Chapter 8

Twin Kernel Embedding with Relaxed Constraint

In the last chapter, we introduced a mapping function $f : \mathcal{Y} \rightarrow \mathcal{X}$ explicitly to the TKE and learned the mapping function through the optimization procedure. For new data, the corresponding low dimensional $\mathbf{x}$ can be easily found by $\mathbf{x} = f(\mathbf{y})$. Other DR methods such as LPP, NPE, BCGP, ONPP [69, 70] adopted the same approach. Notice that in actual implementation, the common way is to embed the mapping function into the algorithm by substituting every $\mathbf{x}$ by $f(\mathbf{y})$ and optimizing f (or the parameters in f) instead of $\mathbf{x}$. Another possible approach is to introduce a regularizer reflecting the relationship between input and output represented by the mapping function. By doing so, the constraints between input data and their embeddings can be relaxed a bit thus leading to a smooth mapping. We apply this idea to the TKE in this chapter by defining mapping functions which will be incorporated in the TKE objective function as regularization terms. The mapping functions are derived from kernel feature mapping and the core part of Least Squares Support Vector Machines (LS-SVM). These mapping functions make use of kernel function so that non-vectorial data will be applicable. Moreover, the mapping function will also enable the TKE to handle new data as BCTKE whence its application can be extended greatly.

8.1 TKE with relaxed constraints

8.1.1 Approach 1: Sum-of-square Error

We first consider a simple mapping function derived from kernel machine [120] which is the one used in BCTKE

$$f_j(\mathbf{y}_i) = \sum_m \alpha_{mj} \kappa_y(\mathbf{y}_i, \mathbf{y}_m) \tag{8.1}$$

where α_{mj}'s are the coefficients to be determined. We let $x_{ij} = f_j(\mathbf{y}_i)$ and x_{ij} is the jth component of $\mathbf{x}_i$. The mapping function is parameterized by α_{mj}'s which should be determined according to some criteria.

As mentioned before, we can substitute the form of x_{ij} into objective function of TKE as what have been done in BCTKE to optimize α_{ij}'s rather than $\mathbf{x}_i$. Actually, it can be included in TKE as a regularization term in minimization and therefore relax the constraint a little to smooth the mapping. Further, the regularization term can reflect the relation between input data and their embeddings. The simplest way is to minimize the sum-of-square error between outputs of mapping function and the targets, that is $E = \frac{1}{2}\sum_{ij}(x_{ij} - f_j(\mathbf{y}_i))^2$. Therefore, the revised objective function to be minimized is

$$L = -\sum_{i,j} \kappa_y(\mathbf{y}_i, \mathbf{y}_j)\kappa_x(\mathbf{x}_i, \mathbf{x}_j) + \lambda_k \sum_{ij} \kappa_x(\mathbf{x}_i, \mathbf{x}_j)^2 + \frac{1}{2}\sum_i \|\mathbf{x}_i - \mathbf{f}(\mathbf{y}_i)\|^2 + \frac{\lambda_\alpha}{2}\sum_{ij} \alpha_{ij}^2 \tag{8.2}$$

where $\mathbf{f}(\mathbf{y}_i) = [f_1(\mathbf{y}_i), \ldots, f_d(\mathbf{y}_i)]^\top$ and the coefficient λ_α governs the relative importance of the regularization term on α_{ij}'s. The interpretation of the last term is similar to the shrinkage in statistics literature or weight decay in neural networks. Note that in the revised objective function (8.2), we omitted the regularizer for $\mathbf{x}_i$'s, i.e. $\sum_i \mathbf{x}_i^\top \mathbf{x}_i$, because it is implicitly included in the sum-of-square error term in (8.2). Let

$$\mathbf{A} = \begin{bmatrix} \alpha_{11} & \ldots & \alpha_{1d} \\ \vdots & & \vdots \\ \alpha_{N1} & \ldots & \alpha_{Nd} \end{bmatrix} \tag{8.3}$$

Rewrite the objective function L in matrix form as

$$\begin{aligned} L =& -\operatorname{tr}[\mathbf{K_x K_y}] + \lambda_k \|\mathbf{K_x}\|_F^2 + \frac{1}{2}\|\mathbf{X} - \mathbf{K_y A}\|_F^2 + \frac{\lambda_\alpha}{2}\|\mathbf{A}\|_F^2 \\ =& -\operatorname{tr}[\mathbf{K_x K_y}] + \lambda_k \operatorname{tr}[\mathbf{K_x}^2] + \frac{1}{2}\operatorname{tr}[\mathbf{X X}^\top] - \operatorname{tr}[\mathbf{K_y A X}^\top] \\ &+ \frac{1}{2}\operatorname{tr}[\mathbf{K_y A A}^\top \mathbf{K_y}^\top] + \frac{\lambda_\alpha}{2}\operatorname{tr}[\mathbf{A A}^\top] \end{aligned} \tag{8.4}$$

where $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. Then compared with the original TKE, we have extra parameters in $\mathbf{A}$ to optimize. To minimize (8.4) using a gradient based algorithm, we obtain the derivatives as follows,

$$\frac{\partial L}{\partial \mathbf{X}} = \frac{\partial L}{\partial \mathbf{K_x}} \frac{\partial \mathbf{K_x}}{\partial \mathbf{X}} + \mathbf{X}, \text{ and, } \frac{\partial L}{\partial \mathbf{K_x}} = 2\lambda_k \mathbf{K_x} - \mathbf{K_y}. \tag{8.5}$$

The derivative of L with respect to $\mathbf{A}$ is

$$\frac{\partial L}{\partial \mathbf{A}} = -\mathbf{K_y X} + \mathbf{K_y}^2 \mathbf{A} + \lambda_a \mathbf{A} \tag{8.6}$$

To start the optimization, the $\mathbf{X}$ will be initialized by KPCA or KLE. After $\mathbf{X}$ is given, we use $\frac{\partial L}{\partial \mathbf{A}} = 0$ to get $\mathbf{A}$. So $\mathbf{A} = (\mathbf{K_y}^2 + \lambda_\alpha \mathbf{I})^{-1} \mathbf{K_y X}$. Different from TKE, the whole kernel Gram matrix $\mathbf{K_y}$ is input into the algorithm without n nearest neighbor filtering in this case. The prediction is made by (8.1) or in matrix form as $\mathbf{X} = \mathbf{K_y A}$ after the optimization.

