an easy problem to define a kernel function. Different application problems will have suitable kernel functions, and moreover theoretical properties of kernel functions have to be studied.

1.2 Purpose of Study

Various types of kernel functions have been studied so far by many researchers [12, 29, 32, 33, 44]. The Gaussian kernel, in particular, has been found to be useful and applied to many problems. A kernel having similar characteristic to the Gaussian kernel is yet unknown. On this point, more studies have to be done in view of the importance of the Gaussian kernel, since to compare similar kernels, if any, will be more useful when they are important in applications.

Kernel functions proposed so far are based on machine learning models and statistical models. It has not yet been shown that fuzzy models have close relation to kernel functions. As noted earlier, we should investigate theoretical aspects of fuzzy models of clustering, and study whether there is a relation between kernel functions and solutions of fuzzy *c*-means clustering.

We thus address three questions: does a fuzzy model of clustering have close relation to kernel functions? Can we derive a new kernel function from a fuzzy model? Assume that a new kernel function is derivable, is the derived kernel function similar to, and as useful

as the Gaussian kernel? A related observation is as follows. We note that the Gaussian kernel is based on the Gaussian error function e^{-t^2} which is used as the most standard probability density function. By analogy, we consider another function $\frac{1}{1+t^2}$. These two functions are typical symmetric functions for probability distributions on the line of real-values. Incidentally, the Gaussian error function is known to be a positive-definite kernel. It thus is natural for us to have a question whether or not the other types of functions are also positive-definite kernel. It will be shown that the second type of functions will provide us a new class of kernel functions. Another significant property is that the second type of functions are closely related to fuzzy *c*-means and possibilistic clustering, as we will see in this thesis.

1.3 Outline of this Thesis

This thesis is organized as follows.

Chapter 1 has presented a brief introduction of kernel method.

Chapter 2 reviews basic methods of *c*-means clustering. That is, hard *c*-means, fuzzy *c*-means, entropy-based fuzzy *c*-means and vector quantization clustering are introduced.

And then, Chapter 3 describes kernel methods and kernel-based clustering.

In Chapter 4, we propose three basic functions derived from fuzzy clustering and establish necessary and sufficient conditions for the basic functions to be positive-definite kernel functions, using the completely monotone functions by Schönberg.

In Chapter 5, we discuss numerical results in which the three kernel functions are applied to kernel fuzzy *c*-means and kernel vector quantization clustering.

Finally, the conclusion of this thesis and problems to be discussed in future studies is given in Chapter 6.

Chapter 2

Basic Methods of c-Means Clustering

In this section we describe several basic methods of c-means clustering.

Let the objects to be clustered be $x_k = (x_k^1, \ldots, x_k^p)^T \in \mathbb{R}^p$, $k = 1, \ldots, N$, a vector in the p-dimensional Euclidean space. Cluster centers are $v_i = (v_i^1, \ldots, v_i^p)^T$, $i = 1, \ldots, c$, where c is the number of clusters. The matrix $U = (u_{ki})$, $(k = 1, \ldots, N,\ i = 1, \ldots, c)$ is used as usual, where u_{ki} means the degree of belongingness of an object x_k to cluster i.

2.1 Hard c-Means

K-means is one of the simplest unsupervised classification algorithms. K-means is also referred to as the hard c-means (HCM). The procedure follows a simple and easy way to classify a given dataset through a certain number of clusters (we assume K clusters. hence the term of K-means) which is fixed a priori. The main idea is to determine K-centers, one

for each cluster. The next step is to take each object out of a given dataset and we allocate it to the cluster of the nearest center. At this point, we adopt new centers as the result of clustering. After all, the K-centers change their location step by step until there are no more changes. In other words, cluster centers do not move any more. This algorithm aims at minimizing an objective function. The objective function of the hard c-means is

$$J_{hcm}(U, V) = \sum_{k=1}^{N} \sum_{i=1}^{c} u_{ki} D(x_k, v_i), \tag{2.1}$$

where $D(x_k, v_i) = \|x_k - v_i\|^2$. In the HCM algorithm, u_{ki} is determined as follows:

$$u_{ki} = \begin{cases} 1, & i = \arg\min\limits_{1 \le i \le c} \|x_k - v_i\|^2. \\ 0, & \text{otherwise.} \end{cases} \tag{2.2}$$

The objective function J_{hcm} is minimized with respect to U.

Let us now fix u_{ki}. We obtain the updating equation of cluster center v_i:

$$\frac{\partial J_{hcm}}{\partial v_i} = 0,$$

thus, v_i is formed as

$$v_i = \frac{\sum\limits_{k=1}^{N} u_{ki} x_k}{\sum\limits_{k=1}^{N} u_{ki}}. \tag{2.3}$$

The algorithm is composed of the following steps:

HCM (Hard c-Means) Algorithm

HCM1. Set c initial value for cluster center v_i (e.g., select c objects randomly as $v_i,\ i = 1, \ldots, c$).

HCM2. Find optimal solution of J_{hcm} with respect to U while $\bar{V}$ is fixed: put

$$\bar{U} = \arg\min_{U} J_{hcm}(U, \bar{V}).$$

HCM3. Find optimal solution of J_{hcm} with respect to V while $\bar{U}$ is fixed: put

$$\bar{V} = \arg\min_{V} J_{hcm}(\bar{U}, V).$$

HCM4. If the solution $(\bar{U}, \bar{V})$ is convergent, stop; else go to **HCM2**.

End HCM.

We have two ways to judge whether or not clusters are convergent in **HCM 4**. That is, clusters are judged to be convergent when the new $\bar{U}$ coincides with the last U, or $\bar{V}$ does not change their position.

2.2 Standard Fuzzy c-Means

There are many extensions of hard c-means. The method of fuzzy c-means (FCM) has been remarked by many researchers [19, 23, 24, 25]. It is an extension of hard c-means using

the concept of fuzzy sets theory. That is, fuzzy *c*-means is a method of clustering which allows an object to belong to more than one cluster.

Standard fuzzy *c*-means (sFCM) is a very simple method of the related fuzzy clustering algorithms. Note that this algorithm proposed by Dunn and Bezdek is called the standard method of fuzzy *c*-means, to distinguish it from entropy-based fuzzy *c*-means [45]. The well-known fuzzy clustering algorithm is derived by minimizing an objective function of the following form

$$J_{sfcm}(U,V) = \sum_{k=1}^{N}\sum_{i=1}^{c}(u_{ki})^m D(x_k, v_i) \tag{2.4}$$

with respect to V and U subject to the constraint

$$M_{fcm} = \{\, U = (u_{ki}) \;:\; \sum_{i=1}^{c} u_{ki} = 1,\ \forall k\ ;\ 0 < \sum_{k=1}^{N} u_{ki} < n,\ \forall i \,\}. \tag{2.5}$$

Through (2.5), the belongingness of an object x_k in the cluster i is related to those of x_k to the remaining clusters. Different values of m of the objective function $J_{sfcm}(U,V)$ in (2.4) determine the degree of fuzziness. In particular, if $m = 1$, we have hard *c*-means clustering.

We first assume that no x_k coincides with the cluster centers. Let us consider the optimal solution of the objective function $J_{sfcm}(U,V)$ with respect to U subject to the constraint (2.5). It leads to the following Lagrangian function

$$L = J_{sfcm} + \sum_{k=1}^{N} v_k(\sum_{i=1}^{c} u_{ki} - 1), \tag{2.6}$$

where v_k is the Lagrange multipliers. Solving L with respect to u_{ki}, we have

$$\frac{\partial L}{\partial u_{ki}} = m(u_{ki})^{m-1} D(x_k, v_i) + v_k. \tag{2.7}$$

Let the partial derivative of L be equal to 0 and solving for u_{ki}, we obtain

$$u_{ki} = \left(\frac{-\nu_k}{mD(x_k, v_i)} \right)^{\frac{1}{m-1}}. \tag{2.8}$$

Substituting u_{ki} of the previous equation into the constraint equation $\sum_{i=1}^{c} u_{ki} = 1$, we obtain

$$\sum_{j=1}^{c} \left(\frac{-\nu_k}{mD(x_k, v_j)} \right)^{\frac{1}{m-1}} = 1. \tag{2.9}$$

Combining (2.9) with (2.8) in order to eliminate ν_k, we obtain

$$u_{ki} = \frac{D(x_k, v_i)^{-\frac{1}{m-1}}}{\sum_{j=1}^{c} D(x_k, v_j)^{-\frac{1}{m-1}}}. \tag{2.10}$$

On the other hand, there is no constraint for v_i, and we have

$$v_i = \frac{\sum_{k=1}^{N} (u_{ki})^m x_k}{\sum_{k=1}^{N} (u_{ki})^m}. \tag{2.11}$$

2.3 Entropy-Based Fuzzy c-Means

The method of entropy-based fuzzy c-means has been proposed in order to regularize the objective function of hard c-means [43, 45, 46, 50]. The regularization has been found in the formulation of ill-posed problems [5]. We consider that a typical regularization is done by adding a regularizing function. In the present context, we consider

$$J^{\lambda}(U,V) = J(U,V) + \lambda^{-1}K, \tag{2.12}$$

where K is a nonlinear regularizing function as follows:

$$K = \sum_{k=1}^{N} u_{ki} \sum_{i=1}^{c} \log u_{ki}, \tag{2.13}$$

and λ is a regularizing parameter.

The following objective function is used for the entropy-based methods.

$$J_{efcm}(U,V) = \sum_{k=1}^{N}\sum_{i=1}^{c} u_{ki}D(x_k,v_i) + \lambda^{-1}\sum_{k=1}^{N} u_{ki}\sum_{i=1}^{c}\log u_{ki}, \tag{2.14}$$

which leads to the following Lagrangian function:

$$L = J_{efcm} + \sum_{k=1}^{N} v_k(\sum_{i=1}^{c} u_{ki} - 1), \tag{2.15}$$

where v_k is the Lagrange multipliers. Solving L with respect to u_{ki}, we have

$$\frac{\partial L}{\partial u_{ki}} = D(x_k,v_i) + \lambda^{-1}(1+\log u_{ki}) + v_k. \tag{2.16}$$

Let the partial derivative of L be equal to 0 and solving for u_{ki}, we obtain

$$u_{ki} = \exp(-\lambda D(x_k,v_i) - \lambda v_k - 1). \tag{2.17}$$

Substituting u_{ki} of the previous equation into the constraint equation $\sum_{i=1}^{c} u_{ki} = 1$, we obtain

$$\sum_{j=1}^{c}\Big(\exp(-\lambda D(x_k,v_j) - \lambda v_k - 1)\Big) = 1. \tag{2.18}$$

Combining (2.18) with (2.17) in order to eliminate v_k, we obtain

$$u_{ki} = \frac{\exp(-\lambda D(x_k, v_i))}{\sum_{j=1}^{c} \exp(-\lambda D(x_k, v_j))}. \tag{2.19}$$

On the other hand, there is no constraint for v_i, and we have

$$v_i = \frac{\sum_{k=1}^{N} u_{ki} x_k}{\sum_{k=1}^{N} u_{ki}}. \tag{2.20}$$

2.4 Possibilistic Clustering

In possibilistic clustering, the constraint $\sum_{i=1}^{c} u_{ki} = 1$ of fuzzy c-means is not imposed on $U = (u_{ki})$, the membership of an object x_k to the cluster i, that is, no practical constraint for U:

$$M_{pos} = \{U = (u_{ki}) : \sum_{k=1}^{N} u_{ki} > 0, \forall i; u_{kj} > 0, \forall k, j\}. \tag{2.21}$$

Possibilistic clustering is more robust to noise and outliers [39, 40, 41]. In fuzzy c-means, u_{ki} denotes the grade of membership of x_k in the cluster i. Here, u_{ki} may be interpreted as the degree of compatibility that x_k with the cluster i or the possibility of belongingness of x_k to the cluster i. Note that the possibility depends exclusively on x_k and cluster i. That is, it is independent of the possibilities of belongingness of x_k to any other clusters.

