國　立　交　通　大　學

資訊工程學系

博 士 論 文

複合式高斯類神經網路之研究

The Study of Mixture Gaussian Neural Networks

研 究 生：徐永煜

指導教授：傅心家 教授

中華民國 九 十 三 年 七 月

複合式高斯類神經網路之研究

The Study of Mixture Gaussian Neural Networks

研 究 生 : 徐永煜　　Student: Yeong-Yuh Xu
指導教授: 傅心家 教授　　Advisor: Prof. Hsin-Chia Fu

國 立 交 通 大 學

資 訊 工 程 學 系

博 士 論 文

A Dissertation
Submitted to Department of Computer Science and Information Engineering
College of Electrical Engineering and Computer Science
National Chiao Tung University
in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy
in
Computer Science and Information Engineering

July 2004
Hsinchu, Taiwan, Republic of China.

中 華 民 國 九十三 年 七 月

複合式高斯類神經網路之研究

研究生:徐永煜　　指導教授: 傅心家 教授

國立交通大學資訊工程學系

摘　　要

本論文在探討複合式高斯類神經網路模組於圖形識別的應用。基於一個類別一個網路 (One class in One Network) 的神經網路架構, 本論文提出了自成長式機率決策神經網路 (Self-growing Probability Decision based Neural Network (SPDNN)) 與 Iteratively Supervised Learning and Unsupervised Growing (ISLUG) 的訓練策略來自動的根據資料的複雜程度 調整 SPDNN 的高斯個數 (Gaussian cluster number)、位置與大小 (mean and covariance matrix) 進而增進辨識的正確率。本論文更進一步提出了一般化機率決 策神經網路 (Generalized Probability Decision based Neural Network (GPDNN)) 來解決當網路的輸入從數值一般化成機率分布的情況。爲了驗證所提的類神經網路模組, 本論文並將所提的類神經網路應用於手寫字辨識與以影像內容爲基礎的影像搜尋的問題上。

在手寫字辨識的應用上, 本論文建立了以 SPDNN 爲基礎的手寫字辨識器。基於 CCL/HCCR1的資料庫, SPDNN手寫字辨識器可以達到86.12%的辨識正確率。此辨識結果可以與 Li and Yu (88.65%) [1] and Tseng *et al* (88.55%) [2]所發表的 正確率相匹配, 但是SPDNN 文字辨識器只用了92維的特徵, 明顯比 [1] 的400維的特徵 與 [2] 的256維的特徵少。除此之外, 本論文也建立了個人手寫字辨識系統。根據 ISLUG 的訓練策略,SPDNN 個人手寫字辨識系統能漸漸地認識個人獨特的手寫字特徵, 並 進而提高對個人手寫字辨識的正確性。實驗結果顯示個人手寫字辨識的正確性可在 SPDNN的學習過程中從44.09%提高到90.03%。在以影像內容爲基礎的影像搜尋上, 本論文基於 GPDNN 建立了一 個影像內容搜尋系統 (http://140.113.216.78/imagequerysystem), 並提 出了兩個新的概念: 視覺關鍵字 (Visual Keyword) 與視覺關鍵字串 (Visual String) 來作爲影像的索引。視覺關鍵字主要是用來描述影像中同質區塊 (homogenous region) 的視覺特性, 如顏色、紋理 與形狀特徵。而視覺關鍵字串則用來描述區塊間於影像上的位置關係。基於 Corel 的資料庫, GPDNN影像內容搜尋器在視覺關鍵字的搜尋上可以達到49.4%的正確率。其結果可與[3] 所提的 IRM 方法 (46.8%) 與[4] 所提的UFM 方法 (47.7%) 相匹配; 另一方面, 當考慮區塊間於影像上的位置關係時, 用視覺關鍵字串搜尋確實 能很明顯的將所感興趣的影像排序在影像序列的前端。由上所述的應用可知, 所提出的複合式高斯類神經 網路模組確實能夠根據的資料特性, 適切的學習並辨識這些資料。

The Study of Mixture Gaussian Neural Networks

Student:Yeong-Yuh Xu Advisor:Prof. Hsin-Chia Fu

Department of Computer Science and Information Engineering
National Chiao Tung University

Abstract

In this dissertation, the mixture Gaussian Neural networks are proposed for pattern recognition. *Self-growing Probabilistic Decision based Neural Network* (SPDNN) is proposed to classify the numerical data. The ISLUG training scheme is introduced to tune the SPDNN parameters to improve the classification accuracy. Furthermore, a generalized version of SPDNN, called *Generalized Probabilistic Decision based Neural Network* (GPDNN), is proposed to handle the general case that the data are in the form of the distributions instead of the numerical quantities. In order to verify the accuracy of the proposed SPDNN and GPDNN, the applications of handwritten character recognition and content-based image retrieval are involved.

An SPDNN-based Handwritten Chinese major hybrid character recognition system is developed. All the major processing modules, including pre-processing and feature selection modules, a coarse classifier, a character recognizer, and a personal adaptation module, are implemented. Based on the CCL/HCCR1 database, the SPDNN character recognizer achieved 86.12% recognition accuracy, which is comparable to the reports by Li and Yu (88.65%) [1] and Tseng *et al* (88.55%) [2], but uses only 92 features compared to the huge number of character features used in [1] (400 features) and [2] (256 features).

In the content-based image retrieval application, a *Neural Networks based Image Retrieval System* (NNIRS) is developed based on GPDNN and is implemented at "http:// 140.113.216.78/ imagequerysystem". Two novel image representation concepts for CBIR are proposed: (1) the *visual keyword* and (2) the *visual string*. The *visual keyword* describes the visual characteristics (color, texture, and spatial features) of a homogenous region, while the *visual string* represents the spatial relation of the regions in an image. The experiment results show the follows: (1) the retrieving performance by *visual keyword* method is 49.4%, which is comparable the 46.8% of IRM method [3] and 47.7% of the UFM method [4], and (2) query by *visual string* can significantly improve the hit rate of finding the interested images while the spatial relation of *visual keywords* is concerned.

誌　謝

本論文得以完成, 首先我要感謝 傅心家教授多年來的悉心指導, 並於研究過程中 不吝其煩的給予精闢的指正與獨到的見解。其次要感謝我的論文指導委員 陳正教授及林正中教授對我的論文方向及內容的指導與建議。另外, 我還要感謝 劉長遠教授、李漢銘教授、張隆紋教授、陶金旭教授等博士學位考試 委員對本論文的費心審查與完成此論文所提之寶貴看法。在此向諸位 教授致上最崇高的敬意與謝意。

其次, 我要感謝在完成論文這段過程中, 實驗室的成員所給予 的協助與幫忙。尤其要謝謝莊舜清學長在百忙中抽出時間, 與我討論 並幫我完成測試程式, 使得系統雛形得以完成。

最後, 我要感謝我的家人, 尤其是我的 父母在我完成學業這段期間, 提供了一個無後顧之憂的環境, 使我能專心於我的研究。另外, 我還要特別感謝我的妻子淑琳, 謝謝她一直以來的關懷與鼓勵, 包容與支持, 使這篇論文得以順利完成。

徐永煜

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The term pattern recognition encompasses a wide range of information processing problems of great practical significance, from speech recognition and the classification of handwritten characters to fault detection in machinery and medical diagnosis [5]. Watanabe [6] defines a pattern "as opposite of a chaos; it is an entity, vaguely defined, that could be given a name." For example, a pattern could be a handwritten character, a nature image, a speech signal, or a human face. In a pattern recognition problem, a pattern or object is given some measurements (the "features") and is assigned with a class membership ("label"). Given a set of training patterns, a pattern recognition system is trained to establish decision boundaries in the feature space which separate patterns belonging to different classes. Two excellent survey papers of pattern recognition can be found in [7] and [8].

Since uncertainty is the nature problem of measurements, the statistical approach has been studied to characterize the uncertainty with a probability distribution. In the statistical approach, the decision boundaries are determined by the probability distributions of the patterns belonging to each class,

which must be specified or learned [9] [10]. A good review of statistical pattern recognition can be found in [11].

More recently, neural network techniques and methods imported from statistical learning theory have been receiving increasing attention [5] [12]. A neural network is a parallel, distributed, adaptive system that has the capability to learn, accumulate knowledge, and apply this knowledge to new situations. The relationship between neural networks and statistical pattern recognition have been discussed in [13] and [14].

Probabilistic decision-based neural networks (PDBNN) [15] is an example of combining the statistical and neural networks approaches for pattern recognition. PDBNN is proposed to attack face and other biometrics recognition systems. The modular structure in PDBNN devotes one of its subnets to the representation of a particular person's face images. In each subnet, the discriminate function of PDBNN is in a form of mixture Gaussian distribution. This yields extremely low false acceptance and rejection rates for the face recognition systems. Nonetheless, the number of Gaussian clusters of the mixture Gaussian distribution is predeterminate. With the incorrect assumption of the number of Gaussian clusters, the recognition or classification accuracy may be poor.

In some pattern recognition problems, the features of an object are in the form of the distributions instead of the numerical quantities. For example, the color feature of an image region may be defined as statistical information about the variation of the color across the region and described by a Gaussian distribution with mean vector and covariance matrix. It is an important issue to measure the dissimilarity between two distributions to classify or recognize the object. However, only few works [16, 17, 18, 19] have been devoted to studying

the dissimilarity measurement between two distributions. Besides, none of these studies showed how to build a class model from a set of distributions.

In this thesis, a new PDBNN, called *Self-growing Probabilistic Decision-based Neural Network* (SPDNN) is proposed. For different classes, the discriminate function of SPDNN is in a form of flexible number of mixture of Gaussian distributions. Furthermore, to handle the general case that each feature is in the form of the distribution, I proposed a generalized version of SPDNN, called *Generalized Probabilistic decision based Neural Network* (GPDNN). In order to verify the accuracy of the proposed SPDNN and GPDNN, the applications of handwritten character recognition and content-based image retrieval are involved. The following section summarizes the contributions of this thesis.

1.1 Summary of Contributions

The goal of this thesis is to study the various problems and develop the neural networks for statistical pattern recognition. Based on the proposed neural networks, a handwritten character recognition system and a content-based image retrieval system are built. The main contributions of the research are summarized below:

Mixture Gaussian Neural Networks (Chapters 2 and 3)

Self-growing Probabilistic Decision based Neural Network (SPDNN) is proposed to classify the numerical data. The ISLUG training scheme is introduced to tune the SPDNN parameters to improve the classification accuracy. Furthermore, a

generalized version of SPDNN, called *Generalized Probabilistic decision based Neural Network* (GPDNN), is proposed to handle the general case that the data are in the form of the distributions instead of the numerical quantities. Two features of SPDNN/ GPDNN make it suitable for implementing in pattern recognition systems. These features are:

- Architecture feature: SPDNN/ GPDNN adapts the modular OCON (One class in One Network) structure, which devotes one of its subnets to representation of a particular class. This kind of structure is beneficial not only for training and recognition performance, but also for hardware implementation.

    1. *The system is easy to train and maintain.* Training an SPDNN/ GPDNN-based pattern recognition system is relatively straightforward. For example, by adding or modifying one or a few clusters (cf. Section 2.3.2) in a subnet of SPDNN/ GPDNN, the character recognition system can adapt a person's style. A centralized system, in contrast, would have to include global updating.

    2. *A distributed computing principle is adopted.* With the large number of class, the computing hardware requirements for recognition system are greater. Due to its modular architecture, an SPDNN/ GPDNN-based recognition system is relatively easy to implement on parallel computers.

- Performance Feature: The discriminant function of SPDNN/GDDNN is in a form of probability density. This yields an effective adaptation process and very accurate recognition.

Handwritten Character Recognition (Chapters 4)

An SPDNN-based Handwritten Chinese major hybrid character recognition system is developed. All the major processing modules, including pre-processing and feature selection modules, a coarse classifier, a character recognizer, and an personal adaptation module, are implemented.

Content-based Image Retrieval (Chapters 5)

A GPDNN-based image retrieval system is developed. The *visual keyword* and the *visual string* are proposed to illustrate the visual key characteristics and their spatial relation of an image. The *visual keyword* is in the form of the mixture Gaussian distribution. Instead of annotating each region in an image by keywords, a region is represented by a *visual keyword*. The key issues for the *visual keyword* representation are as follows: (1) the precise segmentation or skillful sketch of the region is no longer needed, and (2) the approximating a region by mixture Gaussian distributions allows the searching for its similar regions more flexible and robust. In addition, by using the *visual string* to index an image, computing the similarity or dissimilarity between the desired and a tested images becomes as simple as a string matching task.

1.2 Thesis Organization

This thesis is divided into six chapters. In Chapter 2, the architecture and learning scheme of the proposed SPDNN are introduced. By generalizing from

SPDNN, GPDNN is proposed and described in Chapter 3. In Chapter 4, the handwritten character recognition is study, and an SPDNN-based Handwritten Chinese major hybrid character recognition system is developed. Some experiments are involved to verify the proposed handwritten character recognition system. Chapter 5 provides the application of the content-based image retrieval for GPDNN. This chapter introduces the proposed *visual keyword* and *visual string*. Based on these image indexes, a GPDNN-based image retrieval system is developed and evaluated on the COREL Gallery $1,000,000$, containing about 60000 general purpose color images. Chapter 6 provides the conclusions and some research topics in the future.

Chapter 2

Self-growing Probabilistic Decision based Neural Network

Self-growing Probabilistic Decision-based Neural Network (SPDNN) is a probabilistic variant of decision-based modular neural network [20] for classification. One subnet of an SPDNN is designed to represent one object class. There are two properties of the SPDNN learning rules. The first one is the decision-based learning rules. Based on the teacher information which only tells the correctness of the classification for each training pattern, SPDNN performs a distributed and localized updating rules. The updating rule applies *reinforced learning* to a subnet corresponding to the correct class and *antireinforced learning* to the (unduly) winning subnets.