8.1.2 Approach 2: Incorporate LS-SVM objective function

In approach 1, we included the sum-of-square error term as an regularizer in the original TKE objective function to implement the mapping function f. This is derived from kernel machine which implies a certain kernel feature mapping. In essence, we can start with the kernel feature mapping directly and incorporate it as the core part of the LS-SVM (the dual form of the objective function of LS-SVM) into TKE instead of sum-of-square error. This idea is inspired by [126] where LS-SVM was integrated in another NLDR algorithm. We minimizes the following objective function similar to that of LS-SVM with different constraints

$$J = \frac{\upsilon}{2} \sum_{j=1}^{d} \omega_j^\top \omega_j + \frac{\eta}{2} \sum_{ij} e_{ij}^2 \tag{8.7}$$

$$\text{s.t. } x_{ij} = \omega_j^\top \varphi_j(\mathbf{y}_i) + e_{ij} \tag{8.8}$$

corresponding to the maximum margin (the first term) and least square errors (the second term) in (8.7) where υ and η are adjustable coefficients. φ_j's are the feature mappings which map the input $\mathbf{y}_i$ into Hilbert space where the inner product is defined. ω_j is a column vector having the same dimension as the Hilbert space. We can regard the $\mathbf{x}_i$ in (8.8) as the projection of $\varphi(\mathbf{y}_i)$ onto a subspace spanned by ω_j's. Because the difference of the constraints, it does not have the same geometrical interpretation of LS-SVM because the original constraints are inequalities reflecting the correct classification hyperplanes while here they are just feature mapping which build the relation between input data $\mathbf{y}_i$ and its embedding $\mathbf{x}_i$ in low dimensional space. Equation (8.7) can be interpreted as minimizing the error e_{ij} in the reconstruction of $\mathbf{x}_i$ essentially, however it should be done with ω_{ij} properly constrained. This happens to have the form of the LS-SVM objective function. To solve (8.7) with the equality constraints (8.8), we can use Lagrange multipliers as

$$L = \frac{\upsilon}{2}\sum_{j=1}^{d}\omega_j^\top\omega_j + \frac{\eta}{2}\sum_{ij}e_{ij}^2 + \sum_{ij}\alpha_{ij}(x_{ij} - \omega_j^\top\varphi_j(\mathbf{y}_i) - e_{ij}) \tag{8.9}$$

From the saddle points, $\frac{\partial L}{\partial \omega_j} = 0$, $\frac{\partial L}{\partial e_{ij}} = 0$ we have

$$\begin{aligned}
\frac{\partial L}{\partial \omega_j} &= \upsilon\omega_j - \sum_i \alpha_{ij}\varphi_j(\mathbf{y}_i) = 0 \;\Rightarrow\; \omega_j = \frac{1}{\upsilon}\sum_i \alpha_{ij}\varphi_j(\mathbf{y}_i) \\
\frac{\partial L}{\partial e_{ij}} &= \eta e_{ij} - \alpha_{ij} = 0 \;\Rightarrow\; e_{ij} = \frac{1}{\eta}\alpha_{ij}.
\end{aligned}$$

Substitute them back into (8.9) and eliminate ω_j and e_{ij} we have the dual problem to be *maximized* according to the min-max duality

$$\begin{aligned}
L =& \frac{\upsilon}{2}\sum_{j=1}^{d}(\frac{1}{\upsilon}\sum_i \alpha_{ij}\varphi_j(\mathbf{y}_i))^\top(\frac{1}{\upsilon}\sum_i \alpha_{ij}\varphi_j(\mathbf{y}_i)) + \frac{\eta}{2}\sum_{ij}(\frac{1}{\eta}\alpha_{ij})^2 \\
&+ \sum_{ij}\alpha_{ij}\{x_{ij} - (\frac{1}{\upsilon}\sum_i \alpha_{ij}\varphi_j(\mathbf{y}_i))^\top\varphi_j(\mathbf{y}_i) - \frac{1}{\eta}\alpha_{ij}\} \\
=& -\frac{1}{2\upsilon}\sum_j \boldsymbol{\alpha}_j^\top\mathbf{K}_j\boldsymbol{\alpha}_j - \frac{1}{2\eta}\sum_{ij}\alpha_{ij}^2 + \sum_{ij}\alpha_{ij}x_{ij}
\end{aligned} \tag{8.10}$$

where $\boldsymbol{\alpha}_j = (\alpha_{1j}, \ldots, \alpha_{2j})^\top$, $\varphi(\mathbf{y}_m)^\top\varphi(\mathbf{y}_i) = \kappa_j(\mathbf{y}_m, \mathbf{y}_i)$ to which the "kernel trick" applies. We then maximize the above dual problem with respect to α_{ij} instead of ω_j and e_{ij}. From the discussion above we see that the x_{ij}'s are free variables. To limit the choice of them, we combine

(8.10) with the objective function of TKE to incorporate the similarity preserving as

$$L = -\sum_{i,j} k_y(\mathbf{y}_i, \mathbf{y}_j) k_x(\mathbf{x}_i, \mathbf{x}_j) + \lambda_k \sum_{ij} k_x(\mathbf{x}_i, \mathbf{x}_j)^2 + \lambda_x \sum_i \mathbf{x}_i^\top \mathbf{x}_i + \frac{1}{2\upsilon} \sum_j \boldsymbol{\alpha}_j^\top \mathbf{K}_j \boldsymbol{\alpha}_j + \frac{1}{2\eta} \sum_{ij} \alpha_{ij}^2 - \sum_{ij} \alpha_{ij} x_{ij} \tag{8.11}$$

where we turn the maximization of (8.10) to minimization aligned with the TKE objective function and let $\kappa_j(\cdot,\cdot) = \kappa_y(\cdot,\cdot)$ for simplicity. Here we see that the terms related to $\mathbf{y}_i$ are expressed by kernel $\kappa_y(\cdot,\cdot)$. Therefore, this revised objective function is still non-vectorial data applicable. Again we express (8.11) into matrix form to facilitate the differentiation

$$L = -\operatorname{tr}[\mathbf{K_x}\mathbf{K_y}] + \lambda_k \operatorname{tr}[\mathbf{K_x}^2] + \lambda_x \operatorname{tr}[\mathbf{X}\mathbf{X}^\top] + \frac{1}{2\upsilon}\operatorname{tr}[\mathbf{A}^\top \mathbf{K_y} \mathbf{A}] + \frac{1}{2\eta}\operatorname{tr}[\mathbf{A}^\top \mathbf{A}] - \operatorname{tr}[\mathbf{A}^\top \mathbf{X}] \tag{8.12}$$

where $\mathbf{A}$ is the same as that in (8.3). Hence we minimize the above L in (8.11) with respect to $\mathbf{A}$, $\mathbf{X}$ and kernel hyperparameters of $\kappa_x(\cdot,\cdot)$. If we substitute the saddle point solution back into the equality constraints (8.8), the following mapping function can be conveniently applied to predict embeddings of new input samples

$$x_{ij} = \frac{1}{\upsilon} \sum_m \alpha_{mj} \kappa_y(\mathbf{x}_m, \mathbf{x}_i) + \frac{1}{\eta} \alpha_{ij}. \tag{8.13}$$