Let us remind here that the objective function to be minimized is:

$$J(U, V) = \sum_{k=1}^{N} \sum_{i=1}^{c} (u_{ki})^m D(x_k, v_i). \tag{2.22}$$

This objective function is equal to that of fuzzy c-means, except that the constraint is changed to (2.21). However, direct minimization with respect to u_{ki} will produce the trivial zero solution.

Avoiding the trivial solution, we add a term in (2.22):

$$K = \sum_{i=1}^{c} \eta_i \sum_{k=1}^{N} (1 - u_{ki})^m. \tag{2.23}$$

Then, the objective function becomes

$$J_{pos}(U, V) = \sum_{k=1}^{N} \sum_{i=1}^{c} (u_{ki})^m D(x_k, v_i) + \sum_{i=1}^{c} \eta_i \sum_{k=1}^{N} (1 - u_{ki})^m. \tag{2.24}$$

Differentiating (2.24) with respect to u_{ki}, and setting it to 0 leads to the equation

$$\frac{\partial J_{pos}}{\partial u_{ki}} = m(u_{ki})^{m-1} D(x_k, v_i) - m\eta_i (1 - u_{ki})^{m-1} = 0. \tag{2.25}$$

We finally obtain

$$u_{ki} = \frac{1}{1 + \left(\frac{D(x_k, v_i)}{\eta_i}\right)^{\frac{1}{m-1}}}. \tag{2.26}$$

On the other hand, updating of v_i is achieved by (2.27)

$$v_i = \frac{\sum_{k=1}^{N} (u_{ki})^m x_k}{\sum_{k=1}^{N} (u_{ki})^m}. \tag{2.27}$$

2.5 Vector Quantization Clustering

Vector quantization (VQ) is also a well-known prototype-based clustering method. A p-dimensional vector is mapped into one of representative vectors [2, 3, 17, 57].

The goal of vector quantization resides in determining a mapping $\mathcal{M}\colon \mathbb{R}^p \to V$ from the p-dimensional Euclidean space $\mathbb{R}^p$ to a finite subset $V = \{v_1, \ldots, v_c\}$. Each v_i is called a codevector and the set V is called a codebook. The vector quantizer is chosen to minimize the average distortion measure $D(\mathbb{R}^p, V) = E[d(x, \mathcal{M}(x))]$ resulting from the approximation of the input $x \in \mathbb{R}^p$ by the codevector $\mathcal{M}(x)$, and is updated according to

$$v_l(t+1) = v_l(t) + \alpha[x(t) - v_l(t)], \tag{2.28}$$

where $d(\cdot)$ is usually taken as the standard squared Euclidean distance $d(x, y) = \|x - y\|^2$, and $E(\cdot)$ denotes the expectation operator, and t indexes the updating step [2, 17].

A learning rate α which starts from an initial value is reduced monotonically to zero. The mapping of a vector quantizer $\mathcal{M}$ determines a partition of the input space $\mathbb{R}^p$ into c disjoint regions $G_1, \ldots, G_c$, such that $G_i = \{x \in \mathbb{R}^p \colon \mathcal{M}(x) = v_i\}$.

Furthermore, a learning algorithm for SOM can also be used to design vector quantizers. A significant difference of the VQ from the SOM is that the update is restricted to the winner of codebook vectors. VQ and SOM are not clustering algorithms in themselves, but application of them to clustering is similar to the hard c-means algorithm. We thus will call it VQ clustering.

The algorithm of VQ clustering [45] is as follows:

VQc (clustering by VQ) Algorithm

VQc1. Set initial value for codevector v_i (e.g., select c objects randomly as $v_i, i = 1, \ldots, c$).

Set $t = 0$ and repeat from VQc2 to VQc4 until convergence (or until the maximum number of iterations is attained).

VQc2. Select $x(t) \in X$.

VQc3. Let $v_l(t) = \arg\min_{1 \le i \le c} \|x(t) - v_i(t)\|$.

VQc4. Update $v_1(t), \ldots, v_c(t)$:

$$v_l(t+1) = v_l(t) + \alpha[x(t) - v_l(t)],$$

$$v_i(t+1) = v_i(t), \ i \neq l.$$

Object represented by $x(t)$ is allocated to G_l.

$t = t + 1$.

End VQc.

In this algorithm, the parameter $\alpha(t)$ should be taken to satisfy the following:

$$\sum_{t=1}^{\infty} \alpha(t) = \infty, \sum_{t=1}^{\infty} \alpha^2(t) < \infty, t = 1, 2, \ldots .$$

For example, $\alpha(t) = constant/t$ satisfies these conditions. Note that learning rate is a

monotonical decreasing function which approaches to zero.

Chapter 3

Kernel-Based Clustering

Kernel methods are a class of algorithms for data analysis, whose well-known application is the support vector machine (SVM). Such an SVM performs better than other classification algorithms for datasets having nonlinear cluster boundaries [10, 11, 35, 55]. The other algorithms with kernels include linear discriminant analysis, principle components analysis, spectral clustering and so on [6, 7, 20, 22, 54, 56].

3.1 Kernels

In this section, we describe how kernels defined as an inner product in feature space. Kernel K is corresponding to an inner product in a feature space $\mathcal{H}$ via a map Φ,

$$\Phi : \mathcal{X} \rightarrow \mathcal{H}$$

and in order to compute inner products of the form $\langle \Phi(x), \Phi(y) \rangle$, we employ kernel representations of the following form:

$$k(x, y) = \langle \Phi(x), \Phi(y) \rangle,$$

which allow us to compute the value of the inner product in $\mathcal{H}$ without having to explicitly compute the map Φ. Note that kernels are usually complex-valued in the mathematical literature. However we only consider the real-valued kernels, which is the common practice in machine learning.

3.1.1 Positive-Definite Kernels

We begin with some basic definitions.

Definition 1. *(Gram Matrix [8])*

Given a function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ and $x_1, \ldots, x_m \in \mathcal{X}$, the $m \times m$ matrix M with elements

$$M_{ij} = K(x_i, x_j)$$

is called the Gram matrix of M with respect to $x_1, \ldots, x_m$:

$$M = \begin{pmatrix} K(x_1, x_1) & K(x_2, x_1) & \cdots & K(x_m, x_1) \\ K(x_1, x_2) & K(x_2, x_2) & \cdots & K(x_m, x_2) \\ \vdots & \vdots & \ddots & \vdots \\ K(x_1, x_m) & K(x_2, x_m) & \cdots & K(x_m, x_m) \end{pmatrix}.$$

Definition 2. *(Positive-Definite Matrix [8])*

A real $m \times m$ symmetric matrix M satisfying

$$\sum_{i,j=1}^{m} c_i c_j M_{ij} \geq 0,$$

for all $c_1, \ldots, c_m \in \mathbb{R}$ is called a positive-definite matrix.

Definition 3. *(Positive-Definite Kernel [8])*

Let $\mathcal{X}$ be a nonempty set. A function K on $\mathcal{X} \times \mathcal{X}$ with for all $m \in \mathbb{N}$ and all $x_1, \ldots, x_m \in \mathcal{X}$ gives rise to a positive-definite Gram matrix is called a positive-definite kernel.

Occasionally, positive-definite kernel is simply called a kernel. For convenience throughout this thesis, we will use the term positive-definite for kernel functions that simply comply with non-negativity. We do not distinguish semi-positive definite and positive-definite kernels, as in most literature, e.g., [8, 44].

3.1.2 Reproducing Kernel Hilbert Space

Assume that K is a real-valued positive-definite kernel, and $\mathcal{X}$ a nonempty set. We define a map from $\mathcal{X}$ into the space of functions mapping $\mathcal{X}$ into $\mathcal{H}$, denoted as follows

$$\Phi : \mathcal{X} \to \mathcal{H}$$

$$x \to k(\cdot, x),$$

where

$$\mathcal{H} = \{ f : \mathcal{X} \to \mathbb{R} \}.$$

Here, $\Phi(x)$ denotes the function that assigns the value $K(x, y)$ to $x, y \in \mathcal{X}$, i.e., $\Phi(x)(\cdot) = K(\cdot, x)$. In other words, An object x is represented by its similarity to all other objects in the input data space $\mathcal{X}$. Let us consider two linear combinations of the following form:

$$f(\cdot) = \sum_{i=1}^{m} \alpha_i K(\cdot, x_i), \tag{3.1}$$

$$g(\cdot) = \sum_{j=1}^{n} \beta_j K(\cdot, y_j), \tag{3.2}$$

where $m, n \in \mathbb{N}$, $\alpha_i, \beta_j \in \mathbb{R}$ and $x_i, y_j \in \mathbb{R}$. Next, we define an inner product between f and g:

$$\langle f, g \rangle = \sum_{i=1}^{m} \sum_{j=1}^{n} \alpha_i \beta_j K(x_i, y_j). \tag{3.3}$$

Note that this expression explicitly contains the expansion coefficients, which need not be unique. Thus we obtain two equations as follows:

$$\langle f, g \rangle = \sum_{j=1}^{n} \beta_j f(y_j), \quad \langle f, g \rangle = \sum_{i=1}^{m} \alpha_i g(x_i). \tag{3.4}$$

Equation (3.3) is symmetric:

$$\langle f, g \rangle = \langle g, f \rangle. \tag{3.5}$$

Moreover, it is positive-definite, since the kernel function K implies positive-definiteness. We thus have the following equation:

$$\langle f, f \rangle = \sum_{i,j=1}^{m} \alpha_i \alpha_j K(x_i, x_j) \geq 0. \tag{3.6}$$

Definition 4. *(Reproducing Kernel Hilbert Space [8])*

Let $\mathcal{X}$ be a nonempty set and $\mathcal{H}$ be a Hilbert space of functions $f : \mathcal{X} \to \mathbb{R}$. Then $\mathcal{H}$ is called a reproducing kernel Hilbert space endowed with the inner product if there exists a function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ with the following properties:

$$\langle K(\cdot, x), f \rangle = f(x), \quad \forall x \in \mathcal{X}, \forall f \in \mathcal{H}. \tag{3.7}$$

In particular,

$$\langle K(\cdot, x), K(\cdot, y) \rangle = K(x, y). \tag{3.8}$$

This is called reproducing property.