The second property is the *Iteratively Supervised Learning and Unsupervised Growing* (ISLUG). There are two learning phases in this scheme: after each subnet is initialized with one cluster (see Section 2.3.1) or is self-grown with a new cluster (Section 2.3.2), the system enters the *Supervised Learning* (SL) phase. In the SL phase, teacher information is used to reinforce or antireinforce the decision boundaries obtained during the initialization or self-growing

stages. When the supervised training progress becomes very slow or is trapped in a paralysis state, yet the classification or recognition accuracy is not at a satisfied level, the training enters the *Unsupervised Growing* (UG) phase. In the UG phase, an SPDNN creates a new cluster in a subnet according to the proposed *self-growing* rule. Thereafter, the training enters the *Supervised Learning* phase again. The ISLUG learning procedure terminates when the training accuracy reaches a predefined satisfaction level. The detailed description of SPDNN is given in the following sections.

2.1 Overview

Similar to PDBNN, SPDNN devotes one of its subnetworks to represent one object class with a mixture Gaussian distribution. For a K-category classification problem, SPDNN contains K subnets as shown in Fig. 2.1. In each subnet, several φ components are involved to compute the Gaussian clusters of a mixture Gaussian distribution to represent the corresponding class. Fig. 2.2 shows an example of a subnet with five φ components to represent a particular class, where 'o' stands for data belonging to the class. In Fig. 2.2, each ellipse corresponds to one of the φ components in the subnet to show one of the Gaussian clusters of a mixture Gaussian distribution. The following sections details the SPDNN computation.

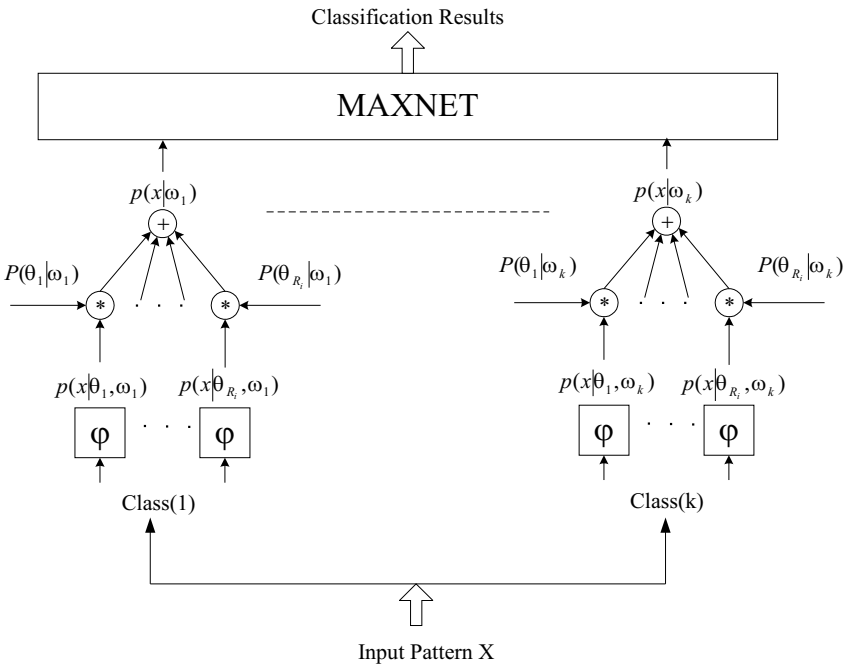


Figure 2.1: Schematic diagram of a *k-class* SPDNN classifier.

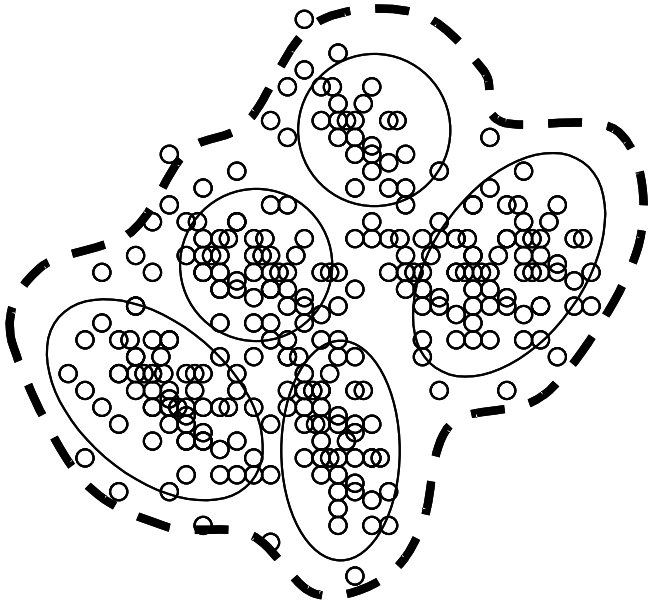


Figure 2.2: An example of data representation via a subnet in SPDNN. 'o' stands for pattern belonging to the target class. Each ellipse corresponds to one of the Gaussian clusters, and a mixture Gaussian distribution with five Gaussian clusters is used to approximate the data distribution of the class.

2.2 Discriminant Functions of SPDNN

One of the major differences between PDBNN [15] and SPDNN is that SPDNN extends the fixed number of clusters in PDBNN to flexible number of clusters. That is, the subnet discriminant functions of SPDNN are designed to model the log-likelihood functions of different complex distributions. Thus, *reinforced* or *antireinforced* learning is applied to *all* the subnets of the global winner and the supposed (i.e., the correct) winner with a weighting distribution proportional to the degree of possible involvement (measured by the likelihood) by each subnet. Given a set of i.i.d. patterns $\mathbf{X}^+ = \{\mathbf{x}(t);\ t = 1, 2, \cdots, N\}$, we assume that the likelihood function $p(\mathbf{x}(t) \mid \omega_i)$ for class ω_i is a mixture of Gaussian distributions.

Define $p(\mathbf{x}(t) \mid \omega_i, \Theta_{r_i})$ to be one of the Gaussian distributions which comprise $p(\mathbf{x}(t) \mid \omega_i)$, where Θ_{r_i} represents the parameter set $\{\mu_{r_i}, \mathbf{\Sigma}_{r_i}\}$ for a cluster r_i in subnet i.

$$p(\mathbf{x}(t) \mid \omega_i) = \sum_{r_i=1}^{R_i} P(\Theta_{r_i} \mid \omega_i) p(\mathbf{x}(t) \mid \omega_i, \Theta_{r_i}), \tag{2.2.1}$$

where $P(\Theta_{r_i} \mid \omega_i)$ denotes the prior probability of the cluster r_i. By definition, $\sum_{r_i=1}^{R_i} P(\Theta_{r_i} \mid \omega_i)$=1, where R_i is the number of clusters in ω_i.

The discriminate function of the multiclass SPDNN models the log-likelihood function

$$\begin{aligned} \phi(\mathbf{x}(t), \mathbf{w}_i) &= log\ p(\mathbf{x}(t) \mid \omega_i) \\ &= log \left[\sum_{r_i=1}^{R_i} P(\Theta_{r_i} \mid \omega_i) p(\mathbf{x}(t) \mid \omega_i, \Theta_{r_i}) \right], \end{aligned} \tag{2.2.2}$$

where $\mathbf{w}_i = \{\mu_{r_i}, \mathbf{\Sigma}_{r_i}, P(\Theta_{r_i} \mid \omega_i), T_i\}$, and T_i is the output threshold of the subnet i.

In most general formulation, the basis function of a cluster should be able to approximate the Gaussian distribution with full rank covariance matrix, i.e., $\phi(\mathbf{x}, \omega_i) = -\frac{1}{2}\mathbf{x}^T\mathbf{\Sigma}_{r_i}^{-1}\mathbf{x}$, where $\mathbf{\Sigma}_{r_i}$ is the covariance matrix. However, for those applications which deal with high-dimension data but finite number of training patterns, the training performance and storage space discourage such matrix modelling. A natural simplifying assumption is to assume uncorrelated features of unequal importance. That is, suppose that $p(\mathbf{x}(t) \mid \omega_i, \Theta_{r_i})$ is a D-dimensional Gaussian distribution with uncorrelated features

$$p(\mathbf{x}(t) \mid \omega_i, \Theta_{r_i}) = \frac{1}{(2\pi)^{\frac{D}{2}}|\mathbf{\Sigma}_{r_i}|^{\frac{1}{2}}} \cdot \exp\left[-\frac{1}{2}\frac{(\mathbf{x}(t) - \mu_{r_i})^T(\mathbf{x}(t) - \mu_{r_i})}{\mathbf{\Sigma}_{r_i}}\right] \tag{2.2.3}$$

where $\mathbf{x}(t) = [x_1(t), x_2(t), \cdots, x_D(t)]^T$ is an input pattern, $\mu_{r_i} = [\mu_{r_i1}, \mu_{r_i2}, \cdots, \mu_{r_iD}]^T$ is the mean vector, and diagonal matrix $\mathbf{\Sigma}_{r_i} = diag[\sigma_{r_i1}^2, \sigma_{r_i2}^2, \cdots, \sigma_{r_iD}^2]$ is the covariance matrix. As shown in Fig. 2.1, an SPDNN contains K subnets which are used to represent a K-category classification problem. Inside each subnet, an elliptic basis functions (EBFs) is used to serve as the basis function for each cluster r_i:

$$\varphi(\mathbf{x}(t), \omega_i, \Theta_{r_i}) = -\frac{1}{2}\sum_{d=1}^{D}\alpha_{r_id}(x_d(t) - \mu_{r_id})^2 + \theta_{r_i} \tag{2.2.4}$$

where $\theta_{r_i} = -\frac{D}{2}\ln 2\pi + \frac{1}{2}\sum_{d=1}^{D}\ln\alpha_{r_id}$. After passing an exponential activation function, $\exp\{\varphi(\mathbf{x}(t), \omega_i, \Theta_{r_i})\}$ can be viewed as a Gaussian distribution, as described in (2.2.3), except a minor notational change: $\frac{1}{\alpha_{r_id}} = \sigma_{r_id}^2$.

After the outputs of all subnets are obtained, the MAXNET in SPDNN is activated to select the maximum one among these outputs. That is, if the output value of subnet i is the maximum one among the outputs of all subnets in SPDNN, the testing datum $\mathbf{x}(t)$ is classified to class ω_i.

2.3 Learning Rules for SPDNN

The training scheme for SPDNN follows the ISLUG training scheme, which contains the following two phases: *supervised learning* and *unsupervised growing*.

2.3.1 Supervised Learning in each subnet

At the beginning of the first *supervised learning* phase, each subnet is initialized with one cluster since the number of cluster in a subnet of SPDNN can be adjusted in the *unsupervised growing* phase. The values of the parameters (mean and covariance) of a cluster in each subnet are initialized as follows: Suppose that $\mathbf{X}_i^+ = \{\mathbf{x}_i(1), \cdots, \mathbf{x}_i(M_i)\}$ is a set of given training patterns, which correspond to one of the L classes $\{\omega_i, i = 1, \cdots, L\}$, the mean μ_i and covariance matrix $\mathbf{\Sigma}_i$ of the initial cluster in subnet i can be calculated as

$$\mu_i = \frac{1}{M_i} \sum_{m=1}^{M_i} \mathbf{x}_i(m) \tag{2.3.1}$$

$$\mathbf{\Sigma}_i = \frac{1}{M_i - 1} \sum_{m=1}^{M_i} (\mathbf{x}_i(m) - \mu_i)(\mathbf{x}_i(m) - \mu_i)^T. \tag{2.3.2}$$

Then, during the *supervised learning* phase, the decision boundaries of each classes are fine tuned corresponding to the training data. The data adaptive scheme of the supervised learning for the multiclass SPDNN is the extension of the GS learning in [15]. Each class is modelled by a subnet with discriminant functions, $\phi(\mathbf{x}(t), \mathbf{w}_i), i = 1, 2, \cdots, L$. At the beginning of each *supervised learning* phase, use the still-under-training SPDNN to classify all the training patterns $\mathbf{X}_i^+ = \{\mathbf{x}_i(1), \mathbf{x}_i(2), \cdots, \mathbf{x}_i(M_i)\}$ for $i = 1, \cdots, L$. A pattern $\mathbf{x}_i(m)$ is classified to class ω_i if $\phi(\mathbf{x}_i(m), \mathbf{w}_i) > \phi(\mathbf{x}_i(m), \mathbf{w}_k)$, $\forall k \neq i$, and $\phi(\mathbf{x}_i(m), \mathbf{w}_i) \geq T_i$, where T_i is the output threshold for subnet i. According to

the classification results, the training patterns for each class ω_i can be divided into three subsets:

- Correctly classified set:

  $D_1^i = \{\mathbf{x}_i(m); \mathbf{x}_i(m) \in \omega_i, \mathbf{x}_i(m)$ is classified to $\omega_i\ \}$;

- False rejection set:

  $D_2^i = \{\mathbf{x}_i(m); \mathbf{x}_i(m) \in \omega_i, \mathbf{x}_i(m)$ is misclassified to other class $\omega_j\}$;

- False acceptance set:

  $D_3^i = \{\mathbf{x}_i(m); \mathbf{x}_i(m) \notin \omega_i, \mathbf{x}_i(m)$ is misclassified to class $\omega_i\ \}$.

The following reinforced and antireinforced learning rules [20] are applied to the corresponding subnets.

$$\text{R}einforced\ Learning:$$

$$\mathbf{w}_i^{(m+1)} = \mathbf{w}_i^{(m)} + \eta \nabla \phi(\mathbf{x}_i(m), \mathbf{w}_i) \tag{2.3.3}$$

$$\text{A}ntireinforced\ Learning:$$

$$\mathbf{w}_j^{(m+1)} = \mathbf{w}_j^{(m)} - \eta \nabla \phi(\mathbf{x}_i(m), \mathbf{w}_j) \tag{2.3.4}$$

In (2.3.3) and (2.3.4), η is an user defined learning rate $0< \eta \leq 1$, and the gradient vectors $\nabla\phi$ can be computed in a similar manner as proposed in [15].