To apply the gradient based algorithm, the derivatives of L with respect to $\mathbf{X}$ and $\mathbf{A}$ are given by

$$\frac{\partial L}{\partial \mathbf{X}} = \frac{\partial L}{\partial \mathbf{K_x}} \frac{\partial \mathbf{K_x}}{\partial \mathbf{X}} - \mathbf{A}, \text{ and, } \frac{\partial L}{\partial \mathbf{K_x}} = 2\lambda_k \mathbf{K_x} - \mathbf{K_y}, \tag{8.14}$$

$$\frac{\partial L}{\partial \mathbf{A}} = \frac{1}{\upsilon} \mathbf{K_y} \mathbf{A} + \frac{1}{\eta} \mathbf{A} - \mathbf{X}. \tag{8.15}$$

$\mathbf{X}$ is still initialized by KPCA or KLE. $\mathbf{A}$ is initially obtained from the solution of $\frac{\partial L}{\partial \mathbf{A}} = 0$ after $\mathbf{X}$ is given. We have $\mathbf{A} = (\frac{1}{\upsilon}\mathbf{K_y} + \frac{1}{\eta}\boldsymbol{I})^{-1}\mathbf{X}$. It suggests that we could update $\mathbf{X}$ and $\mathbf{A}$ alternately in optimization, however we still update them at the same time. In practice, we use conjugate gradient algorithm for optimization. Both approach 1 and approach 2 involve the relaxed constraints in different forms integrated in TKE, we call the algorithms thus derived the Relaxed Constraint TKE (RCTKE).

8.2 Experimental Results

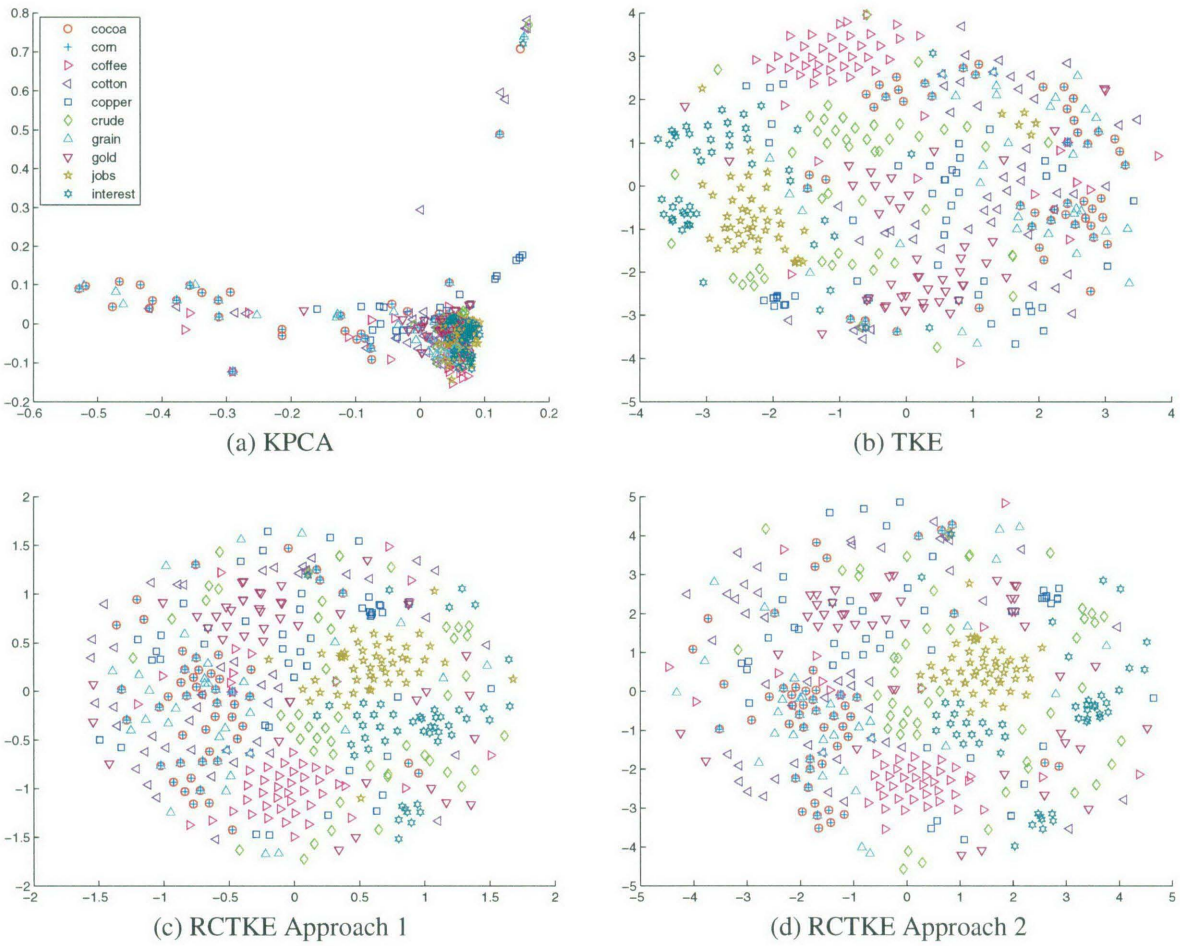
To demonstrate the effectiveness of the proposed RCTKE algorithms, the experiments of visualizing non-vectorial data (the target latent space is a normal plane and hence $d = 2$) were conducted on Reuters-21578 Text Categorization Test Collection and SCOP (Structural Classification Of Protein) database which are recognized as highly structured data. They are also used in Chapter 5.

8.2.1 Parameters Setting

Both RCTKE and TKE have parameters to be determined beforehand. Through empirical analysis (performed a batch of experiments on the same data set varying only the parameters), we found these algorithms are not sensitive to the choice of the parameters, so long as the conjugate gradient optimization can be carried out without premature termination. So for RCTKE and TKE, we use the following parameters throughout the experiments which are determined by experiments. For TKE, $\lambda_k = 0.05$, $\lambda_x = 0.01$ and $n = 10$ in n nearest neighboring; for RCTKE approach 1, $\lambda_k = 0.01$ and $\lambda_\alpha = 1000$; for approach 2, $\lambda_k = 0.05$, $\lambda_x = 0.01$, $\upsilon = \eta = 0.5$. The minimization will stop after 1,000 iterations or when the difference between consecutive updates of the objective function is less than 10^{-7}. $\kappa_x(\cdot, \cdot)$ is the RBF kernel and initialization is given by KPCA for both RCTKE and TKE.