We can show that any positive-definite kernel can be represented as an inner product in a linear space. The reproducing kernel property (3.8) amounts to

$$\langle \Phi(x), \Phi(y) \rangle = K(x, y). \tag{3.9}$$

3.1.3 Kernel Trick

A function $\Phi\colon \mathcal{X} \to \mathcal{H}$ is employed and x is transformed into $\Phi(x)$. An explicit representation of $\Phi(x)$ is not usable in general, however the inner product $\langle \Phi(x), \Phi(y) \rangle$ is expressed by

$$K(x, y) = \langle \Phi(x), \Phi(y) \rangle,$$

where a function K is a positive-definite kernel [8, 55].

We thus are capable of analyzing objects in the high-dimensional feature space $\mathcal{H}$ instead of the input data space $\mathcal{X}$. This substitution is called the kernel trick.

3.2 Kernel-Based Fuzzy c-Means Clustering

Let us apply kernel functions to kernel-based fuzzy c-means clustering [47, 50].

In this section, we assume a transformation into a high-dimensional Euclidean space $\Phi\colon \mathbb{R}^p \to \mathcal{H}$, whereby x_k is mapped into $\Phi(x_k)$.

Objective function of the kernel-based fuzzy c-means is the following:

$$J(U, W) = \sum_{k=1}^{N} \sum_{i=1}^{c} (u_{ki})^m \|\Phi(x_k) - W_i\|_{\mathcal{H}}^2, \tag{3.10}$$

where $\|\cdot\|$ is the norm of $\mathcal{H}$.

The basic procedure **FCM** should be used for iterative minimization, but

$$D_{\mathcal{H}}(x_k, W_i) = \|\Phi(x_k) - W_i\|_{\mathcal{H}}^2$$

should be used instead of $D(x_k, v_i)$. Cluster center is

$$W = (W_1, \ldots, W_c).$$

However, optimal W_i does not have an explicit representation in general. Hence, the dissimilarity for clustering is as follows:

$$
\begin{aligned}
D_{ki} &= D_{\mathcal{H}}(x_k, W_i) \\
&= \langle \Phi(x_k) - W_i, \Phi(x_k) - W_i \rangle \\
&= \langle \Phi(x_k), \Phi(x_k) \rangle - 2\langle \Phi(x_k), W_i \rangle + \langle W_i, W_i \rangle \\
&= \langle \Phi(x_k), \Phi(x_k) \rangle - \frac{2}{\sum_{k=1}^{n} u_{ki}^m} \sum_{j=1}^{n} u_{ji}^m \langle \Phi(x_j), \Phi(x_k) \rangle \\
&+ \frac{1}{(\sum_{k=1}^{n} u_{ki}^m)^2} \sum_{j=1}^{n} \sum_{l=1}^{n} u_{ji}^m u_{li}^m \langle \Phi(x_j), \Phi(x_l) \rangle.
\end{aligned}
$$

Note that $K(x, y) = \langle \Phi(x), \Phi(y) \rangle$, $K_{jl} = \langle \Phi(x_j), \Phi(x_l) \rangle$.

Hence, we use the updating equation D_{ki} and u_{ki} are as follows:

$$
D_{ki} = K_{kk} - \frac{2}{\sum_{k=1}^{n} u_{ki}^m} \sum_{j=1}^{n} u_{ji}^m K_{jk} + \frac{1}{(\sum_{k=1}^{n} u_{ki}^m)^2} \sum_{j=1}^{n} \sum_{l=1}^{n} u_{ji}^m u_{li}^m K_{jl}, \tag{3.11}
$$

$$
u_{ki} = \frac{\left(\frac{1}{D_{ki}}\right)^{\frac{1}{m-1}}}{\sum_{j=1}^{c} \left(\frac{1}{D_{kj}}\right)^{\frac{1}{m-1}}}. \tag{3.12}
$$

On the other hand, the entropy-based fuzzy c-means [43, 45, 46, 50] using a kernel function is used for the following objective function:

$$
J_{efcm}(U, V) = \sum_{k=1}^{N} \sum_{i=1}^{c} u_{ki} \|\Phi(x_k) - W_i\|_{\mathcal{H}}^2 + \lambda^{-1} \sum_{i=1}^{c} \sum_{k=1}^{N} u_{ki} \log u_{ki}. \tag{3.13}
$$

The dissimilarity D_{ki} is modified to:

$$\begin{aligned} D_{ki} &= D_{\mathcal{H}}(x_k, W_i) \\ &= \langle \Phi(x_k) - W_i, \Phi(x_k) - W_i \rangle \\ &= \langle \Phi(x_k), \Phi(x_k) \rangle - 2\langle \Phi(x_k), W_i \rangle + \langle W_i, W_i \rangle \\ &= \langle \Phi(x_k), \Phi(x_k) \rangle - \frac{2}{\sum_{k=1}^{n} u_{ki}} \sum_{j=1}^{n} u_{ji} \langle \Phi(x_j), \Phi(x_k) \rangle \\ &+ \frac{1}{(\sum_{k=1}^{n} u_{ki})^2} \sum_{j=1}^{n} \sum_{l=1}^{n} u_{ji} u_{li} \langle \Phi(x_j), \Phi(x_l) \rangle. \end{aligned}$$

We thus obtain the updating equation D_{ki} and u_{ki} of entropy-based kernel fuzzy c-means:

$$D_{ki} = K_{kk} - \frac{2}{\sum_{k=1}^{n} u_{ki}} \sum_{j=1}^{n} u_{ji} K_{jk} + \frac{1}{(\sum_{k=1}^{n} u_{ki})^2} \sum_{j=1}^{n} \sum_{l=1}^{n} u_{ji} u_{li} K_{jl}, \tag{3.14}$$

$$u_{ki} = \frac{\exp(-\lambda D_{ki})}{\sum_{j=1}^{c} \exp(-\lambda D_{ki})}. \tag{3.15}$$

We use the next algorithm instead of FCM [45, 49] as follows:

KFCM (Kernel-based Fuzzy c-Means) Algorithm.

KFCM1. Set an initial value D_{ki}.

KFCM2. Find solution of $U = (u_{ki})$ using (3.11).

(For eKFCM, use (3.15).)

KFCM3. Update to D_{ki} using (3.12).

(For eKFCM, use (3.14).)

KFCM4. If the solution $U = (u_{ki})$ is convergent, stop; else go to **KFCM2**.

End KFCM.

3.3 Kernel-Based Vector Quantization Clustering

Various discussions of SOM and LVQ based on kernel function have been found [13, 15, 31, 37], and we consider kernel VQ clustering proposed by Inokuchi et al., [38].

Let us remind the algorithm of vector quantization clustering in which the updating equations for codevector are

$$v_l(t+1) = v_l(t) + \alpha[x(t) - v_l(t)],$$

$$v_i(t+1) = v_i(t),\ i \neq l,$$

$$(t = 1, 2, \ldots).$$

Instead of the distance $\|x(t) - v_i(t)\|$ in the input data space, the next distance in the high-dimensional feature space is considered as below:

$$d_{ik} = \|\Phi(x_k) - v_i(t)\|^2_{\mathcal{H}}. \tag{3.16}$$

It should be noted that v_i is the codevector in the high-dimensional feature space. Hence, the updating equation of the codevector is as follows:

$$v_l(t+1) = v_l(t) + \alpha[\Phi(x_h) - v_l(t)], \tag{3.17}$$

where it should be noted that x_h is randomly selected input vector from $\mathcal{X}$.

Since we do not know a concrete form of $\Phi(x)$ and $v_l(t)$, we cannot use $v_l(t+1)$ explicitly. And then, the algorithm should be rewritten using d_{ik} and kernel function. We thus calculate $d_{ik}(t+1)$ using $d_{ik}(t)$ and $d_{lk}(t)$:

$$\begin{aligned} d_{lk}(t+1) &= \|\Phi(x_k) - v_l(t+1)\|^2 \\ &= \langle\Phi(x_k), \Phi(x_k)\rangle - 2\langle\Phi(x_k), v_l(t+1)\rangle + \langle v_l(t+1), v_l(t+1)\rangle. \end{aligned} \tag{3.18}$$

The distance $d_{lk}(t+1)$ in the high-dimensional space is calculated by substituting (3.17) into (3.18).

$$\begin{aligned} d_{lk}(t+1) = \langle\Phi(x_k), \Phi(x_k)\rangle &- 2\{(1-\alpha)\langle\Phi(x_k), v_l(t)\rangle + \alpha\langle\Phi(x_k), \Phi(x_h)\rangle\} \\ &+ \left\{(1-\alpha)^2\langle v_l(t), v_l(t)\rangle + 2\alpha(1-\alpha)\langle\Phi(x_h), v_l(t)\rangle + \alpha^2\langle\Phi(x_h), \Phi(x_h)\rangle\right\}. \end{aligned} \tag{3.19}$$

Subsequently, distance $d_{lk}(t+1)$ in the high-dimensional space is given by the following

formula:

$$\begin{aligned} d_{lk}(t+1) &= (1-\alpha)\{\langle\Phi(x_k),\Phi(x_k)\rangle - 2\langle\Phi(x_k),v_l(t)\rangle + \langle v_l(t),v_l(t)\rangle \\ &\quad - \alpha(1-\alpha)\{\langle\Phi(x_h),\Phi(x_h)\rangle - 2\langle\Phi(x_k),v_l(t)\rangle + \langle v_l(t),v_l(t)\rangle\} \\ &\quad + \alpha\langle\Phi(x_k),\Phi(x_k)\rangle - 2\alpha\langle\Phi(x_k),\Phi(x_h)\rangle + \alpha\langle\Phi(x_h),\Phi(x_h)\rangle \\ &= (1-\alpha)d_{lk}(t) - \alpha(1-\alpha)d_{lh}(t) + \alpha(K_{kk} - 2K_{kh} + K_{hh}), \end{aligned} \tag{3.20}$$

where $K_{ij} = K(x_i, x_j)$. Allocation of an object x_k to a cluster should use minimum of d_{ik}, $(i = 1, \ldots, c)$. The update of the high-dimensional codevectors can be made indirectly. It is not necessary to keep the transformed codevectors which play the same role of the codevectors in the input space.

We do not know a concrete form of $\Phi(x_k)$ and v_i explicitly. Nevertheless, we are able to know the distance between input data and codevector by using the kernel function.

We now have the kernel-based vector quantization clustering (KVQc) algorithm [38, 45] as follows:

KVQc (Kernel Vector Quantization Clustering) Algorithm.