For the data set D_2^i, reinforced and antireinforced learning will be applied to class ω_i and ω_j, respectively. As for the false acceptance set D_3^i, antireinforced learning will be applied to class ω_i, and reinforced learning will be applied to the class ω_j where $x_i(m)$ belongs to.

Threshold Updating The threshold value T_i of a subnet i in the SPDNN recognizer can also be learned by reinforced or antireinforced learning rules.

2.3.2 Unsupervised Growing of a new cluster

The network enters the *unsupervised growing* phase when the *supervised learning* reaches a saturated (learning state) but unsatisfied (classification accuracy) situation. There are three main aspects for the self-growing rules:

I1: *When should a new cluster be created?*

I2: *Which cluster should be partitioned to create a new cluster?*

I3: *How to initialize the center and the covariance of the new cluster?*

On Issue $I1$, when the whole training set has been presented for a few times, the train status (especially the classification accuracy) remains unchanged or unimproved. An extra cluster is suggested to improve the representation power of SPDNN.

On Issue $I2$, when an extra cluster is needed, a new cluster is suggested to be created from the subnet which caused the most of misclassification during the recent *supervised learning* processes.

On Issue $I3$, when a new cluster is needed, its initial values of the *center* and *covariance* need to be properly determined, otherwise poor classification situation may still exist.

Assume that a training pattern $\mathbf{x}$ corresponding to the class ω_i is presented to an SPDNN classifier: the cluster Θ_i is in the class ω_i, and the cluster Θ_j is in the class ω_j which corresponds to the largest response among the classes other than ω_i. Let o_i and o_j be the output of $\mathbf{x}$ from the class ω_i and the class ω_j,

respectively. According to the retrieving scheme of the proposed SPDNN, if o_j is larger than or equal to o_i, the retrieving result for the training pattern $\mathbf{x}$ must be wrong. As shown in Fig. 2.3(a), clearly the best position for the center of the new cluster should be located at $\mathbf{x}$, i.e., $\mu_0 = \mathbf{x}$, so that the class with new cluster Θ_i' will generate the maximal output o_i for the training character $\mathbf{x}$. To determine the covariance matrix $\mathbf{\Sigma}_0$, first let $\mathbf{\Sigma}_0 = \sigma_0 \mathbf{I}$ and σ_0 be a positive constant (to be determined). As shown in Fig. 2.3(b), if σ of the new cluster Θ_i' is not properly determined, the class with the new cluster will have its largest possible output $o_i(\mathbf{x})$ to be smaller than the output $o_j(\mathbf{x})$ of the class ω_j. In other word, the cluster Θ_i' is *overwhelmed* by the cluster Θ_j. Suppose μ_j and $\mathbf{\Sigma}_j$ are the center and covariance of cluster Θ_j. To prevent the *overwhelming* problem, Fig. 2.3(c) presents a properly initiated new cluster Θ_i'. The following two constraints are suggested for a proper initial value of σ.

$$\begin{aligned} o_j(\mathbf{x}) &= \frac{P(\Theta_j \mid \omega_j)}{(2\pi)^{\frac{D}{2}} |\Sigma_j|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}\frac{(\mathbf{x}-\mu_j)^T(\mathbf{x}-\mu_j)}{\mathbf{\Sigma}_j}\right] + \epsilon_j \\ &< o_i(\mathbf{x}) = \frac{P(\Theta_0 \mid \omega_i)}{(2\pi\sigma)^{\frac{D}{2}}} + \epsilon_i \qquad (2.3.5) \\ o_i(\mu_j) &= \frac{P(\Theta_0 \mid \omega_i)}{(2\pi\sigma)^{\frac{D}{2}}} \exp\left[-\frac{1}{2}\frac{(\mathbf{x}-\mu_j)^T(\mathbf{x}-\mu_j)}{\mathbf{\Sigma}_i}\right] + \epsilon_i' \\ &< o_j(\mu_j) = \frac{P(\Theta_j \mid \omega_j)}{(2\pi)^{\frac{D}{2}} |\Sigma_j|^{\frac{1}{2}}} + \epsilon_j' \qquad (2.3.6) \end{aligned}$$

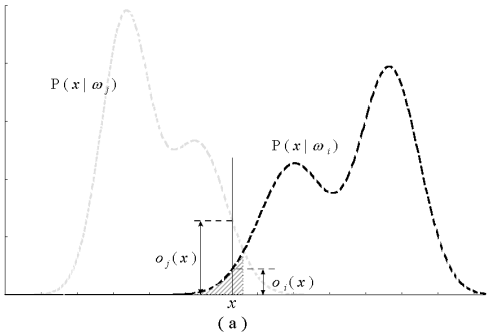
where $P(\Theta_0 \mid \omega_i)$ and $P(\Theta_j \mid \omega_j)$ are the prior probability of the cluster Θ_i' and Θ_j, respectively. ϵ_i and ϵ_j represent the partial output of the classes ω_i and ω_j from the clusters other than Θ_i and Θ_j at $\mathbf{x}$. ϵ_i' and ϵ_j' are the partial output at μ_j of the clusters other than Θ_i and Θ_j. The prior probability $P(\Theta_0 \mid \omega_i)$ of the cluster Θ_i' can be initialized as $(\sigma/\bar{\sigma}_i)P(\Theta_i \mid \omega_i)$, where $\bar{\sigma}_i = (1/R_i)\sum_{r_i=1}^{R_i} \sigma_{r_i}$, in which σ_{r_i} is the covariance of the cluster r_i in the class ω_i. Since ϵ_i,ϵ_j, ϵ_i', and ϵ_j', are very small at $\mathbf{x}$ and μ_i, they are ignored in the following σ estimation.

These two constraints imply that the cluster Θ_i' and the cluster Θ_j will not overwhelm each other. To satisfy (2.3.5), σ is initialized to be less than

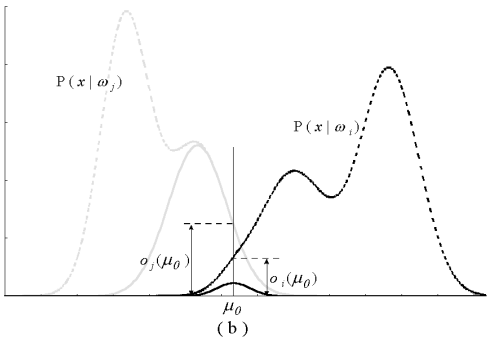
$$\left(\frac{P(\Theta_i \mid \omega_i)}{(2\pi)^{D/2} \overline{\sigma}_i o_j(\mathbf{x})} \right)^{\frac{2}{D-2}}.$$

Then, σ can be iteratively decreased by a small value η ($0 \leq \eta \leq 1$) until (2.3.6) is satisfied, and the final value of σ can be a proper initial value of σ_o for the new cluster Θ_i'.

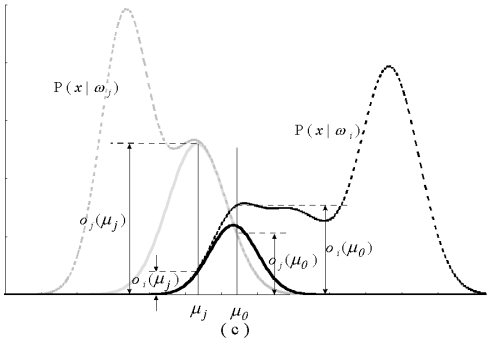
To verify the accuracy of the proposed SPDNN, the SPDNN based handwritten character recognition system is developed. For this application, the system is designed to recognize 5401 traditional Chinese characters. The implementation and measurement of the system are detailed in Section 4.



(a)



(b)



(c)

Figure 2.3: Example of creating a new cluster in a mixture of Gaussian distributions. (a) For $x \in \omega_i$, the x_i is not correctly classified, since $o_i(x)$ is smaller than $o_j(x)$. A new cluster Θ_i' is needed in ω_i. (b) The new cluster Θ_i' is *overwhelmed* by the cluster Θ_j, i.e., $o_i(\mu_0)$ is still smaller than $o_j(\mu_0)$. (c) By having initialized with proper μ_0, σ_0 and $P(\Theta_0 \mid \omega_i)$, the new cluster Θ_i' can contribute enough to support class ω_i. For example, $o_i(\mu_0)$ is larger than $o_j(\mu_0)$, and $o_i(\mu_j)$ is smaller than $o_j(\mu_j)$.

Chapter 3

Generalized Probabilistic Decision based Neural Network

In Chapter 2, we discussed the functionality of SPDNN to model the distribution of the data. SPDNN has a limitation in terms of the data that should be in the form of the numerical quantities. In some pattern recognition problems, the features of an object are in the form of the distributions instead of the numerical quantities. For example, the color feature of an image region may be defined as statistical information about the variation of the color across the region and described by a Gaussian distribution with mean vector and covariance matrix. To handle the general case that each datum is in the form of the distribution, I proposed a generalized version of SPDNN, called *Generalized Probabilistic Decision based Neural Network* (GPDNN).

The schematic of GPDNN is depicted in Fig. 3.1. Compared to SPDNN, GPDNN adapts the similar modular OCON structure, but is input with a distribution instead of a numerical quantity. Thus, the internal computation of each subnet in GPDNN is different from it in SPDNN. A detailed description of GPDNN model is given in the following sections.

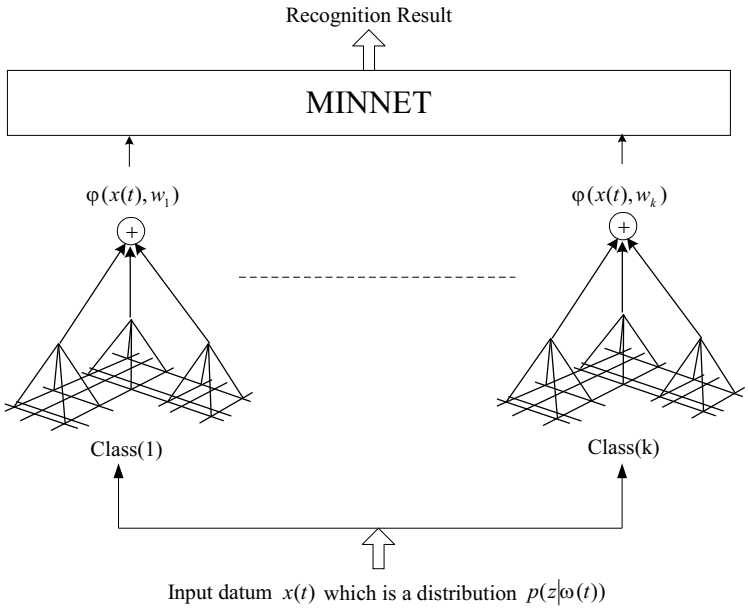


Figure 3.1: The schematic diagram of a *k-class* GPDNN. Each subnet is designed to represent a class. (The detail of a subnet is shown in Fig. 3.2.) Compared to SPDNN shown in Fig. 2.1, GPDNN is input with a distribution instead of a numerical quantity. When a testing datum $x(t)$ (a distribution $p(\mathbf{z} \mid \omega(t))$) is input to GPDNN, each subnet measures the difference between $x(t)$ and the corresponding class. Then, the MINNET assigns $x(t)$ to the class whose corresponding subnet have the minimum output among all subnets in GPDNN.

3.1 Discriminant Functions of GPDNN

Since the input data of GPDNN are distributions, the discriminate function of the multi-class GPDNN measures the difference between the input and the modelled distributions. We assume that the likelihood function for the class ω_i is $p(\mathbf{z} \mid \omega_i)$. For a testing datum $\mathbf{x}(t)$ formed by a distribution $p(\mathbf{z} \mid \omega(t))$, the difference of $p(\mathbf{z} \mid \omega_i)$ and $p(\mathbf{z} \mid \omega(t))$ can be considered as a Gaussiam noise ϵ_z,

so that

$$p(\mathbf{z} \mid \omega_i) = p(\mathbf{z} \mid \omega(t)) + \epsilon_{\mathbf{z}}. \tag{3.1.1}$$

We now assume that the error ϵ_z have a normal distribution with zero mean and standard a derivation σ. Thus, the distribution of ϵ is given by

$$\begin{aligned} p(\epsilon_{\mathbf{z}}) &= \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{\epsilon_{\mathbf{z}}^2}{2\sigma^2}\right) \\ &= \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(-\frac{[p(\mathbf{z} \mid \omega_i) - p(\mathbf{z} \mid \omega(t))]^2}{2\sigma^2}\right). \end{aligned} \tag{3.1.2}$$

Then, the similarity between $p(\mathbf{z} \mid \omega_i)$ and $p(\mathbf{z} \mid \omega(t))$ is defined as

$$\begin{aligned} \varphi(\mathbf{x}(t), \mathbf{w}_i) &= -\int_{R^D} \ln p(\epsilon_{\mathbf{z}}) d\mathbf{z} \\ &\approx \int_{R^D} [p(\mathbf{z} \mid \omega_i) - p(\mathbf{z} \mid \omega(t))]^2 d\mathbf{z} + \varepsilon, \end{aligned} \tag{3.1.3}$$

where ε is a constant. Suppose $p(\mathbf{z} \mid \omega_i)$ is the mixture Gaussian distribution shown in (2.2.1), then $\mathbf{w}_i = \{\mu_{r_i}, \mathbf{\Sigma}_{r_i}, P(\theta_{r_i} \mid \omega_i), T_i\}$, and T_i is the output threshold of the subnet i. When the value of $\varphi(\mathbf{x}(t), \mathbf{w}_i)$ becomes smaller, $p(\mathbf{z} \mid \omega_i)$ and $p(\mathbf{z} \mid \omega(t))$ are similar to each other.