8.2.2 Reuters Texts

We present the result in 2D plane as shown Figure 8.1 with the topics displayed in the legend of (a). The results of RCTKE and TKE reveal clear cluster structure while that of KPCA is indistinguishable. An interesting observation is that the results of both TKE and RCTKE of the texts from topics *cocoa* and *corn* are totally overlapped since the kernel values for the documents in these two categories are very close to 1. This also indicates the similarity preserving ability of RCTKE and TKE, namely, if two objects are similar in the sense of a given kernel, they will stay close together in the latent space. Compare the result of RCTKE with TKE, we observe



(a) KPCA (b) TKE

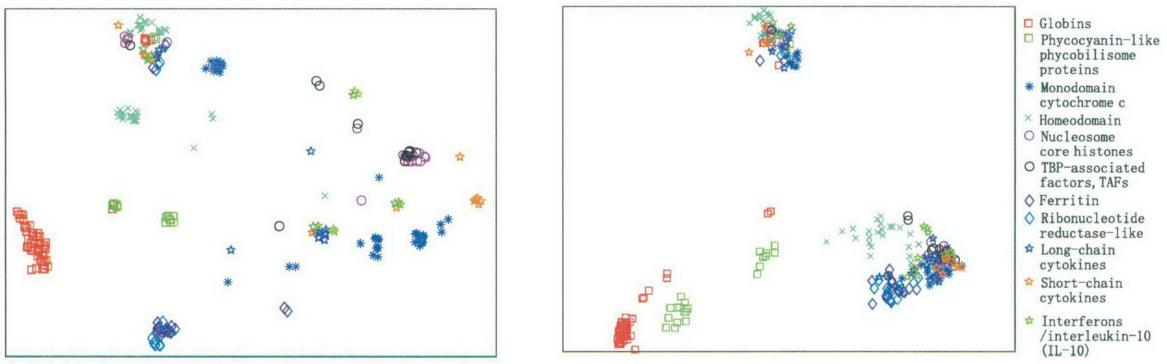
(c) RCTKE Approach 1 (d) RCTKE Approach 2

Figure 8.1: The results on Reuters texts.

that they are quite similar with clear clusters structure while the mapping function obtained in RCTKE which is ready for use to predict embeddings for novel samples, which is absent from the original TKE.

8.2.3 Proteins

For SCOP proteins, we still use the MAMOTH kernel and the results are plotted in Figure 8.2 and 8.3. The result of KPCA is also presented for comparison. Each point (denoted as a shape in the figure) represents a protein. The same shapes with the same colors are the proteins from



(a) TKE (b) KPCA

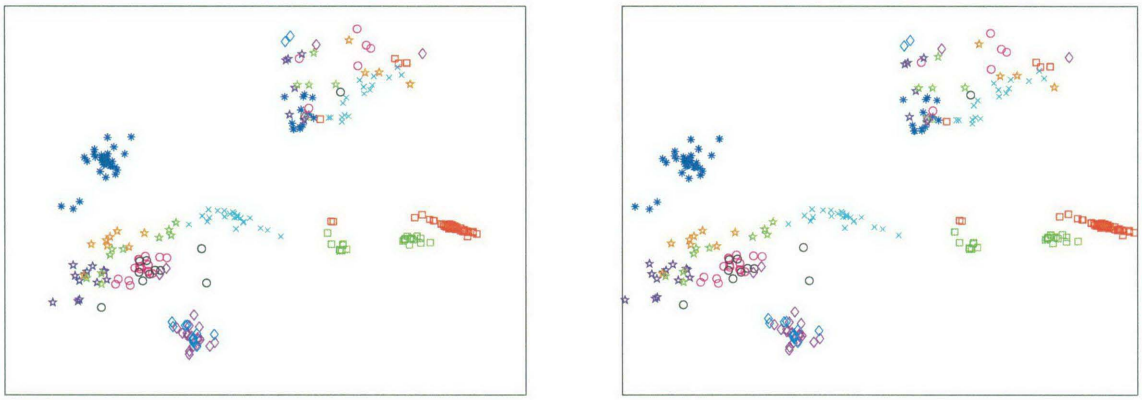
Figure 8.2: The result of TKE and KPCA with MAMOTH kernel

same families while the same shapes with different colors represent the proteins from different families but from the same superfamilies. The legends are shown in Figure 8.2 (b) which are the names of the protein families.

Both RCTKE and TKE revealed the fact that proteins from the same families congregate together as clusters. The proteins locate in groups in two levels: family level and superfamily level. Superfamilies define common features shared by all the proteins in it and these features are further refined in families. Therefore it is natural that the families in the same superfamily cluster together. In Figure 8.3, we can observe that there is little difference between the two approaches of RCTKE in this case. The similar objective function in these two approaches may account for the similar behavior. This experiment also demonstrates that RCTKE has comparable quality as the original TKE. The advantage of RCTKE over TKE is the mapping function learned from training provides an out-of-sample extension for new input samples.

8.3 Summary

In this chapter, we have formulated the mapping functions defined by kernel feature mapping and dual form of LS-SVM objective function in the objective function of TKE as regularization terms and hence generated a new algorithm called RCTKE. Interestingly, although there are more



(a) Approach 1 (b) Approach 2

Figure 8.3: The result of RCTKE with MAMOTH kernel

parameters to be optimized in RCTKE than in TKE, this does not incur much additional computation complexity. However, the mapping function obtained as a result enables TKE to handle out-of-sample extension for novel samples. Hence, similar to BCTKE explained in Chapter 7, RCTKE can equally be applied to classification, manifold learning etc.

Chapter 9

A Learning Framework for Examiner-Centric Fingerprint Classification

9.1 Introduction

In this chapter,we introduce an examiner-centric learning framework for fingerprint classification based on the BCTKE algorithm (cf. Chapter 7) by virtue of exploiting its mapping function for new data. We begin with the basic introduction to fingerprint classification, feature extraction and then present the learning framework constructed on BCTKE.

In recent years, the tasks of fingerprint examiners in law enforcement have been greatly aided by the development of automatic fingerprint classification systems [141]. There are largely two operating scenarios, one for identifying which fingerprint class a particular fingerprint belongs to, and another for deciding which fingerprints in the database that a novel fingerprint might be similar. These two scenarios are respectively called *discrete* and *continuous* classification in the relevant literature [95].

As an automatic fingerprint classification system is a pattern recognition application, its operation relies on comparing salient features extracted from the fingerprint images, normally forming vectors, for its final decision. With these feature vectors, the fingerprints can be projected as points in a very high dimensional space. In the case of discrete classification, due to the existence of only a handful of distinct classes of fingerprints, that include left loop, right loop, whorl, arch and tented arch, in reality the high-dimensional features space could be very sparse. Often the fingerprints are located in clusters in the features space that together form a manifold of a much lower dimension. For the continuous case, ideally fingerprints of the same finger should form compact clusters. However, in reality, the cluster boundaries are not very distinct with much overlapping. This can be caused by a number of reasons like occlusion, scars, etc, leading to high intra-class variations among impressions of the same finger and low inter-class variations among impressions from different fingers. Here, we have focused mainly on the *continuous* classification case.

No matter we are dealing with either discrete or continuous classification, there are two major shortcomings with current automatic fingerprint classification systems. First, the result of classification depends solely on the features selected and the algorithm that matches these features. Second, there is no way of having the systems adapt the result to individual fingerprint examiners, who often have different level of experiences, over time. Taking these two problems together, we can say for the same fingerprint, the result of classification (whether discrete or continuous) will be identical for the same user regardless of who uses the system and how many times he or she has interacted with it.