KVQc1. Fix c, initialize $d_{ik}, i = 1, \ldots, c, k = 1, \ldots, n$, and learning rate $\alpha \in (0, 1)$.

Set $t = 0$ and repeat from KVQc2 to KVQc4 until convergence.

KVQc2. Select $x_h(t) \in X$ randomly.

KVQc3. For $k = 1, \ldots, n$

a. $d_{lk}(t) = \arg\min_{1 \le i \le c} d_{ik}(t)$,

b. Update $d_{lk}(t)$ by (3.20),

and then allocate x_k to cluster l.

KVQc4. Adjust learning rate α. $t = t + 1$.

End KVQc.

In this algorithm, the parameter $\alpha(t)$ should be taken to satisfy the following:

$$\sum_{t=1}^{\infty} \alpha(t) = \infty, \sum_{t=1}^{\infty} \alpha^2(t) < \infty, t = 1, 2, \ldots .$$

For example, $\alpha(t) = constant/t$ satisfies these conditions. Note that learning rate α is assumed to decay to zero as t increases.

Chapter 4

Kernel Functions Derived from Fuzzy Clustering

We propose *three basic functions* derived from fuzzy *c*-means and possibilistic clustering. And then, we prove that the *basic functions* are positive-definite kernel functions. The proof is based on the completely monotone function defined by Schönberg [21].

4.1 Three Basic Functions

We firstly remind *c*-means clustering algorithm. It is well-known that a class of fuzzy *c*-means clustering is based on the optimization of an objective function. The matrix U is used as $U = (u_{ki})$, $(k = 1, \ldots, N, i = 1, \ldots, c)$, where u_{ki} means the degree of belongingness of an object x_k to cluster i. In this section, we consider only the optimal solution u_{ki}.

The optimal solution u_{ki} of fuzzy c-means, entropy-based fuzzy c-means and possibilistic clustering are the following:

Table 4.1: The solutions for FCM, eFCM and PCM.

FCM	**eFCM**	**PCM**
$u_{ki} = \frac{\left(\frac{1}{d_{ki}}\right)^{\frac{1}{m-1}}}{\sum_{j=1}^{c}\left(\frac{1}{d_{kj}}\right)^{\frac{1}{m-1}}}$	$u_{ki} = \frac{e^{-\lambda d_{ki}}}{\sum_{j=1}^{c} e^{-\lambda d_{kj}}}$	$u_{ki} = \frac{1}{1+\left(\frac{d_{ki}}{\eta_i}\right)^{\frac{1}{m-1}}}$

Note that the possibilistic c-means (PCM) is identical with the possibilistic clustering which was proposed to address the drawbacks associated with constraint $\sum_{i=1}^{c} u_{ki} = 1$ of fuzzy c-means [39, 40].

We consider three different types of objective functions.

$$J_1(U,V) = \sum_{k=1}^{N}\sum_{i=1}^{c}(u_{ki})^m(\epsilon + D(x_k, v_i)), \qquad (\epsilon > 0) \tag{4.1}$$

$$J_2(U,V) = \sum_{k=1}^{N}\sum_{i=1}^{c}\{u_{ki}D(x_k, v_i) + \lambda^{-1}u_{ki}(\log u_{ki} - 1)\} \tag{4.2}$$

$$J_3(U,V) = \sum_{k=1}^{N}\sum_{i=1}^{c}(u_{ki})^m D(x_k, v_i) + \eta\sum_{k=1}^{N}(1-u_{ki})^m \tag{4.3}$$

J_1 is similar to the well-known objective function proposed by Bezdek [23] and Dunn [25], but a positive parameter ϵ is added. This objective function has been proposed by Ichihashi et al., [50]. When ϵ approaches zero, J_1 approaches to the standard objective function, and it is also known that the solution converges to the standard solution. J_2 is an entropy-based method proposed by several authors (e.g., [43, 45, 46]). J_3 is the objective function for the possibilistic clustering [39, 40, 41, 50].

Let us define *basic functions* $F_\ell(x, y)$ ($\ell = 1, 2, 3$) for the respective objective functions.

Basic functions for fuzzy *c*-means and possibilistic clustering

$$F_1(x, y) = \frac{1}{(\epsilon + \|x - y\|^2)^{\frac{1}{m-1}}}, \tag{4.4}$$

$$F_2(x, y) = \exp(-\lambda\|x - y\|^2), \tag{4.5}$$

$$F_3(x, y) = \frac{1}{1 + (\|x - y\|^2/\eta)^{\frac{1}{m-1}}}. \tag{4.6}$$

By using *basic functions*, the solution $u_{ki}^{(\ell)}$ for J_ℓ ($\ell = 1, 2$) are represented as

$$u_{ki}^{(\ell)} = \frac{F_\ell(x_k, v_i)}{\sum_{j=1}^{c} F_\ell(x_k, v_j)}. \tag{4.7}$$

Table 4.2: The solution $u_{ki}^{(\ell)}$ for J_ℓ ($\ell = 1, 2$).

Solution u_{ki} of J_1	Solution u_{ki} of J_2
$u_{ki}^{(1)} = \frac{\frac{1}{(\epsilon+\|x_k-v_i\|^2)^{\frac{1}{m-1}}}}{\sum_{j=1}^{c} \frac{1}{(\epsilon+\|x_k-v_j\|^2)^{\frac{1}{m-1}}}}$	$u_{ki}^{(2)} = \frac{\exp(-\lambda\|x_k - v_i\|^2)}{\sum_{j=1}^{c} \exp(-\lambda\|x_k - v_j\|^2)}$

Note that J_3 cannot be used for fuzzy *c*-means clustering. On the other hand, solutions of cluster centers are as follows:

$$v_i = \frac{\sum_{k=1}^{N} (u_{ki})^m x_k}{\sum_{k=1}^{N} (u_{ki})^m}, \tag{4.8}$$

where $m = 1$ for J_2.

In possibilistic clustering, the constraint M_{fcm} of fuzzy c-means is not imposed on $U = (u_{ki})$. Rather, we assume

$$M = (\mathbb{R}^+)^{cn}. \tag{4.9}$$

To put otherwise, no practical constraint for u_{ki}. When J_1 is assumed, the solution u_{ki} is trivial, and hence J_1 objective function is useless in possibilistic clustering.

On the other hand, J_2 and J_3 can be used for possibilistic clustering, and the solution $u_{ki}^{(\ell)}$ for J_ℓ ($\ell = 2, 3$) are represented by the basic functions:

$$u_{ki}^{(\ell)} = F_\ell(x_k, v_i), \tag{4.10}$$

for J_2 and J_3.

The solution for the cluster center is given by the same (4.8).

We note J_2 is applicable to both fuzzy c-means and possibilistic clustering [45, 46, 51]. Hence, the solutions for possibilistic clustering are as in Table. 4.3.

Table 4.3: The solution $u_{ki}^{(\ell)}$ for J_ℓ ($\ell = 2, 3$).

Solution u_{ki} of J_2	Solution u_{ki} of J_3
$u_{ki}^{(2)} = \exp(-\lambda\|x_k - v_i\|^2)$	$u_{ki}^{(3)} = \dfrac{1}{1 + \left(\frac{\|x_k - v_i\|^2}{\eta}\right)^{\frac{1}{m-1}}}$

Note that $F_\ell(x, y)$ is directly employed to solutions u_{ki} in possibilistic clustering by substituting $x = x_k$ and $y = v_i$. We interpret this function to be a degree of effect of x to cluster i. Note also that the solution of fuzzy c-means clustering is the ratio of $F_\ell(x, v_i)$ to the sum of $F_\ell(x, v_j)$ for all cluster $j = 1, \ldots, c$.

4.2 Kernels and Completely Monotone Functions

Recently positive-definite kernel functions and their applications to data analysis have been studied by many researchers, e.g., [4, 8, 9, 33, 36, 44, 48, 49]. It has also been known that completely monotone functions define a class of positive-definite kernels [21].

We introduce a function $f : [0, +\infty) \to \mathbb{R}$ and note the next definition.

Definition 5. *(Schönberg [21])*

A function $f : [0, +\infty) \to \mathbb{R}$ *is said to be completely monotone if* $f^{(2n)}(t) \geq 0$ *and* $f^{(2n-1)}(t) \leq 0$ $(n = 1, 2, 3, \dots)$ *for* $t > 0$ *and* $f(0) = f(+0)$.

In other words, The *odd-numbers order* derivatives of f are *negative* and the *even-numbers order* derivatives of f are *positive*. e.g., $f(t) \geq 0$, $f'(t) \leq 0$, $f''(t) \geq 0$, $f'''(t) \leq 0$, $f^{(4)}(t) \geq 0$, and so forth.

The following theorem has been proved by Schönberg.

Theorem 1. *(Schönberg [21])*

Let $\mathcal{H}$ *be a Hilbert space with norm* $\|\cdot\|_{\mathcal{H}}$. *The function*

$$K(x, y) = f(\|x - y\|^2_{\mathcal{H}})$$

is positive-definite if and only if $f(t)$ *is completely monotone.*

It is easily seen that the Gaussian kernel is associated with the function $f(t) = e^{-t}$ which is obviously completely monotone, and the Gaussian kernel is identical with $F_2(x, y)$. We thus know that basic function $F_2(x, y) = \exp(-\lambda\|x - y\|^2)$ is positive-definite.

In this perspective, we have a question of consideration: Are other basic functions also

positive-definite? We will answer this question affirmatively. We thus consider whether the other basic functions

$$F_1(x,y) = \frac{1}{(\epsilon + \|x-y\|^2)^{\frac{1}{m-1}}}$$

and

$$F_3(x,y) = \frac{1}{1 + (\|x-y\|^2/\eta)^{\frac{1}{m-1}}}$$

are positive-definite functions or not.

We first observe that the next lemma holds.

Lemma 1. *Let* $g\colon [0,+\infty) \to \mathbb{R}$ *be*

$$g(t;\gamma,\delta) = (t^b + c)^{-\gamma} t^{-\delta},$$

where $0 < b \leq 1$ *and* $c > 0$ *are constants, while* γ *and* δ *are positive parameters. Notice that* $g(t;\gamma,\delta) > 0$*. Then*

$$g'(t;\gamma,\delta) = -\gamma b g(t;\gamma+1,\delta+1-b) - \delta g(t;\gamma,\delta+1).$$

In particular, we have $g'(t;\gamma,\delta) < 0$*.*

The proof is straightforward and is omitted. We then have the following proposition.

Proposition 1. *A function* $f\colon [0,\infty) \to \mathbb{R}$

$$f(t) = (t^b + c)^{-a}, \tag{4.11}$$

where $a > 0$*,* $0 < b \leq 1$ *and* $c > 0$ *are constants, is completely monotone.*

Proof. We apply Lemma 1 repeatedly. Since higher-order derivatives of $f(t)$ have terms of the form $g(t;\gamma,\delta)$ multiplied by constants, and the sign of constants changes with differentiation, we have $f^{(2n)}(t) > 0$ and $f^{(2n-1)}(t) < 0$ for all positive n. Since $\lim_{t\to+0} f(t) = c^{-a} = f(0)$, the proposition is proved. □

We hence have the next proposition.