Before showing the expression of (3.1.3), we first introduce the expression of the product moment of two distributions $\mathcal{P}_a$ and $\mathcal{P}_b$:

$$\mathcal{F}(\mathcal{P}_a, \mathcal{P}_b) = \int_{R^D} \mathcal{P}_a \mathcal{P}_b d\mathbf{z}. \tag{3.1.4}$$

Suppose $\mathcal{P}_a$ and $\mathcal{P}_b$ are two mixture Gaussian distributions:

$$\mathcal{P}_a = \sum_{r_a=1}^{R_a} P(\theta_{r_a}) p(\mathbf{z} \mid \theta_{r_a}) \tag{3.1.5}$$

$$\mathcal{P}_b = \sum_{r_b=1}^{R_b} P(\theta_{r_b}) p(\mathbf{z} \mid \theta_{r_b}), \tag{3.1.6}$$

where $p(\mathbf{z} \mid \theta_{r_a})$ and $p(\mathbf{z} \mid \theta_{r_a})$ with the parameter sets $\theta_{r_a} = \{\mu_{r_a}, \Sigma_{r_a}\}$ and $\theta_{r_b} = \{\mu_{r_b}, \Sigma_{r_b}\}$ are the Gaussian clusters in $\mathcal{P}_a$ and $\mathcal{P}_b$, respectively.

$\mu_{r_a} = [\mu_{r_a(1)} \cdots \mu_{r_a(D)}]^T$ and $\mu_{r_b} = [\mu_{r_b(1)} \cdots \mu_{r_b(D)}]^T$ are the mean vectors, and $\Sigma_{r_a} = diag[\sigma^2_{r_a(1)} \cdots \sigma^2_{r_a(D)}]^T$ and $\Sigma_{r_b} = diag[\sigma^2_{r_b(1)} \cdots \sigma^2_{r_b(D)}]^T$ are the diagonal matrixes. Then, $\mathcal{F}$ can be simplified to a closed-form expression as shown in the following lemma.

Lemma 3.1.1. *Suppose $\mathcal{P}_a$ and $\mathcal{P}_b$ are two mixture Gaussian distributions defined in (3.1.5) and (3.1.6), respectively. The product moment of $\mathcal{P}_a$ and $\mathcal{P}_b$,*

$$\mathcal{F}(\mathcal{P}_a, \mathcal{P}_b) = \sum_{r_a=1}^{R_a} \sum_{r_b=1}^{R_b} P(\theta_{r_a}) P(\theta_{r_b}) \mathcal{G}(\theta_{r_a}, \theta_{r_b}), \tag{3.1.7}$$

where

$$\mathcal{G}(\theta_{r_a}, \theta_{r_b}) = \frac{\exp\left\{ -\frac{1}{2} \sum_{d=1}^{D} \frac{(\mu_{r_b(d)} - \mu_{r_a(d)})^2}{\sigma^2_{r_b(d)} + \sigma^2_{r_a(d)}} \right\}}{\sqrt{(2\pi)^D \prod_{d=1}^{D} (\sigma^2_{r_b(d)} + \sigma^2_{r_a(d)})}}. \tag{3.1.8}$$

Proof. See Appendix. □

By applying Lemma 3.1.1, the following theorem presents the expression for (3.1.3).

Theorem 3.1.2. *Let $\mathcal{P}_i$ and $\mathcal{P}_t$ denote $p(\mathbf{z} \mid \omega_i)$ and $p(\mathbf{z} \mid \omega(t))$, respectively. Then, the discriminate function defined in (3.1.3) equals*

$$\mathcal{F}(\mathcal{P}_i, \mathcal{P}_i) - 2\mathcal{F}(\mathcal{P}_i, \mathcal{P}_t) + \mathcal{F}(\mathcal{P}_t, \mathcal{P}_t). \tag{3.1.9}$$

Proof. As shown in (3.1.3),

$$\begin{aligned}
\varphi(\mathbf{x}(t), \mathbf{w}_i) &= \int_{R^D} \mathcal{P}_{i-t}^2 d\mathbf{z} \\
&= \int_{R^D} (\mathcal{P}_i - \mathcal{P}_t)^2 d\mathbf{z} \\
&= \int_{R^D} (\mathcal{P}_i^2 - 2\mathcal{P}_i\mathcal{P}_t + \mathcal{P}_t^2) d\mathbf{z} \\
&= \int_{R^D} \mathcal{P}_i^2 d\mathbf{z} - 2\int_{R^D} \mathcal{P}_i\mathcal{P}_t d\mathbf{z} + \int_{R^D} \mathcal{P}_t^2 d\mathbf{z} \\
&= \mathcal{F}(\mathcal{P}_i, \mathcal{P}_i) - 2\mathcal{F}(\mathcal{P}_i, \mathcal{P}_t) + \mathcal{F}(\mathcal{P}_t, \mathcal{P}_t).
\end{aligned}$$

□

According to Theorem 3.1.2, the discriminate function can be calculated through (3.1.9) by a careful designed subnet, which is described in the following section.

3.1.1 The architecture of a subnet

Based on the expression (3.1.9) for the discriminate function, a subnet is designed as a two-layer neural network shown in Fig. 3.2. The first layer is composed with three structurally identical components, each of which computes the $\mathcal{F}(\mathcal{P}_i, \mathcal{P}_i)$, $\mathcal{F}(\mathcal{P}_i, \mathcal{P}_t)$, and $\mathcal{F}(\mathcal{P}_t, \mathcal{P}_t)$, respectively. By summing the computing results of these three components, the difference from the testing datum $\mathbf{x}(t)$ to the class ω_i is output at the top layer of the subnet.

Fig.3.3 depicts the internal architecture of the subnetwork corresponding to $\mathcal{F}(\mathcal{P}_i, \mathcal{P}_t)$. Suppose that the mixture Gaussian distributions $\mathcal{P}_i$ and $\mathcal{P}_t$ consist of R_i and R_t Gaussian clusters, then the subnetwork for $\mathcal{F}(\mathcal{P}_i, \mathcal{P}_t)$ contains $R_i \times R_t$ input units (called G nodes), R_i hidden units, and one output unit.

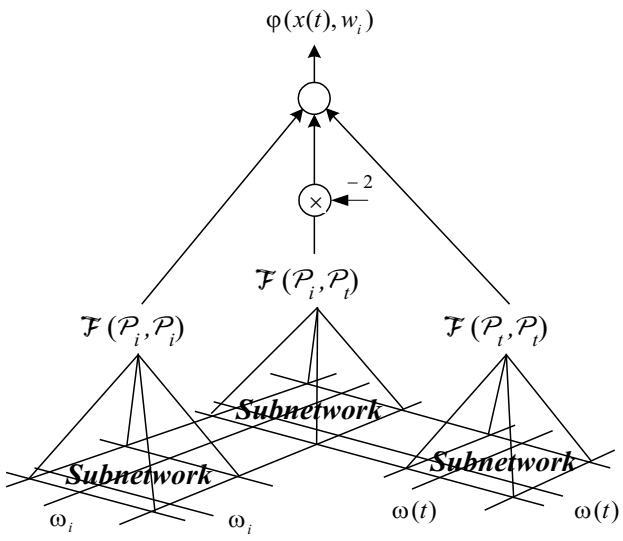


Figure 3.2: The diagram of the subnet in *Generalized Probabilistic decision based Neural Network* (GPDNN). Suppose the likelihood function for the class ω_i is $p(\mathbf{z} \mid \omega_i)$ (denoted as $\mathcal{P}_i$). For a testing datum $\mathbf{x}(t)$ formed by a distribution $p(\mathbf{z} \mid \omega(t))$ (denoted as $\mathcal{P}_t$), three subnetworks first compute $\mathcal{F}(\mathcal{P}_i, \mathcal{P}_i)$, $\mathcal{F}(\mathcal{P}_i, \mathcal{P}_t)$, and $\mathcal{F}(\mathcal{P}_t, \mathcal{P}_t)$, respectively. (The detail of a subnetwork to calculate $\mathcal{F}(\mathcal{P}_i, \mathcal{P}_t)$ is shown in Fig.3.3.) Then, $\mathcal{F}(\mathcal{P}_i, \mathcal{P}_t)$ times 2, and the difference from the testing datum $\mathbf{x}(t)$ to the class ω_i is output at the top of the subnet.

While the component is input with the testing datum $\mathbf{x}(t)$, formed by $\mathcal{P}_t$, each G_{r_i,r_t} node performs the computation of (3.1.8) to measure the difference between the Gaussian clusters r_i in $\mathcal{P}_i$ and the Gaussian clusters r_t in $\mathcal{P}_t$. Then, the outputs of these G nodes are weighted summed to hidden node h_i:

$$h_i = \sum_{r_t=1}^{R_t} P(\theta_{r_t} \mid \omega(t)) G_{r_i,r_t}. \tag{3.1.10}$$

Finally, the output of each hidden node $h_i, i = 1, \ldots, R_i$, are also weight summed

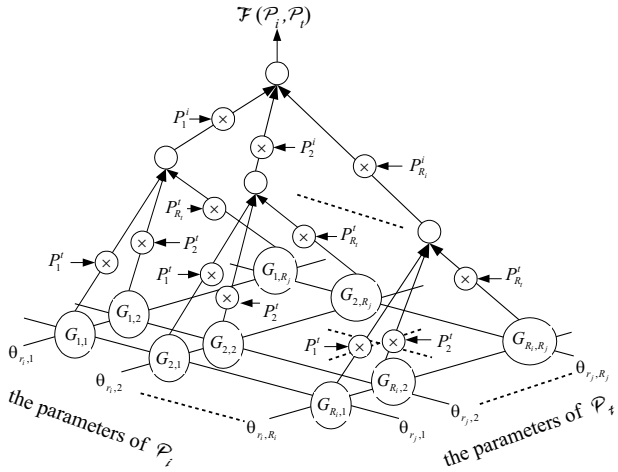


Figure 3.3: The architecture of a subnetwork in Fig. 3.2 to compute $\mathcal{F}(\mathcal{P}_i, \mathcal{P}_t)$. Suppose $\mathcal{P}_i = \sum_{r_i=1}^{R_i} P_{r_i}^i p(\mathbf{z} \mid \theta_{r_i}, \omega_i)$ and $\mathcal{P}_t = \sum_{r_t=1}^{R_t} P_{r_t}^t p(\mathbf{z} \mid \theta_{r_t}, \omega(t))$ are two mixture Gaussian distributions. The component contains $R_i \times R_t$ input units (called G nodes), R_i hidden units, and one output unit. Each G_{r_i,r_t} node measures the difference between the Gaussian cluster r_i in $\mathcal{P}_i$ and the Gaussian cluster r_t in $\mathcal{P}_t$ by performing (3.1.8), and each hidden unit forms a weighted sum of the R_j output values of the G nodes. $\mathcal{F}(\mathcal{P}_i, \mathcal{P}_t)$ is obtained from the output unit after a weighted sum of the R_i output values of the hidden units is computed.

to the output node:

$$\mathcal{F}(\mathcal{P}_i, \mathcal{P}_t) = \sum_{r_i=1}^{R_i} P(\theta_{r_i} \mid \omega_i) h_i. \tag{3.1.11}$$

After the outputs of all subnets are obtained, the MINNET in GPDNN is activated to select the minimum one among these outputs. That is, if the output value of the subnet i is the minimum one among the outputs of all subnets in GPDNN, the testing datum $\mathbf{x}(t)$ is classified to the class ω_i.

3.2 Learning Rules for GPDNN

GPDNN adopts the same ISLUG training scheme of SPDNN. Since the details of the ISLUG training scheme can be referred to Section 2.3, only a brief description of the ISLUG training scheme for GPDNN is provided here.

There are two learning phases in the ISLUG training scheme. In the *Supervised Learning* (SL) phase, teacher information is used to reinforce or antireinforce the decision boundaries between classes. The reinforced and antireinforced learning rules shown in (2.3.3) and (2.3.4) are applied to the misclassified subnets. Suppose that $\mathbf{X}_i^+ = \{\mathbf{x}_i(1), \cdots, \mathbf{x}_i(M_i)\}$ is a set of given training data, which correspond to one of the L classes $\{\omega_i, i = 1, \cdots, L\}$. A training datum $\mathbf{x}_i(m)$, formed by a distribution $p(\mathbf{z} \mid \omega_{i(m)})$, is classified to class ω_i if $\varphi(\mathbf{x}_i(m), \mathbf{w}_i) < \varphi(\mathbf{x}_i(m), \mathbf{w}_k)$, $\forall k \neq i$, and $\varphi(\mathbf{x}_i(m), \mathbf{w}_i) \leq T_i$, where T_i is the output threshold for subnet i. With the different discriminate function, the gradient vectors in (2.3.3) and (2.3.4) are computed as follows:

$$
\begin{aligned}
&\left.\frac{\partial \varphi(\mathbf{x}_i(m), \mathbf{w}_i)}{\partial \mu_{r_i(d)}}\right|_{\omega_{i(m)}=\omega_{i(k)}} \\
&= \frac{\partial}{\partial \mu_{r_i(d)}} \left(\mathcal{F}(\mathcal{P}_i, \mathcal{P}_i) - 2\mathcal{F}(\mathcal{P}_{i(k)}, \mathcal{P}_i)\right) \\
&= 2P(\theta_{r_i} \mid \omega_i) \left[\sum_{r_n=1}^{R_i} \left(\frac{P(\theta_{r_n} \mid \omega_i)\mathcal{G}(\theta_{r_n}, \theta_{r_i})}{\sigma_{r_n(d)}^2 + \sigma_{r_i(d)}^2}\right)\right. \\
&\cdot(\mu_{r_n(d)} - \mu_{r_i(d)}) \\
&- \sum_{r_{i(k)}=1}^{R_{i(k)}} \left(\frac{P(\theta_{r_{i(k)}} \mid \omega_{i(k)})\mathcal{G}(\theta_{r_{i(k)}}, \theta_{r_i})}{\sigma_{r_{i(k)}(d)}^2 + \sigma_{r_i(d)}^2}\right) \\
&\left.\cdot(\mu_{r_{i(k)}(d)} - \mu_{r_i(d)})\right], \qquad (3.2.1)
\end{aligned}
$$