To address these problems, in this chapter we introduce a personalized learning framework that can adapt and improve the classification result for individual fingerprint examiners through repeated interactions with the system. This is achieved by exploiting relevance feedback from a fingerprint examiner by means of choosing both positive and negative examples in an iterative fashion. The outcome is a personalized and persistent semantic space for each fingerprint examiner in which the quality of classification is improved. This idea resembles long-term learning in Content-based Image Retrieval research proposed in [55, 54], but differs in both the application and the method of learning. The fingerprint features that induce the original features space from

which individual semantic spaces are being learned are obtained by multispectral decomposition of fingerprints (taken as texture images) using a bank of Gabor filters aligned in eight different directions.

In this research, the learning model can be considered as a dimensionality reduction process in which the features space corresponds to the input space while the semantic space to the embedding (or latent) space. From this, we apply BCTKE to incrementally learn both the semantic space and the mapping function by which novel fingerprints can be classified in the semantic space [47, 48]. Experimental evaluation conducted on a subset of the open Fingerprint Verification Competition 2002 (FVC2002) Db2 database reveals the potential of this learning framework for examiner-centric fingerprint classification [94].

The learning framework will be developed as follows. In Section 9.2, a diagram of the learning framework is presented, with major components highlighted and explained. In Section 9.3, the procedure of extracting spectral features by multispectral decomposition of fingerprints using Gabor filters will be briefly described. In Section 9.4, the relevance feedback process and the dimensionality reduction algorithm by which the examiner-centric semantic space is learned will be explained. In Section 9.5, experimental evaluation using a subset of the FVC2002's Db2 database will be presented. Finally, we will conclude and mention future directions in Section 9.6.

9.2 Overview Of Learning Framework

The proposed learning framework consists of two major components, namely *Features Extraction and Semantic Space Learning*. Semantic Space Learning is in turn comprised by the *relevance feedback* and the *dimensionality reduction* modules. A diagram illustrating the relationship between these components and modules, together with their input and output, is shown in Figure 9.1. In the remaining of this section, we will briefly overview the role of each of these main components and modules while leaving further details to their respective sections.

First, the input to the *Features Extraction* component is a set of m fingerprint images that will altogether constitute the database. The size and resolution of these images depend on the

particular fingerprint scanner that was used. Within this module, a series of image processing steps are performed, like image enhancement, segmentation of fingerprint regions, detection and extraction of fingerprint features, and mapping features to numeric values. The set of numeric values will be merged to form a vector. Depending on the number of features involved, each of these vectors can be viewed as a point in a high-dimensional Euclidean space of dimension D. The distance between any pair of fingerprints, i and j, can be measured by the Minkowski distance of order p (i.e., $\|v_i - v_j\|_p$) of the difference vector. Based on these distances, an m by m distance matrix can be obtained.

Second, a fingerprint examiner interacts with the classification system via the *relevance feedback* module. In the training phase, an image q in the database can either be picked randomly by the system or chosen by the examiner. Based on the image selected, the system returns a subset of images (excluding q) that are similar based on the nearest neighbor criterion. The examiner indicates as positive examples those images that are judged similar based on prior experiences. The negative examples are those that are judged dissimilar. With the positive and negative examples, the corresponding entries in the distance matrix will be decreased or increased accordingly. The relevance feedback loop repeats until the examiner decides to exit. The outcome is a distance matrix that has learnt the semantic judgment of the examiner.

Finally, through the *dimensionality reduction* module, both the d-dimensional ($d \ll D$) semantic space of the examiner and a mapping function that is capable of embedding a novel fingerprint in the semantic space *directly* can be obtained. Here, BCTKE is the inference engine behind this DR module. By mapping novel fingerprints directly onto the personalized semantic space, examiner-centric fingerprint classification can be achieved.

9.3 Extraction Of Spectral Features

A fingerprint reflects the pattern of individual epidermal ridges and furrows that appear on the surface of a finger. Its uniqueness depends upon the overall pattern of ridges and furrows and the local ridge anomalies known as minutiae. Out of more than 150 minutiae that were identified, the ridge endings and ridge bifurcations are the most common (Refer to Figure 9.2). Normally, the

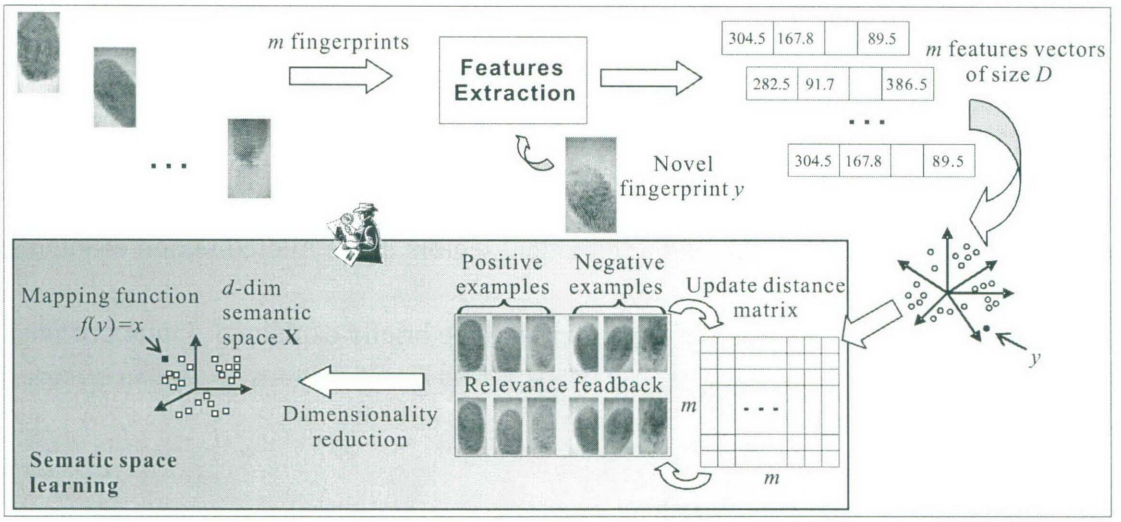


Figure 9.1: The Learning Framework showing the major components, *Features Extraction* and *Semantic Space Learning*. Semantic Space Learning is in turn comprised by Relevance feedback and Dimensionality reduction modules.

pattern of flow of ridges and furrows on a finger varies continuously so that it can be considered an oriented texture field. Often, textured images (including fingerprints) contain a small range of spatial frequencies. Textures that are mutually distinct differ considerably in their dominant frequencies. By decomposing the texture field into a number of spatial frequency and orientation channels, textured regions possessing different spatial frequency, orientation, or phase can be easily discriminated.

In this research, we made use of a features extraction algorithm proposed in [59]. It employs both global and local ridge characteristics to construct a short and fixed length vector for every fingerprint called FingerCode. Each FingerCode is comprised of an ordered enumeration of the features extracted from the local ridge characteristics contained in each sub-image or sector specified by a tessellation. Thus, each sector captures the local information and the ordered enumeration of the tessellation captures the invariant global relationships among these local patterns. Finally, Gabor filters are applied to decompose the local discriminatory characteristics in each sector into bi-orthogonal components based on their spatial frequencies. In summary, this algorithm consists of four processing stages (refer to Figure 9.2):

1. locate a reference point in the fingerprint image;

2. extract and tessellate the region of interest into sectors around the reference point;

3. spectral decomposition in eight different directions using a bank of Gabor filters;

4. compute the FingerCode based on individual sectors in the filtered region of interest.