Proposition 2. *The three basic functions $F_\ell(x,y)$ ($\ell = 1,2,3$) are all positive-definite.*

Proof. The above form (4.11) includes

$$f_1(t) = (t+\epsilon)^{-\frac{1}{m-1}}, \quad (m > 1, \epsilon > 0) \tag{4.12}$$

and

$$f_3(t) = (t^{\frac{1}{m-1}} + 1)^{-1}, \quad (m > 1). \tag{4.13}$$

We hence have the desired conclusion. □

Table 4.4: Three basic functions are positive-definite kernels.

Name of kernels	$K(x,y)$
F_1-**Kernel**	$\dfrac{1}{(\epsilon + \|\|x-y\|\|^2)^{\frac{1}{m-1}}}$
F_2-**Kernel**	$\exp(-\lambda\|\|x-y\|\|^2)$
F_3-**Kernel**	$\dfrac{1}{1+(\|\|x-y\|\|^2/\eta)^{\frac{1}{m-1}}}$

Chapter 5

Kernel-Based Fuzzy c-Means and Kernel-Based Vector Quantization Clustering Using Different Kernel Functions

We consider an application of three basic functions to kernel FCM and kernel VQ clustering. We show numerical results of non-kernel clustering and kernel clustering using the basic functions. Moreover we compare the results of classification from kernel functions $F_1(x, y)$ and $F_3(x, y)$ with those from the Gaussian kernel $F_2(x, y)$.

5.1 Kernel-Based Fuzzy c-Means Clustering

For the numerical examples of kernel FCM, we used two kinds of synthetic datasets which is called a ball in a circle and two crescents. It is well-known that an ordinary fuzzy and crisp c-means cannot divide the whole set into two clusters of the ball and the circle in Fig. 1.1 and the upside crescent and the downside crescent in Fig. 1.2, since the classification boundary is nonlinear.

Figure 5.1 shows the datasets classified perfectly into the ball and the circle. It is known that the results from a Gaussian kernel function can be perfect as shown in this figure. It has been observed that the other kernel functions $F_1(x, y)$ and $F_3(x, y)$ can also divide the ball and the circle perfectly as in Fig. 5.1.

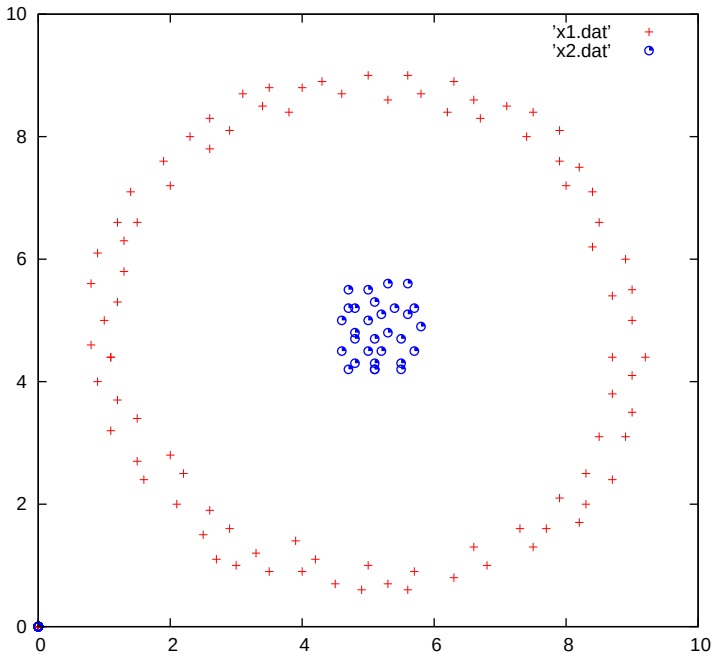

Figure 5.1: An output of kernel fuzzy c-means using $K(x, y) = F_\ell(x, y),\ (\ell = 1, 2, 3)$.

In the case of two crescents dataset, Figure 5.2 shows well-classified results from kernel function $F_1(x, y)$ where the parameters are $m = 2.0$ and $\epsilon = 1.0$. Figure 5.4 shows the re-

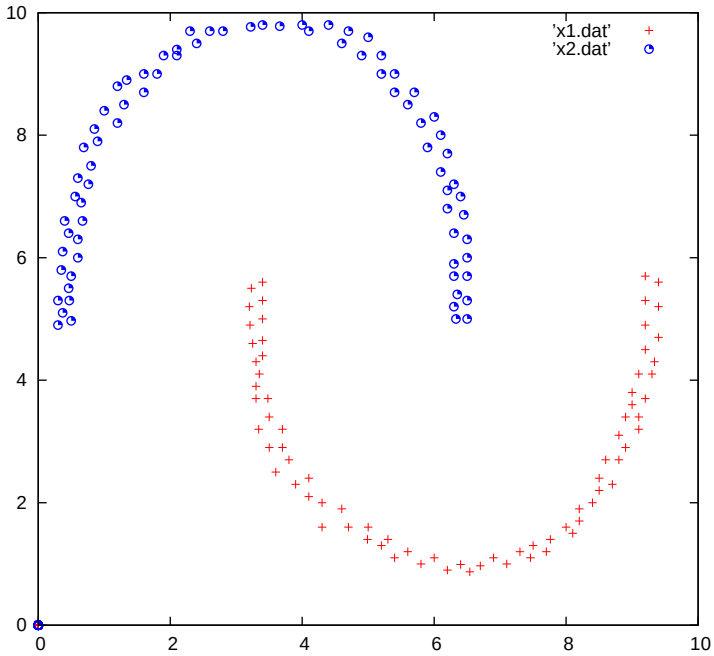

Figure 5.2: An output of kernel fuzzy c-means using $K(x, y) = F_1(x, y)$.

sults from $F_3(x, y)$ where the parameters are $m = 2.0$ and $\eta = 1.0$. Kernel function $F_3(x, y)$ can also divide the upside crescent and downside one. However, KFCM using kernel function $F_2(x, y)$ cannot separate them into two distinct classes mostly, and it produces many misclassifications. A typical output is shown in Fig. 5.3, where the parameter is $\lambda = 1.0$.

In contrast, $F_1(x, y)$ and $F_3(x, y)$ have less misclassifications. Moreover the method of the standard fuzzy c-means seems to work better than the entropy-based fuzzy c-means, as in Tables 5.1–5.4.

In summary, we describe the numerical examples of KFCM using the three kernel func-

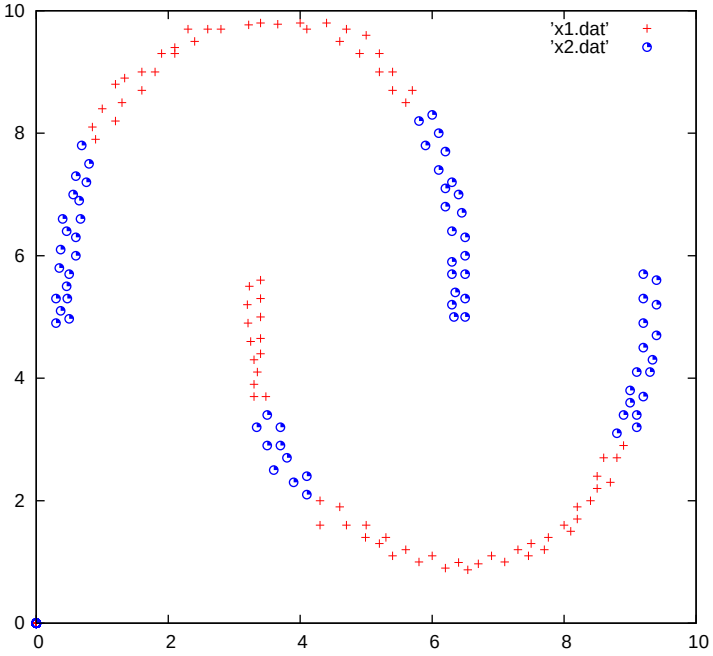

Figure 5.3: An output of kernel fuzzy c-means using $K(x, y) = F_2(x, y)$.

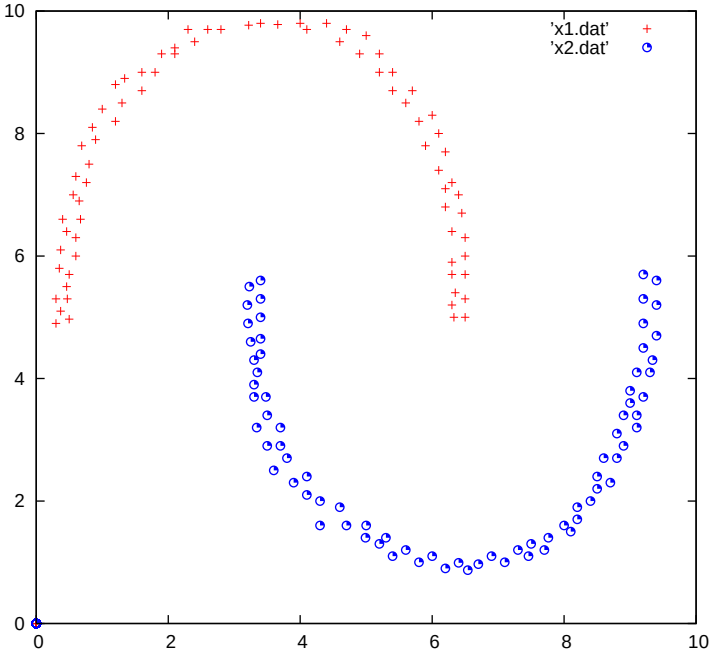

Figure 5.4: An output of kernel fuzzy c-means using $K(x, y) = F_3(x, y)$.

Table 5.1: The result of KFCM using $F_1(x,y)$, $F_2(x,y)$ and $F_3(x,y)$ for a ball in a circle datasets (50 times).

Number of errors	F_1**-Kernel**	F_2**-Kernel**	F_3**-Kernel**
$0 \sim 12(under10\%)$	**50**	**50**	**50**
$13 \sim 24(under20\%)$	0	0	0
$25 \sim 36(under30\%)$	0	0	0
$37 \sim (over30\%)$	0	0	0

Table 5.2: The result of entropy-based KFCM using $F_1(x,y)$, $F_2(x,y)$ and $F_3(x,y)$ for a ball in a circle datasets (50 times).

Number of errors	F_1**-Kernel**	F_2**-Kernel**	F_3**-Kernel**
$0 \sim 12(under10\%)$	**39**	**44**	**39**
$13 \sim 24(under20\%)$	5	0	6
$25 \sim 36(under30\%)$	2	3	3
$37 \sim (over30\%)$	4	3	2

Table 5.3: The result of KFCM using $F_1(x,y)$, $F_2(x,y)$ and $F_3(x,y)$ for two crescents datasets (50 times).