and

$$\left.\frac{\partial\varphi(\mathbf{x}_i(m),\mathbf{w}_i)}{\partial\sigma^2_{r_i(d)}}\right|_{\omega_{i(m)}=\omega_{i(k)}}$$
$$= \frac{\partial}{\partial\sigma^2_{r_i(d)}}\left(\mathcal{F}(\mathcal{P}_i,\mathcal{P}_i) - 2\mathcal{F}(\mathcal{P}_{i(k)},\mathcal{P}_i)\right)$$
$$= P(\theta_{r_i} \mid \omega_i)\left[\sum_{r_n=1}^{R_i}\left(\frac{P(\theta_{r_n} \mid \omega_i)\mathcal{G}(\theta_{r_n},\theta_{r_i})}{\sigma^2_{r_n(d)}+\sigma^2_{r_i(d)}}\right)\right.$$
$$\cdot\left(\frac{(\mu_{r_n(d)}-\mu_{r_i(d)})^2}{\sigma^2_{r_n(d)}+\sigma^2_{r_i(d)}}-1\right)$$
$$-\sum_{r_{i(k)}=1}^{R_{i(k)}}\left(\frac{P(\theta_{r_{i(k)}} \mid \omega_{i(k)})\mathcal{G}(\theta_{r_{i(k)}},\theta_{r_i})}{\sigma^2_{r_{i(k)}(d)}+\sigma^2_{r_i(d)}}\right)$$
$$\left.\cdot\left(\frac{(\mu_{r_{i(k)}(d)}-\mu_{r_i(d)})^2}{\sigma^2_{r_{i(k)}(d)}+\sigma^2_{r_i(d)}}-1\right)\right], \tag{3.2.2}$$

where $r_i = 1, 2, \ldots, R_i$ and $d = 1, 2, \ldots, D$. D is the dimension of the feature space. Since the discriminate function of GPDNN is a quadratic function, its minimum can be found in terms of the solution of a set of linear equations. By supervised learning, the setting of the mixture Gaussian function parameters may be prone to finding local minima of the discriminant function defined in (3.1.3). However, if the initial cluster parameters of GPDNN are well determined, any given input datum will only generate a small movement of the cluster parameters. Since the EM algorithm [21] is guaranteed to decrease the error function of the quadratic form to the global minimum, it is used to initialize the cluster parameters of GPDNN.

When the supervised training progress becomes very slow or is trapped in a paralysis state, yet the classification or recognition accuracy is not at a satisfied level, the training enters the *unsupervised growing* (UG) phase. In the UG phase, a GPDNN creates a new cluster in a subnet according to the proposed *self-growing* rule (cf. Section 2.3.2). Thereafter, the training enters

the *supervised learning* phase again. The ISLUG learning procedure terminates when the training accuracy reaches a predefined satisfaction level.

GPDNN classifies the data which are in the form of distributions. In the case of representing image with the visual feature (color and/or texture) distribution over the image, the content-based image retrieval system can be built up based on GPDNN. The implementation and measurement of the GPDNN based content-based image retrieval are detailed in Section 5.

Chapter 4

Handwritten Character Recognition

In recent years, there is a significant increase in the electronic management of information by multimedia information systems. The handling of multimedia documents consists of the editing and display of texts, graphics, images, and handwriting. Currently, the keyboard and the mouse are still the dominant input devices for personal computer based multimedia systems. However, in preparing a first draft and concentrating on content creation, pencil and paper are often superior to keyboard entry. By incorporating character recognition with a text-to-speech technology, converting handwriting directly to voice will be an interesting multimedia application.

In this application, we propose a three phase (stage) handwriting recognition system, including (1) a global coarse classifier, (2) a user independent hand written character recognizer, and (3) a user adaptation module. In particular, an SPDNN is used to implement the kernel of the proposed personal handwriting recognition system.

4.1 Overview

The machine recognition of characters has been a topic of intense research since the 1960s [22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33]. After more than 30 years of rigorous attacks, studies in the field of handwriting recognition remain as active as ever.

Multi-linguistic documents are considered as a mixture of two or more kinds of character with different graphical structures, such as Chinese mixed with English. Documents contain several languages with same or similar graphical structures, such as most of the western languages, are not the interest of this research because they can be considered as one type of character on the recognition point of view. However, Chinese (including Japanese Kanji) characters are unique and different from those of western languages in that they are non-alphabetic and have quite complicated stroke structures. In general, directly applying several monolanguage character recognition techniques to each individual type of characters in a multilinguistic documents can be quiet difficult, owing to the following aspects:

1. separating mixed characters of different languages efficiently and correctly sometimes can be as hard as recognizing characters,

2. implementing two or more different types of recognition modules in a system is not time and space (in both software and hardware) efficient, and

3. combining recognition results from two different types of recognition modules is somewhat an unnecessary or a non-productive extra work.

Therefore, it is desirable to design a uniform recognition architecture for multilinguistic character recognition. First, select a set of general features for characters of different languages in a document, so that every character can be represented by a uniform feature vector. Comparing to a large character set like Chinese,[1] alphanumerics can be considered as a small subset of special characters to the larger character set. Then, a character recognition architecture for large character set can be adopted directly (or with minor modification) to multilinguistic character recognition. Thus, the uniform feature selection and the recognition architecture can be applied.

An SPDNN-based Handwritten Chinese major hybrid character recognition system is developed. The system configuration is depicted in Fig. 4.1. All the major processing modules, including pre-processing and feature selection modules, a coarse classifier, a character recognizer, and a personal adaptation module, are implemented on a personal computer.

The system built upon the proposed SPDNN model has been demonstrated to be applicable under reasonable variations of character orientation, size, and stroke width. This system also has been shown to be very robust in recognizing characters written using various tools, such as pencils, ink pens, marking pens and Chinese calligraphy brushes. As to the processing speed of the prototype system, the whole recognition process (including image preprocessing, feature selection, and character recognition) consumes approximately 0.14 second/character on a Pentium-II based personal computer, without using a hardware accelerator or co-processor.

[1]There are more than 40,000 modern Chinese characters, and 5401 characters are used frequently in daily life.

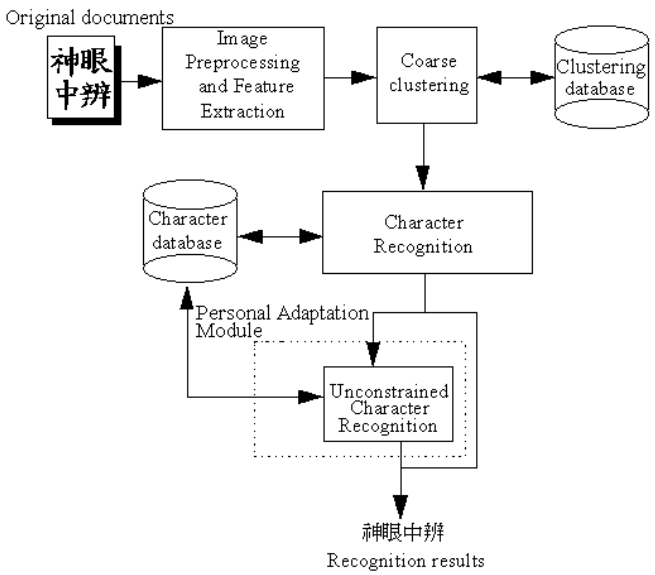


Figure 4.1: System configuration of the multistage character recognition system. Character recognition system acquires images from a scanner. The coarse classifier determines an input character image to be one of the predefined subclass. The character recognizer matches the input character with a reference character. The personal adaptive module learns the user's own written style to enhance the recognition accuracy.

4.2 Image Pre-processing and Feature Selection

Image preprocessing of a multilinguistic character recognition is by no means of any different from the monolanguage character recognition. Character segmentation on free format handwritten character is a very difficult task, thus it is usually an interactive task between segmentation and recognition. Since the multilinguistic character recognition is already a complicated recognition problems, and the interactive segmentation methods would slow down the pro-

cessing speed, we must restrict our handwritten character domain to be free format on Chinese characters and handprinted characters on alphanumerics. Thus, an *Interactive Rule-Based Character Segmentation* [34] is applied to slice and separate the whole page image into a sequence of character image. Basically, this method is based on some heuristic rules to combine several isolated connected-components into a separated character.

The binary images of a handwritten character are then passed through a series of image processing stages, such as boundary smoothing, noise removing, space normalization, and stroke thinning operations. Figure 4.2 depicts a series of preprocessing results of some Chinese and English characters.

Figure 4.2: Image preprocessing on handwritten characters: (from top) original text image, smoothed text, linear normalized text, nonlinear normalized text and thinned text.

Feature extraction Using statistical features in pattern recognition has been very successful for a long time. A character can be well represented by a 2-D image pattern, thus many statistical pattern recognition techniques have been applied in this type of character recognition. Among various statistical

features [35], we selected the *crossing count* (CCT), *belt shape pixel number* (BSPN), and *stroke orientation feature* (STKO) as candidate features for the proposed character recognition system. As shown in Figure 4.3, features such as *CCT* and *BSPN* represent the stroke complexity and the pixel density of a character image. In [36], Kimura et al. proposed the directional code histogram, and used this feature for Chinese character recognition successfully. As shown in Figure 4.5, the *STKO* is a simplified version of Kimura's directional code histogram.

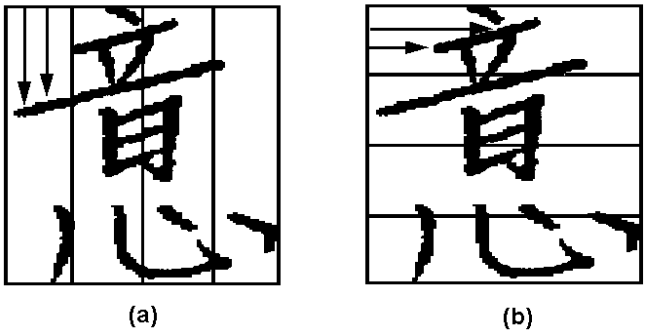


Figure 4.3: Extracting the CCT and BSPN features from a Chinese character.

4.3 Multistage character recognition

Since there are as many as 5401 commonly used characters, and 62 alphanumerics and symbols in a Chinese majored multilinguistic document, it is desirable to perform a coarse classification (or clustering) to reduce the number of candidate characters for the character recognition. With a smaller candidate set, not only the overall recognition speed and recognition accuracy can be greatly improved, but also the training on the SPDNN character recognizer can be much easier and faster. The architecture of a multistage SPDNN character recognizer

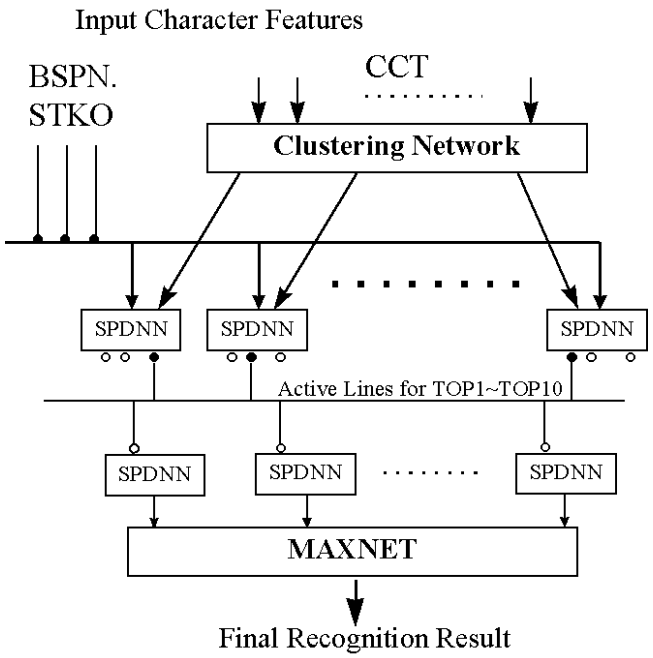


Figure 4.4: The architecture of the three-stage recognition system.

is depicted in fig. 4.4.

4.3.1 Global training on coarse classification

In order to achieve a balanced recognition performance in a multistage recognition system, the coarse classifier needs to maintain a very high accuracy, (e.g., $\geq 99.9\%$). Although this is a difficult task, we proposed to use the *CCT* feature and SPDNN with *overlapped* boundaries to implement the coarse classifier. This design is to achieve low sensitivity in personal writing style and high classification rate among characters. By applying the *ISLUG* principle on the two public databases, i.e., CCL/HCCR1 [37] and CEDAR [38], we have trained the proposed coarse classifier to achieve this goal. The training and testing results are listed in Table 4.1. At the end of the retrieving phase of the coarse classi-

fier, the number of candidate characters with respect to the input character is reduced to 516 characters in average.

Table 4.1: The training and testing results of coarse classification on the CCL/HCCR1 and the CEDAR Databases. Half of the randomly selected characters in each of databases are used for training and the other half are used for testing.

Number of cluster	Ave. No. of characters in a cluster	Training Accuracy	Testing Accuracy
61	516	99.9 %	99.8%

4.3.2 Batch training in character recognition

The design of the character recognizer is also based on the SPDNN model. For a K-character recognition problem, an SPDNN character recognizer consists of K subnets. A subnet i in the SPDNN recognizer estimates the distribution on the patterns of character i only, and treats those patterns which do not belong to character i as the "non i" patterns. The combined features such as CCT, BSPN, and STKO are used in the SPDNN character recognizer. The training of the character SPDNN was conducted with the *ISLUG* principle. During the retrieving phase, each of the subnets corresponding to the candidate characters from the coarse classifier produces a score according to its discriminate function $\phi(\mathbf{x}(t), \mathbf{w}_i)$. The subnet which produces the highest score is the winner and its corresponding reference character is considered as the result of the character recognizer.