In the following sub-sections, each of these steps will be briefly explained. Interested readers are referred to the original paper [59] for more details.

9.3.1 Locate Reference Point

In [59], the reference point (x_r, y_r) is defined as the location of maximum curvature along the concave ridges of the fingerprint image. The method used was based on multiple resolution analysis of the orientation field. For a fingerprint image, the orientation field $\mathcal{O}$ is defined as an M by N image where $\mathcal{O}(i, j)$ indicates the local ridge orientation at pixel (i, j). Instead of having one value at every pixel, the local ridge orientation is often specified for a block of size w by w.

There are largely six steps in this processing stage:

1. Estimate the orientation field $\mathcal{O}$ using a block of size w by w (w =15, 10, or 5 pixels could be used)[1].

2. Construct a smoothed orientation field $\mathcal{O}'$ by convolving $\mathcal{O}$ with a two-dimensional low-pass filter W with unit integral, whose size is w_ϕ by w_ϕ as follow:

$$\mathcal{O}'(i, j) = \frac{1}{2} \arctan\left(\frac{\Phi'_y(i, j)}{\Phi'_x(i, j)}\right), \tag{9.1}$$

where

$$\Phi'_x(i, j) = \sum_{u=-w_\Phi/2}^{w_\Phi/2} \sum_{v=-w_\Phi/2}^{w_\Phi/2} W(u, v) \cdot \cos((2\mathcal{O}(i - uw, j - vw))), \tag{9.2}$$

[1]Refer to [59] p.850 for the orientation field estimation algorithm based on least mean square.

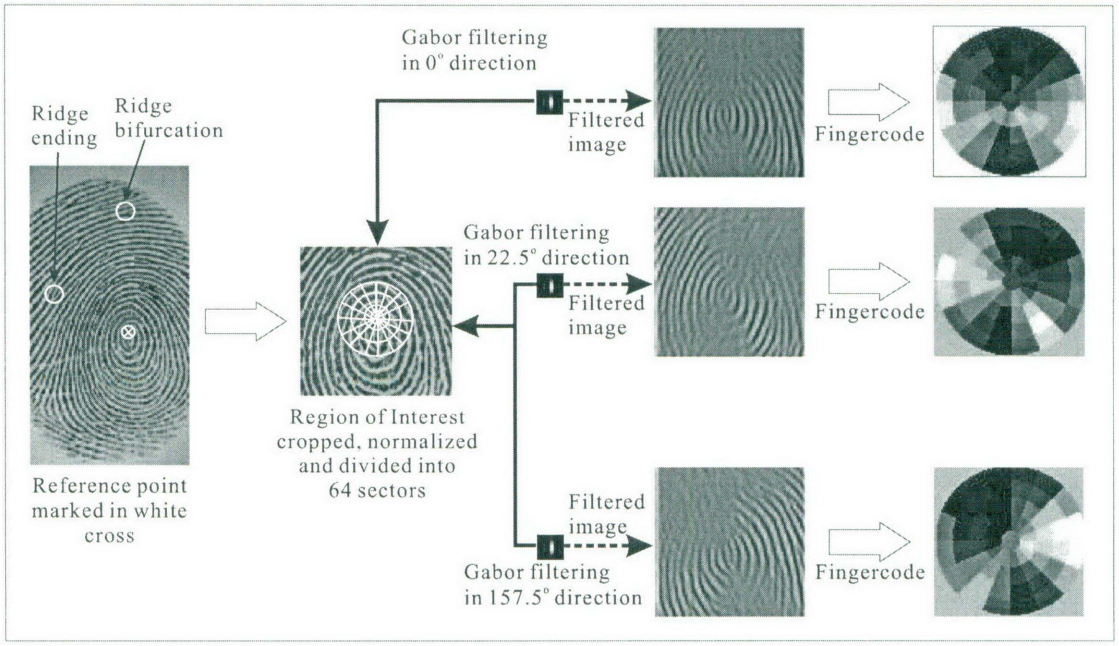


Figure 9.2: Extraction of Spectral Features by multispectral decomposition using a bank of Gabor filters aligned in eight different directions including 0^o, 22.5^o, 45^o, 67.5^o, 90^o, 112.5^o, 135^o, 157.5^o. *All images are shown in half their original sizes.*

$$\Phi'_y(i,j) = \sum_{u=-w_\Phi/2}^{w_\Phi/2} \sum_{v=-w_\Phi/2}^{w_\Phi/2} W(u,v) \cdot \sin((2\mathcal{O}(i-uw, j-vw))). \tag{9.3}$$

3. Using only the sine component of $\mathcal{O}'$, construct an image S as:

$$S(i,j) = \sin((\mathcal{O}'(i,j))). \tag{9.4}$$

4. Next, another image L is constructed from S by taking the difference between the sums of pixel intensities computed in two geometric regions[2] R_1 and R_2 in the neighborhood of each pixel (i,j) as:

$$L(i,j) = \sum_{R_1} S(i,j) - \sum_{R_2} S(i,j). \tag{9.5}$$

5. Determine the pixel in L that has the maximum value, and assign its coordinate as the reference point.

[2]Refer to [59] p.851 (Figure 7) for a diagram of these geometric regions.

6. Repeat steps 1-5 using a different block size w' by w' $(w' < w)$ for a pre-defined number of iterations until a stable reference point is found.

9.3.2 Extract and Tessellate Region of Interest

Once the reference point (x_r, y_r) on a fingerprint image is located, the region of interest can be extracted and tessellated. In [59], the region of interest is defined as the set of all sectors S_i tessellated in terms of two parameters r and θ as:

$$S_i = \{(x, y)|b(T_i + 1) \leq r \leq b(T_i + 2),\ \theta \leq \theta < \theta_{i+1},\ 1 \leq x \leq N,\ 1 \leq y \leq M\}, \tag{9.6}$$

where

$$T_i = i/k, \tag{9.7}$$

$$\theta_i = (i \bmod k) \times (2\pi/k), \tag{9.8}$$

$$r = \sqrt{(x - x_r)^2 + (y - y_r)^2}, \tag{9.9}$$

$$\theta = \arctan((y - y_r)/(x - x_r)). \tag{9.10}$$

Here, b and k denote the width and the number of sectors of each band, respectively. Also, i takes value from 0 to $(B \times k - 1)$, where B is the number of concentric bands encircling the reference point (x_r, y_r). Note that the values of b, k and B are dependent on both the size and resolution of the fingerprint images that one is working with. In our own experiments, $b = 20$, $k = 16$, and $B = 4$ were used.