Number of errors	F_1**-Kernel**	F_2**-Kernel**	F_3**-Kernel**
$0 \sim 15$(*under*10%)	**14**	**1**	**11**
$16 \sim 30$(*under*20%)	10	3	13
$31 \sim 45$(*under*30%)	8	8	12
$46 \sim$ (*over*30%)	18	38	14

Table 5.4: The result of entropy-based KFCM using $F_1(x,y)$, $F_2(x,y)$ and $F_3(x,y)$ for two crescents datasets (50 times).

Number of errors	F_1**-Kernel**	F_2**-Kernel**	F_3**-Kernel**
$0 \sim 15$(*under*10%)	**7**	**0**	**10**
$16 \sim 30$(*under*20%)	12	5	13
$31 \sim 45$(*under*30%)	13	8	15
$46 \sim$ (*over*30%)	18	37	12

tions for the synthetic datasets. Kernel-based fuzzy c-means (KFCM) and entropy-based KFCM using $F_1(x,y)$, $F_2(x,y)$ and $F_3(x,y)$ perfectly classified a ball in a circle. However, it has been observed that the Gaussian kernel function cannot divide the two crescents dataset in Fig. 5.3, and $F_1(x,y)$ and $F_3(x,y)$ of non-Gaussian kernels have better performance than $F_2(x,y)$. Calculations were repeated 50 times for each kernel with different initial random values.

5.2 Kernel-Based Vector Quantization Clustering

In this section, we consider an application of three basic functions to kernel VQ clustering discussed by Inokuchi et al., [38] (see also [15], [37]). We show numerical results of non-kernel VQ clustering and kernel VQ clustering using the three basic functions. Moreover we compare the results from kernel functions $F_1(x,y)$ and $F_3(x,y)$ with those from the Gaussian kernel $F_2(x,y)$. In the numerical examples, we demonstrate the usefulness of the present kernel functions for synthetic datasets and several real-world datasets.

5.2.1 Numerical Examples for synthetic datasets

We used two kinds of synthetic datasets.

- A ball in a circle: 100 data points from two classes, which include 40 points in an inner ball and 60 points in a surrounding outer circle as Fig. 1.1.

- Two crescents: 100 data points from two classes shaped like a crescent as Fig. 1.2.

The results of KVQ clustering using the present kernel functions are shown in Fig. 5.5 and Fig. 5.6. Figure 5.5 shows the datasets classified perfectly into the ball and the circle. It is well known that an ordinary fuzzy and hard c-means and non-kernel VQ clustering cannot divide the whole set into such clusters, because the classification boundary is non-linear. It is also known that the result from a kernel function $F_2(x, y)$ (Gaussian kernel) can be perfectly classified as shown in Fig. 5.5. It has been confirmed that the other kernel functions $F_1(x, y)$ and $F_3(x, y)$ can also divide these datasets.

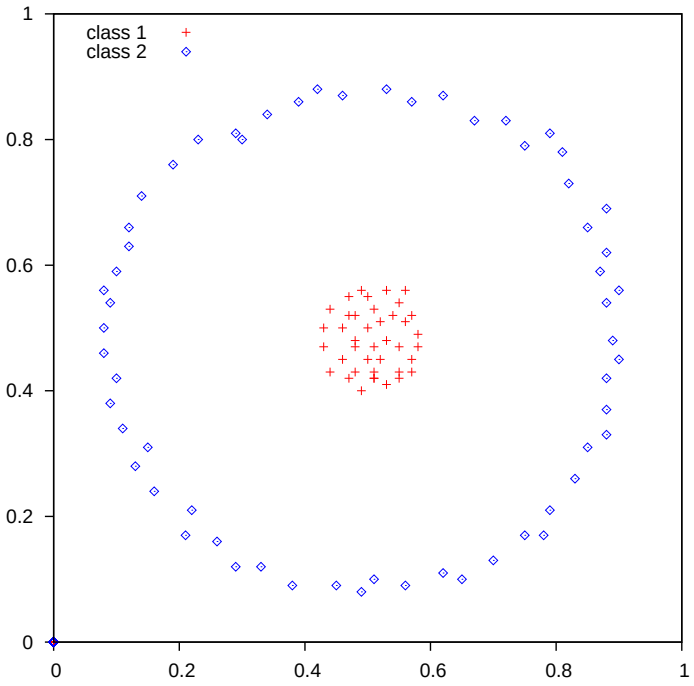

Figure 5.5: An output of kernel VQ clustering using $K(x, y) = F_\ell(x, y), (\ell = 1, 2, 3)$.

Two crescents are also classified, producing results with errors (see Fig. 5.6). Generally it is more difficult to divide this datasets into two clusters of crescents. KVQ clustering using kernel functions $F_2(x, y)$ and $F_3(x, y)$ cannot divide these datasets at all.

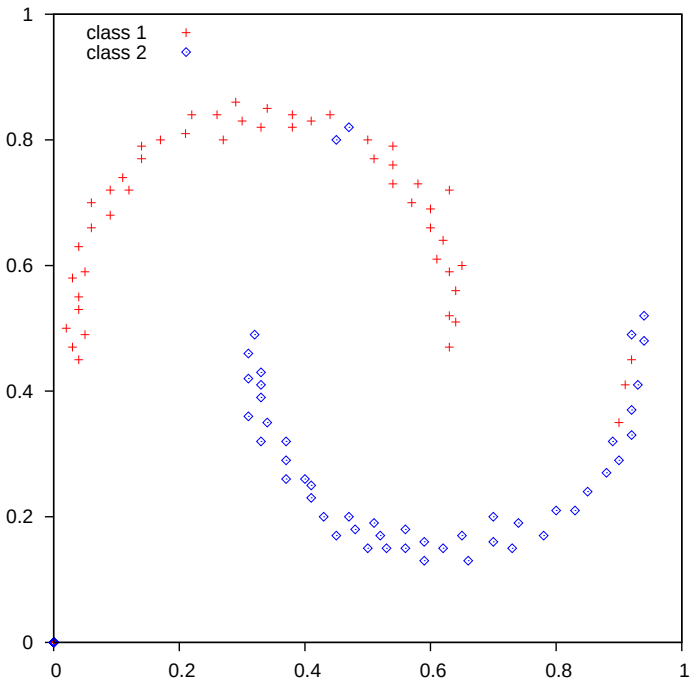

Figure 5.6: An output of kernel VQ clustering using $K(x, y) = F_1(x, y)$ with $m = 35.0$, $\epsilon = 1.0$.

We calculated the ratio of well-classified data and measured the running time in each trial. Note that "well-classified" means that the total number of errors is less than 10% to whole dataset. We presented the results of KVQ clustering using the present kernel functions for the synthetic datasets in Table 5.5. Calculations were repeated 100 times for each kernel with initial random values.

Table 5.5: Averaged results of KVQ clustering using the three kernel function applied to the synthetic datasets.

	F_1-Kernel		F_2-Kernel		F_3-Kernel	
	seconds & ratio	parameters	seconds & ratio	parameters	seconds & ratio	parameters
A ball	5.8×10^{-2}	$m = 20.0$	5.4×10^{-2}	$\lambda = 0.03$	$\mathbf{1.5 \times 10^{-2}}$	$\eta = 27.0$
in a circle	**9.5/100**	$\epsilon = 1.0$	5.5/100		9.0/100	$m = 1.98$
Two crescents	$\mathbf{2.6 \times 10^{-1}}$	$m = 35.0$	–	–	–	–
	3.0/100	$\epsilon = 1.0$	–	–	–	–

5.2.2 Numerical Examples for real-world datasets

We also report the results for several real-world datasets, which is taken from UCI machine learning repository [58].

- Iris datasets: Iris datasets are used as a test for clustering algorithms. It contains three classes of 50 instances each, where each class refers to a type of iris plant. One class is linearly separable from the other two classes but the latter classes are not linearly separable from each other.

- Wisconsin Breast Cancer datasets: 683 instances from two classes which include 444 benign instances and 239 malignant instances. We removed 16 instances with missing attribute values in original datasets. Each instance has 11 attributes including an ID number.

- Glass Identification datasets: 214 instances from two classes which include 163 window glass (building windows and vehicle windows) and 51 non-window glass. Each instance has 10 attributes including an ID number.

For non-kernel VQ clustering, setosa (class 1) of IRIS datasets is classified from versicolor (class 2) and virginica (class 3). However it is difficult to classify versicolor and virginica, since those classes cannot be linearly separated from each other as Fig. 5.7.

The kernel functions discussed above work on classification for datasets with nonlinear boundary. It has been observed that the performance of these kernel functions depends on their parameters: $F_1(x,y)$ and $F_3(x,y)$ kernel functions are characterized by two parameters.

For $F_1(x,y)$ kernel function, as m is greater, all data points are assigned to the same cluster. If ϵ is more smaller, data points are assigned sparsely as Fig. 5.8.

For $F_3(x,y)$ kernel function, if η is more greater, all data points are assigned to one cluster. As m is smaller, they are assigned sparsely as Fig. 5.10. By using an appropriate value of the parameters, however, Iris datasets are well-classified as Fig. 5.9 and Fig. 5.11.

The ratio of well-classified data to the number of all trial is expressed as a percentage in Table 5.6.

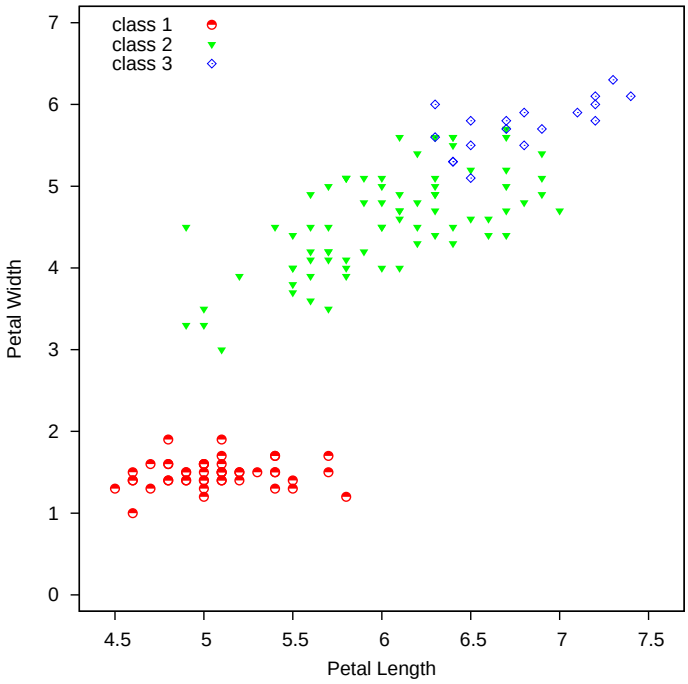

Figure 5.7: An output of non-kernel VQ clustering for Iris datasets.