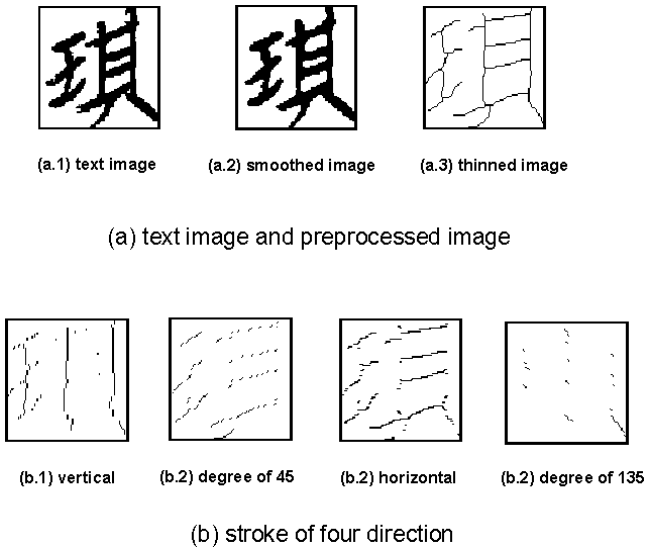


Figure 4.5: Preprocessing and STKO feature extraction of a Chinese character.

Two experimental results from the SPDNN character recognizer will be discussed. The first type of recognition experiment were performed on the CCL/HCCR1 [37] handwriting database, which has been used by several handwriting recognition research groups [34, 35, 1]. The second type of experiment explored the ability of SPDNN to deal with the multi-linguistic handwriting recognition problem, which has seldom been discussed in the character recognition literature.

(1) *Experiment 1–Handwritten Chinese Recognition*:

We have conducted experiments on the CCL/HCCR1 database, which contains more than 200 samples of 5401 frequently used Chinese characters. The samples were collected from 2600 people including junior high school and college students as well as employees of ERSO/ITRI. According to the most recent survey on handwriting recognition [25, 27, 39], most of the handwritten Chinese OCR studies are designed for small databases, i.e., training and testing

on very small character sets, e.g., a few hundred characters. As for studies conducted on recognition using a complete set of commonly used Chinese characters, Xia [40] developed an experimental system with a 3755 character set and achieved an 80% of recognition rate. In [1], Li and Yu reported 88.65% recognition accuracy on the CCL/HCCR1 database. Recently, Tseng *et al.* [2] used the *M_distance* method in their recognition system to achieve 88.55% accuracy on the CCL/HCCR1 database. Table 4.2 shows the training and testing accuracy of Gaussian model as well as SPDNN with and without the *unsupervised growing* phase. Each subnet of SPDNN is initialized with one Gaussian cluster. At the end of the training, the distribution of the number of clusters in each subnet of SPDNN is shown in Table 4.3. The recognition accuracy of Gaussian model without any training is 83.11%, and is improved to 85.18% by fine tuning the decision boundary between classes via the *supervised learning* of the SPDNN. After the *unsupervised growing*, the recognition accuracy can be further improved to 86.12%.

Table 4.2: The training and testing accuracy of Gaussian model as well as SPDNN with and without the *unsupervised growing* phase. Each subnet of SPDNN is initialized with one Gaussian cluster.

Various	Gaussian Model	SPDNN (mix=1)	SPDNN
Training set	89.49%	94.87%	97.78%
Testing set	83.11%	85.18%	86.12%

Table 4.4 summarizes a performance comparison of these systems evaluated using the CCL/HCCR1 database. We would like to comment on the overall performance of these systems as follows. First, compared to the huge number

Table 4.3: The distribution of the number of clusters in each class.

Number of clusters	1	2	3	4	5
Number of class	3697	1376	252	69	7

of character features used by other researchers, *e.g.*, 400 features used in [1] or 256 features used in [2], the SPDNN recognizer uses only 92 features. A more relevant comparison could be made if a comparable number of training and testing features for these two systems were available. In fact, the SPDNN character recognizer is designed to use no more than 100 sets of features since more feature sets would require more memory storage and longer recognition time. Two reasons explain why an SPDNN-based system can have fewer features yet achieve comparable performance. (1) The mixed Gaussian-based discrimination function permits SPDNN to learn the character decision boundary precisely. (2) The self-growing rules allow a small number of Gaussian clusters to be sufficient to represent the character image distribution.

Table 4.4: Performance of different handwriting recognizers on the CCL/HCCR1 database. A portion of this Table is adapted from Li et al. [1] and Tseng et al. [2].

Various Systems	Recognition Accuracy	Features Used	Train & testing data used	Classification time
SPDNN	86.12%	92	50-50	0.24 sec/char
Li *et al.*	88.65%	400	50-1	NA
Tseng *et al.*	88.55%	256	100-100	0.6 sec/char

(2) *Experiment 2–Multilinguistic Handwriting Recognition:* By searching

on major conference proceedings, journals as well as Web sites, we have not found any performance test report on this type of handwritings. We therefore conducted experiments on the combined databases of CCL/HCCR1 and CEDAR [38]. The training and testing data sets for Chinese characters are selected by the same way used in the *Experiment 1*. The CEDAR database contains various style of handwritten alphanumerics, which were lifted from envelop address blocks from USA. Among the data, 4000 alphanumerics were used for training and 2000's for testing. We also conducted experiments with rejection. Rejection criteria was implemented through the threshold value T_i, which can be learned by the reinforced and antireinforced learning rules. In general, when an input character is correctly recognized with certain confidence, its output of the discriminate function should maintain a certain gap larger than T_i with respect to the second largest output from other discriminate functions.

The experimental results are discussed as follows: For the sake of comparison, we adjusted the thresholds T_i so that the proposed system has 0% false rejection rate during the training phase. The recognition accuracy with 0% and 6.7% of false rejection rates at the testing phase are shown in Table 4.5. Li and Yu's method can not provide rejection function in their Bayesian rule based statistical recognition system [1]. However, SPDNN's rejection function is based on the reinforced and antireinforced learning rules, thus each subnet, which represents a character in SPDNN, can have its own rejection criteria. We think this characteristic is beneficial for real world applications.

Table 4.5: Performance of SPDNN handwritten character recognizers with and without rejection on the CCL/HCCR1 and CEDAR databases.

Systems	Top 1 Accu.	Top 2 Accu.	Top 3 Accu.
SPDNN (rej=0%)	90.12%	93.49%	94.75 %
SPDNN (rej=6.7%)	94.11%	97.01%	97.67 %

4.3.3 Personal adaptation in handwriting recognition

Most of the recently announced handwritten character recognition systems claimed their benchmarking recognition performance to be higher than 90%. However, when they were tested on unconstrained freehand-writing, most of their recognition accuracy fell between 40% and 50% [41]. Hence, we suggested an unconstrained freehand-writing recognition module to adaptively fine tune the parameters of the SPDNN character recognizer in order to learn the user's own writing style. When input characters were misclassified, the erroneous recognition results will be manually corrected by a user. In the mean time, the parameters or the decision boundaries of the corresponding character SPDNN are modified and improved by performing the reinforced and antireinforced learning processes. In addition, when it is necessary, clusters in a character SPDNN may be created (self-growing rules) to better approximate the partition boundaries. In order to prevent the excessive learning of the designated character boundary, the adaptive learning process usually include a verification process. Naturally, the reinforced and antireinforced learning processes are applied to SPDNN associated with the mismatched character and its similar characters (the TOP 10 candidates). When more and more uncon-

strained freehand-written characters are presented to the system, each character SPDNN will gradually learn the user's personal writing style.

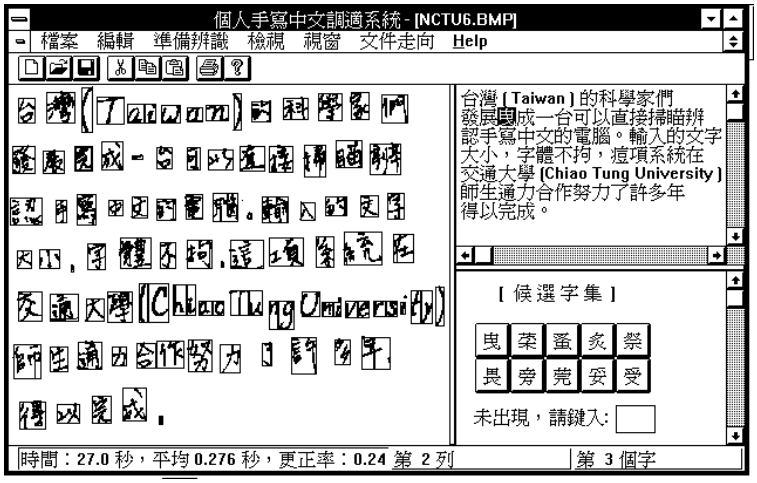


Figure 4.6: The user interface and a recognition snapshot of the proposed three-stage recognizer.

Experimental results and Performance evaluation

In order to evaluate the performance of the unconstrained freehand-writing recognition module for its adaptation and recognition capabilities, we prepared our in house database (NCTU/NNL) in the following manner. We first selected the most commonly used 300 characters from the Chinese textbooks for the elementary schools in Taiwan. And then, these 300 Chinese characters and the alphanumerics were written without any restriction on the writing style by several students in our university for 10 times in several days. We intended to simulate a natural and general unconstrained freehand-written database in this manner. The testing results for 5 user's adaptation processes are illustrated in Table 4.6. The recognition rates was raised from 44.09% to 82.2% during the 5 learning cycles. And the performance may finally increase up to 90.03%

in 10 learning cycles. Figure 4.6 depicts the user interface and a snapshot of recognition results of the prototype system.

Table 4.6: By applying 300 commonly used characters written without any constraints by five students, the proposed adaptive system shows significant improvement on the recognition accuracy during the 10 learning cycles.

Trial	user#1	user#2	user#3	user#4	user#5	avg.
1st	50.6%	33.6%	38.1%	52.3%	45.7%	44.0%
2nd	67.7%	69.0%	55.5%	56.6%	61.1%	62.0%
3rd	78.6%	80.0%	69.9%	71.5%	72.7%	74.5%
4th	84.3%	78.7%	69.3%	75.9%	86.0%	79.5%
5th	84.6%	87.6%	73.9%	79.9%	85.1%	82.2%
6th	81.9%	89.0%	76.2%	80.2%	84.0%	82.2%
7th	86.5%	89.6%	79.9%	78.5%	84.6%	83.8%
8th	89.5%	90.3%	79.5%	86.9%	89.3%	87.1%
9th	90.5%	90.6%	81.2%	87.7%	89.3%	87.9%
10th	93.6%	91.4%	84.6%	90.5%	90.1%	90.0%

Chapter 5

Content-based Image Retrieval

The ongoing proliferation of digital content available over Internet leads to an increasing demand for systems that can automatically query, search, and retrieve of relevant images from large content databases and/or library. To construct such systems, two issues have to be considered: (1) how to properly index an image, and (2) how to design a user friendly query method. Over the past decades, a considerable number of studies have been made on content-based image retrieval (CBIR), where images are indexed and retrieved by their visual features, such as object shape, position, color, texture, etc. [42, 43, 44, 3, 4, 45, 46].

5.1 Overview

According to the different query methods, image query systems can be divided into (1) the full automatic, and (2) the user feedback query categories. For the full automatic query systems [47, 48, 49, 50], a user specifies several related images to vaguely reflect his/her desired images, and then the query systems

respond with a bunch of so called related images. Sometimes, most of these images may be undesired due to misinterpretation between users and the query system. On the other hand, to make the retrieval results to be more satisfactory, the user feedback query systems [51, 52] specify some details of contents instead of just the image itself, such that the query systems can directly use these information for searching and matching suitable images.

In a full automatic query system, indexing an image with its global features, such as color histograms, are often used for image retrieval. Some early developed systems, such as QBIC [47], Virage [48], Photobook [49], VisualSEEk and WebSEEk [50] basically applied global features for image retrieval. However, using global features for image indexing, a query system may ignore some significant local details of an image, so as to retrieve undesired images.

Instead of using global features of an image, the user feedback query systems, such as the Netra [51] and Blobword [52], adopt local features to represent or to index an image. In these systems, the local features are obtained from some regions or subimages, which are segmented or sketched from an image first, and then various visual features of these regions are extracted. In general, the query and retrieving precision of these systems are usually better than the global feature based systems, but their performance depend heavily on the precise segmentation or skillful sketch of a region.

For the past decades, segmenting an image into semantic meaningful regions is still a difficult task in image processing [42, 43, 44]. Instead of emphasizing on the precise region segmentation, the Integrated Region Matching (IRM) metric [3] is proposed to robust measure the similarity between regions and reduces the influence of inaccurate segmentation. In addition, a region-

based fuzzy feature matching approach, called unified feature matching (UFM) [4], is proposed to characterize each region with a fuzzy feature set; thus, an image is associated with a family of fuzzy feature sets. Since fuzzy features naturally characterize the blurry boundaries between regions, the influence of inaccurate segmentation is reduced.

Since an image can be partitioned into several sub-images, called regions or objects, the spatial relationship of these regions plays an important role in representing an image. The 2D B-string [53] is proposed to represent spatial relation of regions, where each region is represented by two symbols: the begin boundary and the end boundary symbols. With these symbols, a 2D B-string can represent the spatial relation of partial overlap regions without a boundary cutting process.

Since the 2D B-string represents only the spatial relation of regions in an image, it may correspond to two sets of regions of similar spatial relations but completely different in shapes and sizes. Thus, in addition to 2D B-string, more visual features, such as color, texture, and shape are needed to represent regions and to index an image for query and retrieving purposes.

The kernel problem of image retrieval lies on the representing a user's desired images, which is conceptually resided in his/her mind, into a set of computable image processing formulas or models. Similar to the keywords for the text query, the *visual keywords* are proposed for the image query and retrieval. Instead of annotating each region in an image by keywords, I represent a region with a *visual keyword*, and spatial relation of regions with a *visual string*.