9.3.3 Spectral Decomposition using Gabor Filters

A bank of Gabor filters, oriented in eight different directions, is used in filtering the region of interest in order to obtain the spatial frequency along a particular orientation. Prior to filtering, the region of interest is normalized in order to minimize the errors introduced by noises and other

deformations occurred during fingerprint capture[3]. Each of the Gabor filters used in [59] is even symmetric and has the following general form:

$$G(x, y; f, \theta) = \exp(\left\{ \frac{-1}{2} \left[\frac{x'^2}{\delta_{x'}^2} + \frac{y'^2}{\delta_{y'}^2} \right] \right\}) \cos((2\pi f x')) \tag{9.11}$$

where

$$x' = x \sin(\theta) + y \cos(\theta), \tag{9.12}$$

$$y' = x \cos(\theta) - y \sin(\theta). \tag{9.13}$$

Here, f denotes the frequency of the sine wave along one of the eight directions θ (0^o, 22.5^o, 45^o, 67.5^o, 90^o, 112.5^o, 135^o, and 157.5^o), measured from the x-axis. $\delta_{x'}$ and $\delta_{y'}$ are the widths of the Gaussian envelope along the x' and y' axes, respectively. In actual implementation, the filter mask is set to size 33 by 33 pixels.

9.3.4 Compute the FingerCode

Every sector S_i of the eight filtered images provides one value for the $k \times B \times 8$ dimensional FingerCode. This value, $V_{i\theta}$, is the *average absolute deviation* (AAD) from the mean of the particular filtered image that S_i belongs to, and is defined as:

$$V_{i\theta} = \frac{1}{n_i} \left(\sum_{n_i} |F_{i\theta}(x, y) - P_{i\theta}| \right). \tag{9.14}$$

where $F_{i\theta}(x, y)$ is the θ-direction filtered image for sector S_i.

9.4 Learning Examiner-Centric Semantic Space

In this section, we will first discuss how to incorporate an examiner's subjective judgment through relevance feedback by iteratively updating the distance matrix. Then, we will explain the dimensionality reduction process by which both the semantic space and the mapping function can be obtained so that a novel fingerprint can be embedded in the semantic space directly for classification.

[3] Refer to [59] p.851 for the method of normalizing the region of interest prior to filtering.

9.4.1 Relevance Feedback

Relevance feedback is closely linked with information retrieval in the literature. The goal is to exploit user's subjective judgment to improve the quality of retrieval often measured in terms of the precision and recall metrics. As data other than texts such as images and multimedia proliferate, studies on relevance feedback for Content-based Image Retrieval (CBIR) systems grew. In this research, we exploit relevance feedback to incorporate a fingerprint examiner's subjective judgment in the formation of a *personalized* and *persistent* semantic space for classification. The idea has similarities to long-term learning in image retrieval proposed in [54, 55], but differ both in the application and the method of learning. In terms of application, we are concerned with classification of novel fingerprints rather than retrieval of similar images, as in the case of [54, 55]. The difference in the method of learning will be explained in further details in the next sub-section on dimensionality reduction.

The relevance feedback procedure can be summarized as follows. Inputs are the m by m distance matrix and a parameter k indicating the number of fingerprints in the feedback. By either accepting an image picked by the examiner or selecting an image q randomly from the database, it returns a subset of images (excluding q) that are based on smallest distances or largest similarities. The examiner then indicates as positive or negative examples those images that are judged similar or dissimilar based on prior experiences. Based on the set of positive and negative examples, their entries in the distance matrix will be updated. In this work, we also exploit the apriori *class* information by adjusting entries for fingerprints that belong to the same classes as those of the positive and negative examples which was not done in [54, 55]. The relevance feedback procedure repeats until the examiner is satisfied. The output is a modified distance matrix that has *learnt* the subjective judgment of the examiner. The pseudo code of the overall procedure is given in Table 9.1.

9.4.2 Dimensionality Reduction

As mentioned above, the construction of an examiner's semantic space and the mapping function for embedding novel fingerprints is being cast as a dimensionality reduction problem. In

INPUT: $m \times m$ distance matrix, k
OUTPUT: updated $m \times m$ distance matrix
UPDATE FUNCTION: $\sigma(d) = \frac{1}{1+\exp((\log(e^d-1)\pm\delta))}$ where $+\delta/-\delta$ for positive/negative images respectively.

```
bool satisfied = FALSE;
while (not satisfied)
    if (examiner selects an image)
        q = selected image;
    else
        generate a random number;
        select an image q from the database;
    end
    display the nearest k images to image q based on smallest distances or largest similarities
    (excluding q);
    the examiner selects both positive and negative examples;
    for (positive images i that are similar to q)
        update their entries in the distance matrix by d_iq = d_qi = σ(d_iq);
    end
    for (negative images i' that are not similar to q)
        update their entries in the distance matrix by d_i'q = d_qi' = σ(d_i'q);
    end
        if (the examiner is satisfied) satisfied = TRUE; end
end
return updated distance matrix;
```

Table 9.1: Pseudo code of relevance feedback procedure.

He et al. (2004) [55]	**Our method**
Steps: 1. Update distance matrix by relevance feedback; 2. Construct neighborhood graph by kNN; 3. Apply Laplacian Eigenmaps (LE); 4. LE semantic space without a mapping function; 5. Using FingerCode and LE embeddings, train a Radial Basis Function Neural Network to obtain the mapping function; 6. Use mapping function to embed novel images in approximated LE semantic space for retrieval.	Steps: 1. Update distance matrix by relevance feedback; 2. Apply BCTKE; 3. BCTKE semantic space with a mapping function; 4. Use mapping function to embed novel fingerprints in TKE semantic space for classification.

Table 9.2: Comparison of He et al. (2004) [55]'s dimensionality reduction procedure and the one adopted in this research.

essence, we seek a lower-dimensional representation of the original high-dimensional features space that preserves (and reflects) the examiner's subjective judgment as close as possible. Future classification of novel fingerprints can thus be performed in the newly constructed semantic space which is expected to be more efficient and more amiable to visualization.

Similar to relevance feedback, this idea is inspired by the long term learning model proposed in [55][4]. However, there are several differences which will become clear in the following comparison.

One might notice in the above comparison that [55]'s dimensionality reduction procedure requires two stages (Steps 2-4 and Step 5) to arrive at the actual semantic space with the mapping function. In contrast, ours is a more direct process that requires only one stage (Steps 2-3). Also, the semantic space constructed in [55] is *approximate*, which is obtained by a previous training step using a neural network. The semantic space constructed by our method is not an approximated one, which is obtained directly through BCTKE. Table 9.2 shows the procedures of our method and the one proposed in [55].

[4]Refer to [55] Section 3.2 and Section 3.3 for a detailed description of their dimensionality reduction process.