Table 5.6: Averaged results of Kernel VQ clustering using the three kernel function applied to the real-world datasets.

	non-Kernel	F_1-Kernel		F_2-Kernel		F_3-Kernel	
	seconds & ratio	seconds & ratio	parameters	seconds & ratio	parameters	seconds & ratio	parameters
Iris	5.1×10^{-1}	3.1×10^{-1}	$m = 74.5$	3.0×10^{-2}	$\lambda = 0.0035$	$\mathbf{3.0 \times 10^{-2}}$	$\eta = 21.0$
	–	**2.5/100**	$\epsilon = 0.3$	0.37/100		2.0/100	$m = 1.6$
Wisconsin	–	$\mathbf{2.92 \times 10^{-1}}$	$m = 55$	1.73×10^{0}	$\lambda = 0.001$	3.44×10^{-1}	$\eta = 225.0$
Breast Cancer	–	22/100	$\epsilon = 2.1$	30/100		**33/100**	$m = 1.9$
Glass	–	1.1×10^{-1}	$m = 35$	1.45×10^{-1}	$\lambda = 0.004$	$\mathbf{2.8 \times 10^{-2}}$	$\eta = 40.0$
Identification	–	8/100	$\epsilon = 1.8$	9/100		**17/100**	$m = 1.6$

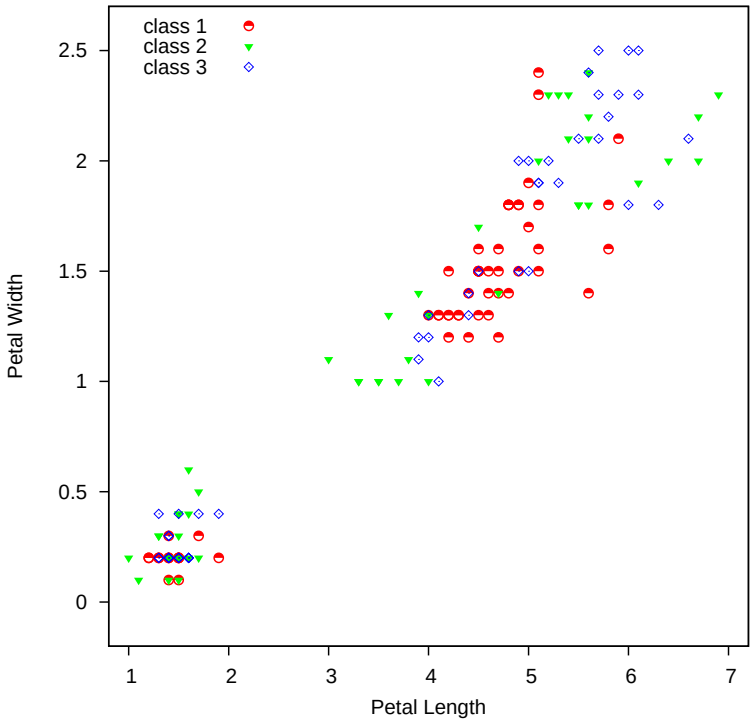

Figure 5.8: An output of kernel VQ clustering using $K(x, y) = F_1(x, y)$ with $m = 2.0$, $\epsilon = 1.0$.

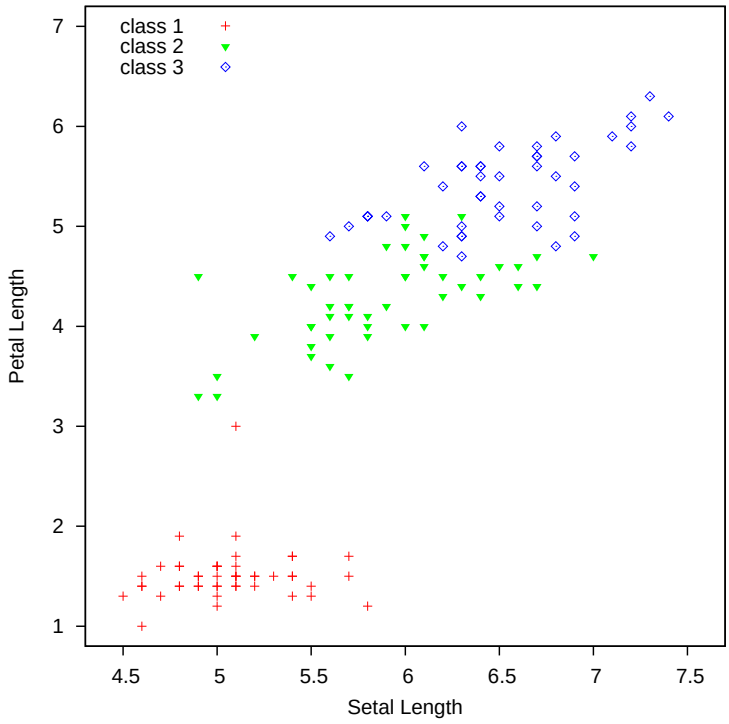

Figure 5.9: An output of kernel VQ clustering using $K(x, y) = F_1(x, y)$ with $m = 75.0$, $\epsilon = 1.0$.

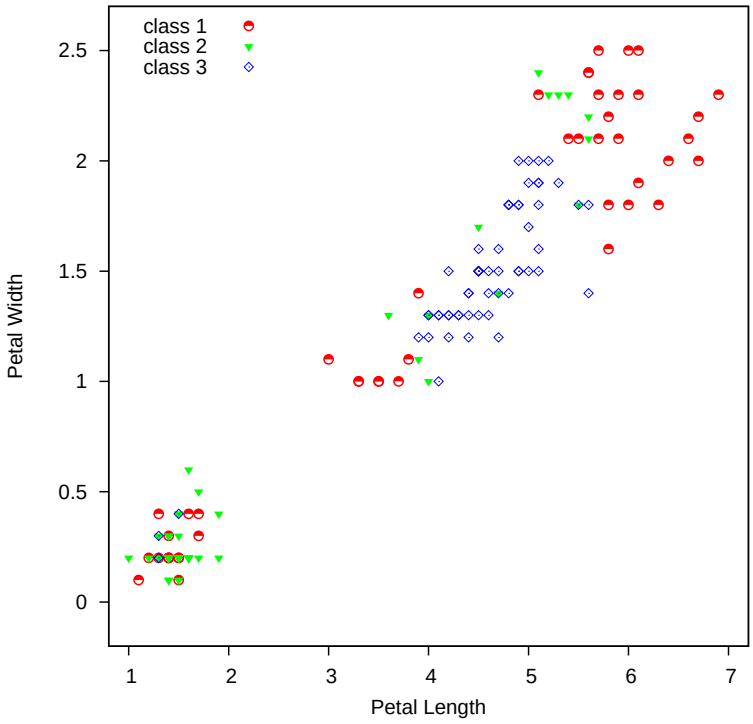

Figure 5.10: An output of kernel VQ clustering using $K(x,y) = F_3(x,y)$ with $m = 1.6$, $\eta = 1.0$.

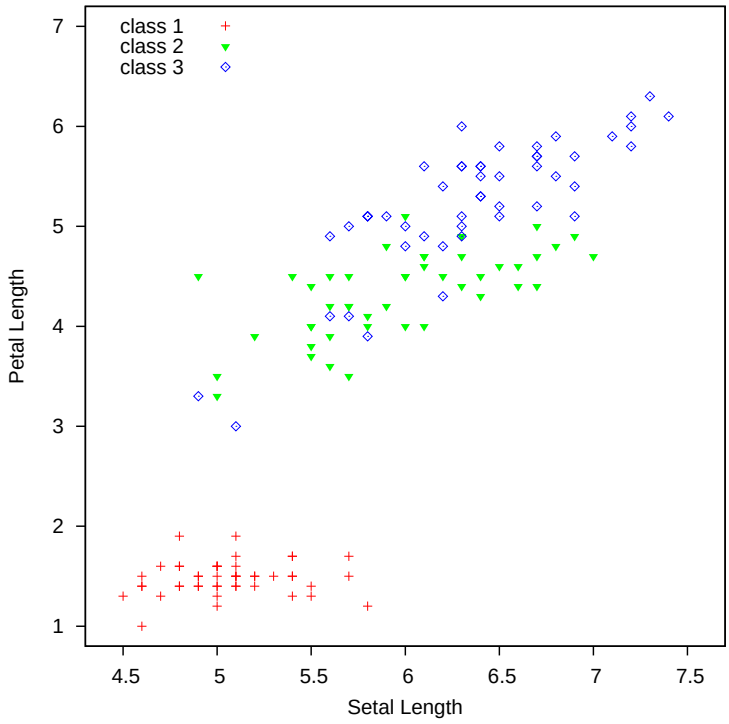

Figure 5.11: An output of kernel VQ clustering using $K(x, y) = F_3(x, y)$ with $m = 1.6$, $\eta = 21.0$.

In summary, the performance of kernel functions depends heavily on their parameters. KVQ clustering based on kernel $F_1(x,y)$ and $F_3(x,y)$ have frequently shown good classification and fast running time on synthetic datasets. Likewise on real-world datasets, it have been observed that datasets are frequently well-classified and faster than the Gaussian kernel.

Chapter 6

Conclusion

We have defined the basic functions that are basic components of solutions in fuzzy *c*-means clustering and possibilistic clustering. We have proved that they are positive-definite kernels using the completely monotone functions by Schönberg [21]. This fact itself is useful in the sense that we have a larger class of kernel functions. Moreover we have shown that the non-Gaussian kernel functions have worked well in typical clustering examples of nonlinear classification boundaries. They have worked better than the Gaussian kernel in the numerical examples in Chapter 5. We also have studied kernel based VQ clustering using the three basic functions. The two non-Gaussian kernels have worked as well as the Gaussian kernel. A reason why the non-Gaussian kernels work well is that they have two parameters while the Gaussian kernel has only one. To summarize, we have found two new kernel functions that are comparable with the Gaussian kernel.

Thus we have succeeded in solving problems addressed in the introduction: to obtain

a new class of kernel functions, to uncover relations between solutions of fuzzy *c*-means clustering and related methods, and to show usefulness of the derived kernel functions.

As a future study, for example, various clustering algorithms as well as kernel-based validity measures should be investigated. Moreover, SVM using the non-Gaussian kernels should be studied. Other problems to be solved in near future are how to select appropriate parameters to the three kernels, and various numerical examples should be tested for this purpose. Furthermore, theoretical studies of fuzzy clustering and related kernel functions should be necessary.

Bibliography

[1] A. B. Hur, D. Horn, H. T. Siegelmann, V. Vapnik, A Support Vector Clustering, *Journal of Machine Learning Research*, Vol. 2, pp. 125–137, 2001.

[2] A. Gersho, On the structure of vector quantizers, *IEEE transactions on Information Theory*, Vol. 28, Issue 2, pp. 157–166, 1982.