A *Neural Networks based Image Retrieval System* (NNIRS) is developed

at "http:// 140.113.216.78/ ImageQuerySystem". The system configuration is depicted in Fig. 5.1. All the major processing modules, including pre-processing and feature extraction modules, a *Visual Keyword based Retrieval Module* (VKRM), and a *Visual String based Retrieval Module* (VSRM), are implemented on a personal computer.

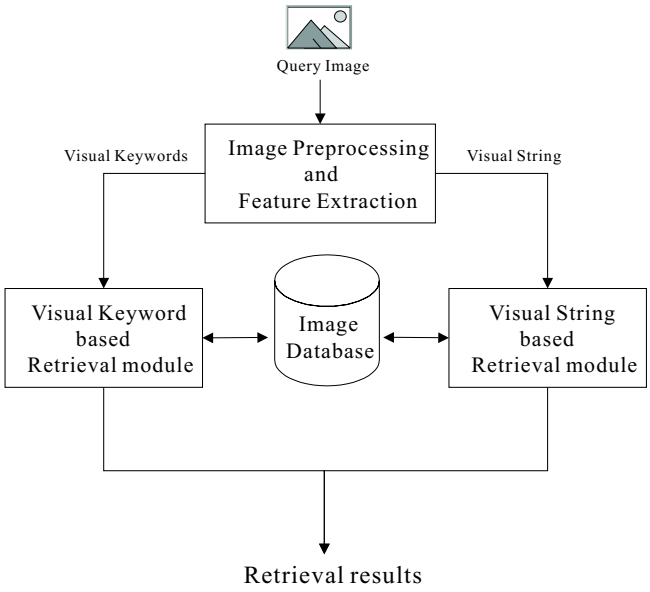


Figure 5.1: System configuration of the Neural Networks based Image Retrieval System (NNIRS). The pre-processing and feature extraction module is used to extract the *visual keyword* or *visual string* from the image. The *visual keyword* and *visual string* based retrieval modules are used to find the relevant images in the database using *visual keywords* or *visual string*, respectively.

The user can query the NNIRS by *visual keyword* or *visual string*. When a query image is submitted, the pre-processing and feature extraction module is first activated to extract the *visual keyword* or *visual string* from the image depending on the query requirement. Then, the VKRM or VSRM is used to find the relevant images in the database using *visual keywords* or *visual string*.

The following sections show the details of these modules in the NNIRS.

5.2 Image Pre-processing and Feature Extraction

The flowchart of *Image Pre-processing and Feature Extraction* is shown in Fig. 5.2. First, an image is blurred to remove the noise. Then, the pixels of closer color or similar texture features form several homogeneous regions. Finally, the *visual keywords* and *visual string* are generated to represent these homogeneous regions and the spatial relation of regions, respectively.

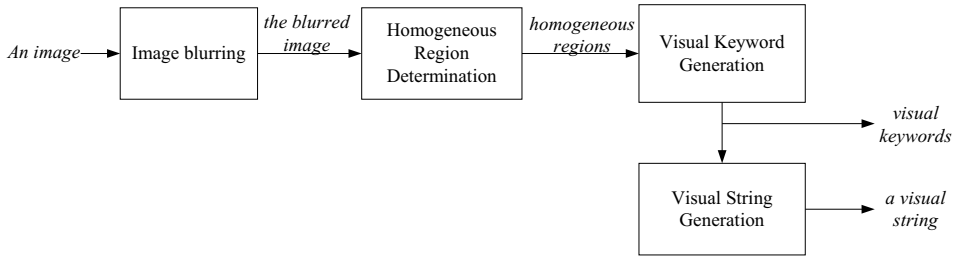


Figure 5.2: The flowchart of *Image Pre-processing and Feature Extraction*. The given image is first blurred. Then, several homogeneous regions are determined based on the pixels of closer color or similar texture features in the blurred image. At last, the *visual keywords* and *visual string* are generated to represent these homogeneous regions and the spatial relation of regions, respectively.

5.2.1 Image blurring

The Gaussian kernel based low-pass filter [54] is commonly used for image blurring. However, applying this technique may also blur natural edges or boundaries of an image. To preserve the sharpness of the natural boundaries, the anisotropic diffusion method was proposed [55]. The anisotropic diffusion process would blur only the interior of a natural region with closer color value, and leave the boundaries cleared and sharped. An example of the image blurring is shown in Fig. 5.3. The original image is shown in Fig. 5.3(a); the blurred images by the Gaussian kernel low pass filter, and the anisotropic diffusion are shown in Fig. 5.3(b) and (c), respectively.

(a) The original image

(b) The blurred image after Gaussian kernel low pass filter

(c) The blurred image after the anisotropic diffusion method

Figure 5.3: An example of the blurred images by the Gaussian kernel low pass filter (b) and the anisotropic diffusion method (c).

5.2.2 Homogeneous Region Determination

After an image is blurred, a homogeneous region is determined based on a set of the pixels with closer color and similar texture in the blurred image. The following color and texture features are involved in the *Homogeneous Region*

Determination.

The color feature: The $L^*a^*b^*$ color system is chosen to represent the color component of a pixel since it is approximately perceptual uniform [56]; thus the distance between two colors in this color system conforms well to human perceptual distance as measured by psychophysicists. Suppose the dimension of the image I is $N \times M$. Then, the color feature of a pixel $\mathbf{x}$ is represented by a 3-dimension vector $\mathbf{c_x}$ where each dimension corresponds to one of the color components in the $L^*a^*b^*$ color space.

The texture feature: Since the Gabor representation is optimal [57] in the sense of minimizing the uncertainty in the space and the frequency domain, the Gabor wavelet decomposition [58] is used to extract the texture features from the image of multiple scales and orientations. Before to decompose an image, a Gabor filter set is created from a two-dimensional Gaussian-modulated complex sinusoid function

$$g(x,y) = \left(\frac{1}{2\pi\sigma_x\sigma_y}\right) \exp\left[-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}\right) + j\omega x\right], \tag{5.2.1}$$

where σ_x and σ_y determine the scale of the Gaussian envelope along the respective axes, and ω is the filter center frequency. By selecting T dilations and K rotations of the rectilinear coordinates of $g(x,y)$, a Gabor filter set $G_f = \{g_{tk} : 1 \le t \le T, \text{ and } 1 \le k \le K\}$ is created from

$$g_{tk}(x,y) = a^{-t} g(x', y'), a > 1$$

where

$$\begin{aligned} x' &= a^{-t}(x\cos\theta + y\sin\theta), \\ y' &= a^{-t}(-x\sin\theta + y\cos\theta), \end{aligned}$$

where θ is the angle of the rotation of the rectilinear coordinate, i.e., $\theta = k\pi/K$. With a Gabor filter g_{tk}, the filter response of the image I can be calculated by the following convolution:

$$I_{tk} = I * g_{tk},$$

and its spectrogram is calculated as

$$S_{tk}(\mathbf{x}) = |I_{tk}(\mathbf{x})|^2. \tag{5.2.2}$$

Then, the texture feature around a pixel $\mathbf{x}$ is represented by a $K \times T$-dimension vector $\mathbf{t_x}$ where each dimension corresponds to one of the elements in $\{S_{tk}(\mathbf{x}) : 1 \leq t \leq T \text{ and } 1 \leq k \leq K\}$. By combining the color and texture features, an $N \times M$ visual feature array C_I is built up, where each element $C_I(\mathbf{x})$ is a visual feature vector $[\mathbf{c_x}, \mathbf{t_x}]^T$.

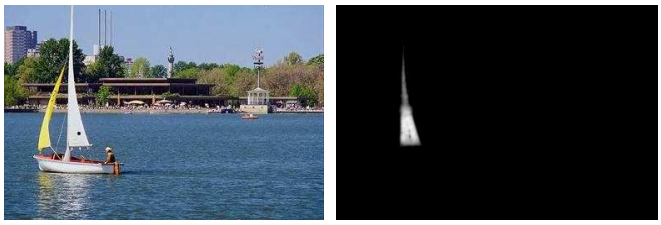
Instead of performing the precise image segmentation, a homogeneous region is "grown up" from a given reference pixel until the visual features of the pixels around the homogeneous region are far from of the reference pixel. Based on the visual feature array, a homogeneous region A_i is determined as the surrounding area of the reference pixel $\mathbf{x}_s$ with closer color and similar texture features, such that

$$A_i = \{\mathbf{x} : \mathbf{x} \in I, \|C_I(\mathbf{x}) - C_I(\mathbf{x}_s)\| \leq \lambda\}, \tag{5.2.3}$$

where $\|\cdot\|$ is a norm operator, and λ is a given tolerance threshold. Meanwhile, the similarity degrees between the pixels in A_i and the reference pixel $\mathbf{x}_s$ are stored in an $N \times M$ *Homogeneous Region Array* (HRA) H_i, where the similarity degree of a pixel $\mathbf{x}$ is calculated as

$$H_i(\mathbf{x}) = \begin{cases} 1 - \frac{\|C_I(\mathbf{x}) - C_I(\mathbf{x}_s)\|}{\lambda} & \text{for } \mathbf{x} \in A_i \\ 0 & \text{otherwise.} \end{cases} \tag{5.2.4}$$

An example of a homogenous region to represent the sail of the boat is shown in Fig. 5.4. The original image with a reference pixel (the black dot) is depicted in Fig. 5.4(a), and its HRA is illustrated as a brighten region in Fig. 5.4(b). It is clear that the HRA exactly presents the sail of the boat. After a homogeneous region is formed, it is represented by a *visual keyword*, which is introduced in the following section.



(a) (b)

Figure 5.4: An example of a homogenous region determination. (a) The original image with a reference pixel showing as a black dot on a sail. (b) The corresponding HRA of the reference pixel is shown as a light gray image.

5.2.3 Visual Keyword Generation

Similar to the (text) keywords for representing the key information of a document, the *visual keyword* is proposed to illustrate the visual key characteristics of an image. In general, an image can be characteristically specified by a few objects, each of which usually is composed of one or few near homogenous regions. For each of these regions, a set of visual features such as color, texture, and shape can be extracted to represent the region. With these visual features, the *visual keyword* is defined as follows.

Definition 5.2.1 (Visual Keyword). Given a homogeneous region i in an image, the *visual keyword* ω_i is a triple of Gaussian mixture models (GMM)

$\{G_i^s, G_i^c, G_i^t\}$ to formulate the **s**patial, **c**olor, and **t**exture features of the region i. The 2D GMM G_i^s approximates the spatial features (location and shape) of the region i according to its means and the covariance matrices. The other GMMs G_i^c and G_i^t formulate the average and variation of color and texture features over the region i by their means and the covariance matrices, respectively.

The key issues for the *visual keyword* representation are shown as follows: (1) The precise segmentation or skillful sketch of the region is no longer needed, and (2) the approximating a region by mixture Gaussian distributions allows the searching for its similar regions more flexible and robust. An exemplar of the *visual keyword* is shown in Fig. 5.5. The three *visual keywords*, shown as the elliptic regions ω_1, ω_2, and ω_3, are created to cover the two sails and the boat body. As we can see, the shape and the location of these regions can be formulated by three 2D mixture Gaussian distributions, respectively.

Figure 5.5: An example of the *visual keyword*. The *visual keywords* ω_1, ω_2 and ω_3 are used to represent the sailboat in the image.

A *visual keyword* is generated to formulate the spatial, color and texture features of a homogeneous region via two steps: (1) the spatial modelling and

(2) the color and texture modelling.

Spatial Modelling

For illustration purpose, we often use an elliptic region to illustrate a 2D Gaussian distribution. In addition, the shape of an elliptic region can be altered by changing the parameters (the mean, the covariance matrix, and the prior probability) of its corresponding 2D Gaussian distribution. Thus, an arbitrary shaped region can be approximated by the union of several elliptic regions. In the following, we will present the methods and procedures of adjusting the parameter values of a 2D mixture Gaussian distribution.