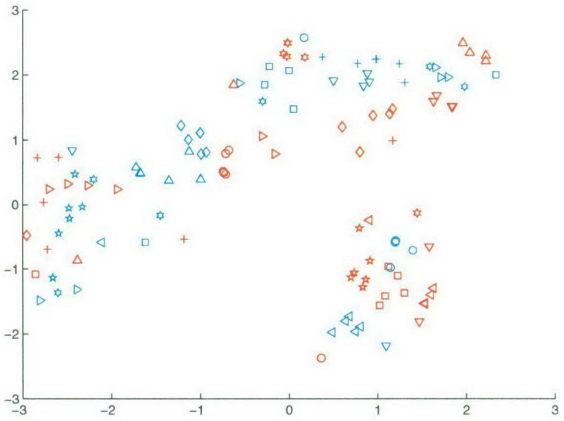
9.5 Experimental Evaluation

A set of experiments was conducted on a subset of the Fingerprint Verification Competition 2002 (FVC2002)'s Db2 database. The fingerprints were captured by an optical sensor. The image size is 296×560 pixels while the resolution is 569 dpi. The Db2 database is divided into two sets, Db2_a and Db2_b. The width and depth of the Db2_a set are 100 and 8 respectively, meaning there are 100 fingers each having 8 impressions. Similarly, the width and depth of the Db2_b set are 10 and 8 respectively. Altogether, Db2 has 880 fingerprint images.

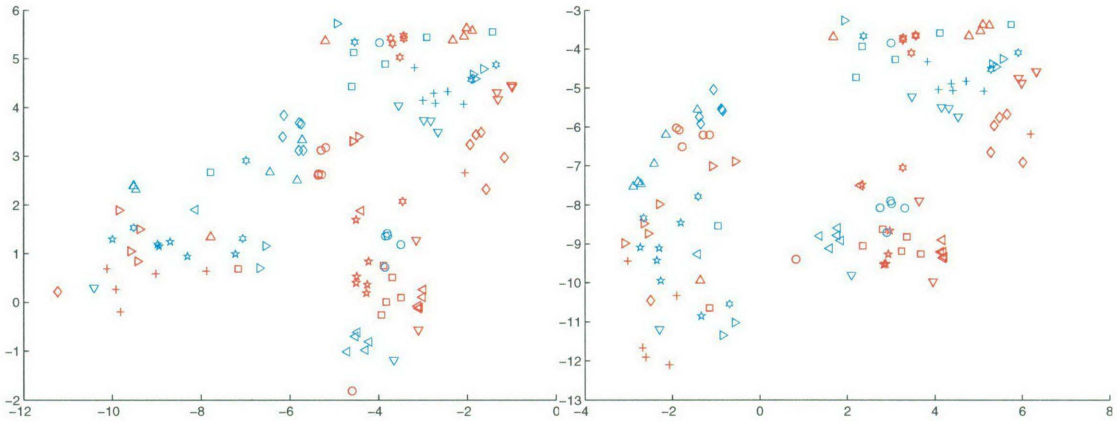
For our experiments, we randomly select twenty fingers from Db2. Six of the eight impressions in each finger are taken to form the initial database, totaling 120 fingerprint images. In other words, the dimension of the initial distance matrix is 120×120. The remaining two impressions from each finger are grouped to form the set of novel fingerprints to test the quality of classification. Although the database size here is limited, it is adequate to illustrate the benefits of the proposed learning framework. In production setup, we would expect a larger database for the fingerprint examiners to work with.

In Figures 9.3(a)-(e), the semantic space before relevance feedback, after 50 times, after 100 times, after 150 times and after 200 times are shown respectively. k is set to 10 in order to avoid degrading the quality of feedback. It is clear that as relevance feedback repeats the clusters of fingerprints are becoming more and more compact. This is consistent with the improvement in quality of classification as subjective judgment is increasingly incorporated into the distance matrix.

In Table 9.3, we compare the quality of classifying the set of 40 test fingerprints between the baseline (that is simply using the initial distances), semantic space in 2-dimension before and after certain number of relevance feedback. The metric used is the k-NN classification errors. While the initial semantic space has worse classification quality than the baseline (due to information loss in DR), the quality improves as the examiner's subjective judgment is increasingly incorporated.

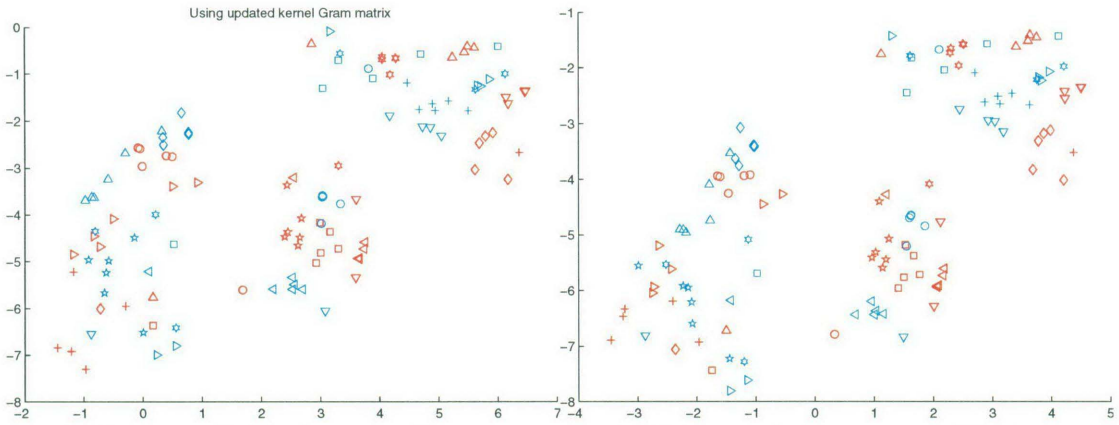


(a) Semantic Space in 2-dim without relevance feedback



(b) Relevance feedback repeats 50 times

(c) Relevance feedback repeats 100 times



(d) Relevance feedback repeats 150 times

(e) Relevance feedback repeats 200 times

Figure 9.3: Semantic space embedded in two dimensions by TKE before and after 50, 100, 150 and 200 relevance feedback iterations.

Errors	Baseline	No relevance feedback	50 times	100 times	150 times	200 times
1-NN	7	26	17	20	14	10
2-NN	6	25	17	21	14	9
3-NN	10	28	17	19	16	10
4-NN	12	28	17	20	19	13
5-NN	19	29	17	20	20	15
6-NN	30	34	28	25	24	22

Table 9.3: Comparing classification quality between baseline, semantic space in 2-dim before and after relevance feedback.

9.6 Summary

By exploiting relevance feedback from fingerprint examiners, a personalized and persistent semantic space over the database of fingerprints for each examiner can be incrementally learned. The fingerprint features that induce the initial features space from which semantic spaces are being learned were obtained by multispectral decomposition of fingerprints (taken as texture images) using a bank of Gabor filters. In this learning framework, we apply the BCTKE to learn both the semantic space and the mapping function for classifying novel fingerprints. Experimental evaluation verifies the potential of this learning framework for examiner-centric fingerprint classification.