[3] A. Gersho, R. M. Gray, *Vector Quantization and Signal Compression*, Kluwer, Boston, 1992.

[4] A. K. Jain, M. N. Murty, P. J. Flynn, Data clustering: a review, *ACM Computing Surveys*, Vol. 31, No. 3, pp. 264–323, 1999.

[5] A. N. Tikhonov, V. Y. Arsenin, *Solutions of Ill-posed of problems*, Wiley, New York, 1977.

[6] A. Y. Ng, M. I. Jordan and Y. Weiss, On Spectral Clustering: Analysis and an Algorithm, *Advances in Neural Information Processing Systems 14*, Vol. 2, pp. 849–856, 2002.

[7] B. Schölkopf, A. Smola, K. R. Müller, Nonlinear Component Analysis as a Kernel Eigenvalue Problem, *Neural Computation*, Vol. 10, No. 5, pp. 1299–1319, 1998.

[8] B. Schölkopf, A. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*, MIT Press, 2002.

[9] C. A. Micchelli, M. Pontil, Learning the kernel function via regularization, *Journal of Machine Learning Research*, Vol. 6, pp. 1099–1125, 2005.

[10] C. Cortes, V. N. Vapnik, Support-Vector Networks, *Journal of Machine Learning*, Vol. 20, Issue 3, pp. 273–297, 1995.

[11] C. J. C. Burges, A Tutorial on Support Vector Machines for Pattern Recognition, *Data Mining and Knowledge Discovery*, Vol. 2, pp. 121–167, 1998.

[12] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[13] D. Bacciu, A. Starita, Expansive competitive learning for kernel vector quantization, *Pattern Recognition Letters*, Vol 30, Issue 6, pp. 641–651, 2009.

[14] D. Horn, Clustering via Hillbert space, *Physica A: Statistical Mechanics and its Applications*, Vol. 302, Issues 1–4, pp. 70–79, 2001.

[15] D. Macdonald, C. Fyfe, The kernel self-organising map, *Proceeding of Fourth International Conference on Knowledge-Based Intelligent Engineering Systems and Allied Technologies 2000*, Vol. 1, pp. 317–320, 2000.

[16] D. S. Satish, C. C. Sekhar, Kernel based clustering and vector quantization for speech recognition, *Machine Learning for Signal Processing, 2004. Proceeding of the 2004 14th IEEE Signal Processing Society Workshop*, pp. 315–324, 2004.

[17] E. Yair, K. Zeger, A. Gersho, Competitive Learning and Soft Competition for Vector Quantizer Design, *IEEE transactions on Signal Processing*, Vol. 40, No. 2, pp. 294–309, 1992.

[18] F. Camastra, A. Verri, A Novel Kernel Method for Clustering, *IEEE Transactions on Pattern analysis and Machine Intelligence*, Vol. 25, No. 5, pp. 801–804, 2005.

[19] F. Höppner, F. Klawonn, R. Kruse, T. Runkler, *Fuzzy Cluster Analysis: Methods for Classification, Data Analysis and Image Recognition*, Wiley, Chichester, 1999.

[20] F. R. Bach, M. I. Jordan, Learning Spectral Clustering, *Technical Report UCB/CSD-03-1249*, EECS Department, University of Califonia, Berkeley, 2003.

[21] I. J. Schönberg, Metric spaces and completely monotone functions, *Annals of Mathematics*, Vol. 39, No. 4, pp. 811–841, 1938.

[22] I. S. Dhillon, Y. Guan, B. Kulis, Kernel *k*-means: spectral clustering and normalized cuts, *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 551–556, 2004.

[23] J. C. Bezdek, *Pattern Recognition with Fuzzy Objective Function Algorithms*, Plenum Press, 1981.

[24] J. C. Bezdek, J. Keller, R. Krishnapuram, N. R. Pal, *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing*, Kluwer, Boston, 1999.

[25] J. C. Dunn, A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters, *Journal of Cybernetics*, Vol. 3, pp. 32–57, 1974.

[26] J. Keller, R. Krisnapuram, M. R. Pal, J. C. Bezdek, *Fuzzy Models and Algorithms for Pattern Recognition and Image Processing* Kluwer Academic Pub, 1999.

[27] J. Lafferty and G. Lebanon, Diffusion Kernels on Statistical Manifolds, *The Journal of Machine Learning Research*, Vol. 6, pp. 129–163, 2005.

[28] J. MacQueen, Some methods for classification and analysis of multivariate observations, *Proceeding of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, University of California Press, Vol. 1, pp. 281–297, 1967.

[29] J. Shawe-Taylor, N. Cristianini, *Kernel Methods for Pattern Analysis*, Cambridge University Press, 2004.

[30] J. Vesanto, E. alhoniemi, Clustering of the self-organizing map, *IEEE transactions on Neural Networks*, Vol. 11, Issue 3, pp. 586–600, 2000.

[31] K. Mizutani, S. Miyamoto, Possibilistic Approach to Kernel-Based Fuzzy *c*-Means Clustering with Entropy Regularization, *2th International conference on Modeling Decisions for Artificial Intelligence*, pp. 144–155, 2005.

[32] M. G. Genton, Classes of Kernels for Machine Learning: A Statistics Perspective, *Journal of Machine Learning Research*, Vol. 2, pp. 299–312, 2002.

[33] M. Girolami, Mercer kernel-based clustering in feature space, *IEEE Transaction on Neural Networks*, Vol. 13, Issue 3, pp. 780–784, 2002.

[34] N. Aronszajn, Theory of Reproducing Kernels, *Transaction of the American Mathematical Society*, Vol. 68, Issue 3, pp. 337–404, 1950.

[35] N. Cristianini, J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, Cambridge University Press, 2000.

[36] O. Chapelle, J. Weston, B. Schölkopf, Cluster Kernels for Semi-Supervised Learning, *Advances in Neural Information Processing Systems*, Vol.15, pp. 585–592, MIT Press, 2003.

[37] P. András, Kernel-Kohonen Network, *International Journal of Neural Systems*, Vol. 12, No. 2, pp. 117–135, 2002.

[38] R. Inokuchi, S. Miyamoto, LVQ Clustering and SOM using a Kernel Function, *Proceeding of IEEE International Conference on Fuzzy Systems*, Vol. 3, pp. 1497–1500, 2004.

[39] R. Krishnapuram, J. M. Keller, A Possibilistic Approach to Clustering, *IEEE Transaction on Fuzzy Systems*, Vol. 1, pp. 98–110, 1993.

[40] R. Krishnapuram, J. M. Keller, The Possibilistic *C*-Means Algorithm: Insignts and Recommendations, *IEEE Transactions on Fuzzy Systems*, Vol. 4, No. 3, pp. 385–393, 1996.

[41] R. N. Davé, R. Krishnapuram, Robust clustering methods: a unified view, *IEEE Transaction on Fuzzy Systems*, Vol. 5, Issue 2, pp. 270–293, 1997.

[42] R. O. Duda, P. E. Hart, D. G. Stork, *Pattern Classification, 2nd ed.*, Wiley, New York, 2001.

[43] R. P. Li, M. Mukaidono, Gaussian clustering method based on maximum-fuzzy-entropy interpretation, *Fuzzy Sets and Systems*, Vol. 102, pp. 253–258, 1999.

[44] S. Akaho, *Kernel Data Analysis*, Iwanami-Shoten, Tokyo, 2008 (in Japanese).

[45] S. Miyamoto, *Introduction to Cluster Analysis: Theory and Applications of Fuzzy Clustering*, Morikita-Shuppan, Tokyo, 1990 (in Japanese).

[46] S. Miyamoto, M. Mukaidono, Fuzzy *c*-means as a regularization and maximum entropy approach, *Proceding of the 7th International Fuzzy Systems Association World Congress (IFSA'97)*, June 25-30, 1997, Prague, Czech, Vol. 2, pp. 86–92, 1997.

[47] S. Miyamoto, D. Suizu, Fuzzy *c*-Means Clustering Using Transformations into High Dimensional Spaces, *Proceeding of FSKD'02: 1st International Conference on Fuzzy Systems and Knowledge Discovery*, Nov. 18-22, 2002, Singapore, Vol. 2, pp. 656–660.

[48] S. Miyamoto, Y. Nakayama, Algorithms of Hard *c*-Means Clustering Using Kernel Functions in Support Vector Machines, *Journal of Advanced Computational Intelligence and Intelligent Informatics*, Vol. 1, No. 1, pp. 19–24, 2003.

[49] S. Miyamoto, D. Suizu, Fuzzy *c*-Means Clustering Using Kernel Functions in Support Vector Machines, *Journal of Advanced Computational Intelligence and Intelligent Informatics*, Vol. 7, No. 1, pp. 25–30, 2003.

[50] S. Miyamoto, H. Ichihashi, K. Honda, *Algorithms for Fuzzy Clustering*, Springer, 2008.

[51] S. Miyamoto, Formulation of Fuzzzy *c*-Means Clustering Using Calculus of Variations and Twofold Membership Clusters, *Journal of Advanced Computational Intelligence and Intelligent Informatics*, Vol. 12, No. 5, 2008.

[52] S. Theodoridis, K. Koutroumbas, *Pattern Recognition*, Academic Press, 2008.

[53] T. Kohonen, *Self Organizing Maps*, Springer, Berlin, 1997.

[54] U. Lusburg, A tutorial on spectral clustering, *Journal of Statistics and Computing*, Vol. 17, No. 4, pp 394–416, 2007.

[55] V. N. Vapnik, *Statistical Learning Theory*, Wiley, NewYork, 1998.

[56] Y. Bengio, P. Vincent, J. F. Paiement, Spectral clustering and kernel PCA are learning eigenfunctions, *Technical Report, No. 1239*, Université de Montréal, Québec, Canada, 2003.

[57] Y. Linde, A. Buzo, R. Gray, An Algorithm for Vector Quantizer Design, *IEEE Transactions on Communications*, Vol. 28, Issue 1, pp. 84–95, 1980.

[58] ”UCI repository of machine learning databases”, `http://archive.ics.uci.edu/ml/index.html`.

List of Publications

1. Jeongsik Hwang, Sadaaki Miyamoto, Kernel Functions Derived from Fuzzy Clustering and Their Application to Kernel Fuzzy c-Means, *Journal of Advanced Computational Intelligence and Intelligent Informatics*, Vol.15, No.1, pp. 90–94, 2011.

2. Jeongsik Hwang, Sadaaki Miyamoto, Kernel Functions Derived from Fuzzy Clustering and Application to VQ Clustering, *Joint 5th International Conference on Soft Computing and Intelligent Systems and 11th International Symposium on Advanced Intelligent Systems*, pp. 390–396, 2010.

3. Sadaaki Miyamoto, Jeongsik Hwang, Kernel Functions Derived from Fuzzy Clustering, *9th International Conference on Modeling Decisions for Artificial Intelligence*, pp. 255–263, 2009.