For a given homogeneous region $A_i = \{\mathbf{x}(l) : l = 1, 2, \ldots, L\}$ and its corresponding *Homogeneous Region Array* (HRA) H_i, where L is the number of pixels in A_i, suppose a 2D mixture Gaussian distribution $p_s(\mathbf{x}(l) \mid \omega_i)$ formulates the spatial feature of A_i. Define $p_s(\mathbf{x}(l) \mid \theta_{s,r_i}, \omega_i)$ as a Gaussian cluster to comprise $p_s(\mathbf{x}(l) \mid \omega_i)$, i.e.,

$$p_s(\mathbf{x}(l) \mid \omega_i) = \sum_{r_i=1}^{N_i} P_s(\theta_{s,r_i} \mid \omega_i) p_s(\mathbf{x}(l) \mid \theta_{s,r_i}, \omega_i), \tag{5.2.5}$$

where θ_{s,r_i} represents the parameter set $\{\mu_{s,r_i}, \Sigma_{s,r_i}\}$, and $P_s(\theta_{s,r_i} \mid \omega_i)$ denotes the prior probability of the cluster r_i. By definition, $\sum_{r_i=1}^{N_i} P_s(\theta_{s,r_i} \mid \omega_i) = 1$, where N_i is the number of clusters in $p_s(\mathbf{x}(l) \mid \omega_i)$. Suppose the cluster r_i is a 2D Gaussian distribution:

$$p_s(\mathbf{x}(l) \mid \theta_{s,r_i}, \omega_i) = \frac{\exp\left\{-\frac{1}{2}(\mathbf{x}(l) - \mu_{s,r_i})^T \Sigma_{s,r_i}^{-1} (\mathbf{x}(l) - \mu_{s,r_i})\right\}}{2\pi |\Sigma_{s,r_i}|^{1/2}}. \tag{5.2.6}$$

The dissimilarity between the HRA H_i and 2D mixture Gaussian distribu-

tion $p_s(\mathbf{x}(l) \mid \omega_i)$ can be measured by the cross-entropy function

$$E = -\sum_{l=1}^{L} H_i(\mathbf{x}(l)) \ln(p_s(\mathbf{x}(l) \mid \omega_i)), \tag{5.2.7}$$

regarded as an error function between the region A_i and *visual keyword* ω_i. By applying the EM algorithm, (5.2.7) is minimized by the following update equations for the parameters of 2D mixture Gaussian distribution: At each epoch j,

$$\mu_{s,r_i}^{(j+1)} = \frac{\sum_{l=1}^{L} H_i(\mathbf{x}(l)) p_s^{(j)}(\theta_{s,r_i} \mid \mathbf{x}(l), \omega_i)\mathbf{x}(l)}{\sum_{l=1}^{L} H_i(\mathbf{x}(l)) p_s^{(j)}(\theta_{s,r_i} \mid \mathbf{x}(l), \omega_i)},$$

$$\Sigma_{s,r_i}^{(j+1)} = \frac{\sum_{l=1}^{L} H_i(\mathbf{x}(l)) p_s^{(j)}(\theta_{s,r_i} \mid \mathbf{x}(l), \omega_i)(\mathbf{x}(l) - \mu_{s,r_i}^{(j)})(\mathbf{x}(l) - \mu_{s,r_i}^{(j)})^T}{\sum_{l=1}^{L} H_i(\mathbf{x}(l)) p_s^{(j)}(\theta_{s,r_i} \mid \mathbf{x}(l), \omega_i)},$$

$$P_s^{(j+1)}(\theta_{s,r_i} \mid \omega_i) = \frac{\sum_{l=1}^{L} H_i(\mathbf{x}(l)) p_s^{(j)}(\theta_{s,r_i} \mid \mathbf{x}(l), \omega_i)}{\sum_{l=1}^{L} H_i(\mathbf{x}(l))},$$

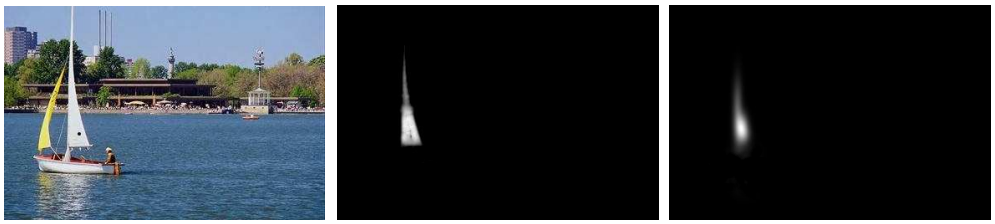
where

$$p_s^{(j)}(\theta_{s,r_i} \mid \mathbf{x}(l), \omega_i) = \frac{P_s^{(j)}(\theta_{s,r_i} \mid \omega_i) p_s^{(j)}(\mathbf{x}(l) \mid \theta_{s,r_i}, \omega_i)}{\sum_{r_i=1}^{N_i} P_s^{(j)}(\theta_{s,r_i} \mid \omega_i) p_s^{(j)}(\mathbf{x}(l) \mid \theta_{s,r_i}, \omega_i)}.$$

The iteration of EM computation is continuous until (5.2.7) becomes less than a given threshold. Fig. 5.6 illustrates the spatial modelling of a sail boat. The original image with a reference point (the black dot) is depicted in Fig. 5.6(a), and the corresponding homogenous region and its spatial model, a 2D GMM comprised of two Gaussian clusters, is depicted in Fig. 5.6 (b) and (c, respectively.

Color and Texture Modelling

After the spatial modelling is done, the homogeneous region A_i is approximated by the 2D mixture Gaussian distribution $p_s(\mathbf{x}(l) \mid \omega_i)$. Suppose $p_s(\mathbf{x}(l) \mid \omega_i)$



(a) (b) (c)

Figure 5.6: The spatial feature modelling of a sail region. (a) The original image with a reference point shown as a black dot on a sail. (b) The corresponding homogeneous region of the reference point. (c) The modelling results of a 2D mixture Gaussian approximation on the sail region. The pictures in (b) and (c) are shown as gray level images.

consists of N_i Gaussian clusters, then A_i can be divided into N_i elliptic regions, $\{a_{r_1}, a_{r_2}, \ldots, a_{r_{N_i}}\}$, each of which corresponds to an Gaussian cluster in $p_s(\mathbf{x}(l) \mid \omega_i)$. For each elliptic region, its color and texture features are modelled by one of Gaussian clusters in G_i^c and G_i^t, respectively. In the following, only the color modelling is illustrated. The notations and formulas for texture modelling can be obtained by replacing the subscript c by t.

Suppose a Gaussian distribution $p_c(\mathbf{c}_{\mathbf{x}(l)} \mid \theta_{c,r_i}, \omega_i)$ is used to approximate the color (texture) feature distribution in an elliptic region a_{r_i}. Let $\mathbf{c}_{\mathbf{x}(l)}$ be a D_c-dimensional color (texture) feature vector at a pixel $\mathbf{x}(l)$, then the *visual keyword* ω_i models the color (texture) with a GMM $p_c(\mathbf{c}_{\mathbf{x}(l)} \mid \omega_i)$ comprised by N_i Gaussian clusters:

$$p_c(\mathbf{c}_{\mathbf{x}(l)} \mid \omega_i) = \sum_{r_i=1}^{N_i} P_c(\theta_{c,r_i} \mid \omega_i) p_c(\mathbf{c}_{\mathbf{x}(l)} \mid \theta_{c,r_i}, \omega_i), \tag{5.2.8}$$

where θ_{c,r_i} represents the parameter sets $\{\mu_{c,r_i}, \Sigma_{c,r_i}\}$, and $P_c(\theta_{c,r_i} \mid \omega_i)$ denotes the prior probability of the cluster r_i. Suppose $p_c(\mathbf{c}_{\mathbf{x}(l)} \mid \theta_{c,r_i}, \omega_i)$ is a Gaussian

distribution,

$$p_c(\mathbf{c}_{\mathbf{x}(l)} \mid \theta_{c,r_i}, \omega_i) = \frac{\exp\left\{-\frac{1}{2}(\mathbf{c}_{\mathbf{x}(l)} - \mu_{c,r_i})^T \Sigma_{c,r_i}^{-1} (\mathbf{c}_{\mathbf{x}(l)} - \mu_{c,r_i})\right\}}{(2\pi)^{D_c/2} |\Sigma_{c,r_i}|^{1/2}}. \quad (5.2.9)$$

In order to estimate parameters of Gaussian distribution $p_c(\mathbf{c}_{\mathbf{x}(l)} \mid \theta_{c,r_i}, \omega_i)$, the following formula are applied.

$$\mu_{c,r_i} = \frac{\sum_{\mathbf{x}(l) \in a_{r_i}} \mathbf{c}_{\mathbf{x}(l)}}{N(a_{r_i})} \quad (5.2.10)$$

$$\Sigma_{c,r_i} = \frac{\sum_{\mathbf{x}(l) \in a_{r_i}} (\mathbf{c}_{\mathbf{x}(l)} - \mu_{c,r_i})(\mathbf{c}_{\mathbf{x}(l)} - \mu_{c,r_i})^T}{N(a_{r_i}) - 1}, \quad (5.2.11)$$

where $N(a_{r_i})$ denotes the number of pixels in the elliptic region a_{r_i}.

After modelling the spatial, color, and texture of an elliptic region a_{r_i}, $p_s(\mathbf{x}(l) \mid \theta_{s,r_i}, \omega_i)$, $p_c(\mathbf{c}_{\mathbf{x}(l)} \mid \theta_{c,r_i}, \omega_i)$, and $p_t(\mathbf{t}_{\mathbf{x}(l)} \mid \theta_{c,r_i}, \omega_i)$ can be merged into a Gaussian cluster as

$$\begin{aligned} p(\mathbf{z}(l) \mid \theta_{r_i}, \omega_i) &= p_s(\mathbf{x}(l) \mid \theta_{s,r_i}, \omega_i) p_c(\mathbf{c}_{\mathbf{x}(l)} \mid \theta_{c,r_i}, \omega_i) \\ &\times p_t(\mathbf{t}_{\mathbf{x}(l)} \mid \theta_{c,r_i}, \omega_i), \end{aligned} \quad (5.2.12)$$

where $\mathbf{z}(l) = (\mathbf{x}(l), \mathbf{c}_{\mathbf{x}(l)}, \mathbf{t}_{\mathbf{x}(l)})^T$, and $\theta_{r_i} = (\theta_{s,r_i}, \theta_{c,r_i}, \theta_{t,r_i})$. Then, for the homogenous region A_i, the *visual keyword* ω_i formulates its spatial, color, and texture features by a uniformed GMM:

$$p(\mathbf{z}(l) \mid \omega_i) = \sum_{r_i=1}^{N_i} P(\theta_{r_i} \mid \omega_i) p(\mathbf{z}(l) \mid \theta_{r_i}, \omega_i), \quad (5.2.13)$$

where N_i is the number of Gaussian clusters, and $P(\theta_{r_i} \mid \omega_i)$ is the prior probability of clusters r_i.

Since a *visual keyword* is in the form of mixture Gaussian distribution, its difference from the other one can be measured by (3.1.9), which is described in Section 3.1. While the spatial relation of regions is concerned, the *visual string* is generated and presented in the following section.

5.2.4 Visual String Generation

The *visual string* can be considered as an extension of the *visual keyword* for the image querying. In addition to only using key regions, the spatial relation of regions plays an important rules for image indexing and retrieving. In order to describe the spatial relation of each region, the *visual string* is proposed and defined as follows.

Definition 5.2.2 (Visual String). For a given image I containing a set of *visual keywords* $\Omega = \{\omega_i : i = 1, 2, \ldots, N\}$, a *visual string* is a 2D sequence $\mathbb{S} = (S^h, S^v)$, where S^h and S^v are sequences of the *visual keywords* to indicate the spatial order of each *visual keyword* along the horizontal and vertical directions, respectively.

By using the *visual string* to index an image, computing the similarity or dissimilarity between a desired image and a tested image becomes as simple as a string matching task. Constructing a *visual string* from *visual keywords* can be implemented in a similar manner as proposed in [53]. One simple example is shown in Fig.5.7. There are two *visual keywords* ω_1 and ω_2 in the figure, and the *visual string* $\mathbb{S} = (S^h, S^v)$ is composed of $S^h = \omega_1\omega_2\omega_1\omega_2$ and $S^v = \omega_1\omega_2\omega_2\omega_1$. The procedure to construct the *visual string* is shown in Fig. 5.8.

Difference measure between visual strings

When the query and testing *visual strings* is compared, several *visual keywords* have to selected from the testing *visual string* to match the *visual keywords* in the query *visual string*. The spatial relations of the selected *visual keywords*

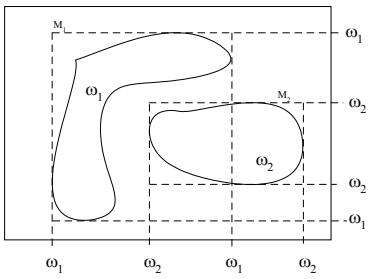


Figure 5.7: An example of constructing the *visual string* from two *visual keywords* ω_1 and ω_2. Rectangles M_1 and M_2 are the minimum enclosing rectangles of ω_1 and ω_2, respectively. ω_1 and ω_2 are the projecting points of he boundary, i.e., M_1 and M_2 on the x-axis and y-axis, respectively. According to the spatial order of the projection points of M_1 and M_2 on the x-axis and y-axis, the *visual string* $\mathbb{S} = (S^h, S^v)$, where $S^h = \omega_1\omega_2\omega_1\omega_2$ and $S^v = \omega_1\omega_2\omega_2\omega_1$

and of the *visual keywords* in the query *visual string* have to be similar to each other. An example of the *visual string* comparison is shown in Fig. 5.9. In order to compare the *visual string* $\mathbb{S}_q = (\omega_1^q\omega_2^q, \omega_1^q\omega_2^q)$ shown in Fig. 5.9(b), we have to select ω_3^t and ω_2^t from the *visual string* $\mathbb{S}_t = (\omega_1^t\omega_3^t\omega_2^t, \omega_1^t\omega_2^t\omega_3^t)$ shown in Fig. 5.9(b).

A recursive computing strategy is proposed to select the *visual keywords* from a testing *visual string*. Suppose a *visual string* $\mathbb{S} = (S^h, S^v)$ represents the spatial relation of M *visual keywords* $\Omega = \{\omega_i : i = 1, 2, \ldots, M\}$. In order to consider the horizontal and vertical spatial relations at the same time while comparing two *visual strings*, we transfer $\mathbb{S}$ into a sequence $\mathbb{U} = u_1u_2\cdots u_m\cdots u_M$, where u_m is a pair (ω_m, v_m), ω_m is the m^{th} *visual keyword* in S^h, and v_m is the position of ω_m in S^v. The position of an element u_m in $\mathbb{U}$ indicates the horizontal order, and the v_m in the element u_m presents the vertical order of the *visual keyword* ω_m in $\mathbb{S}$.

Visual String Creation procedure

Notations:

- $\Omega = \{\omega_i : i = 1, 2, \ldots, N\}$: an input *visual keyword* set
- $\mathbb{S} = (S^h, S^v)$: the output *visual string*
- M_i: the minimum enclosing rectangle (MER) of ω_i

BEGIN

1. Set $S^h = \varepsilon$ and $S^v = \varepsilon$, where ε is an empty string.
2. MER creation:

   For each *visual keyword* ω_i in Ω,

   find the M_i of ω_i;
3. S^h creation:

   Project all the MERs along the x-axis of the image.

   Scan the x-axis from left to right.

   If there is a left or right boundary of the MER M_i,

   set $S^h = S^h\omega_i$.
4. S^v creation:

   Project all the MERs along the y-axis of the image.

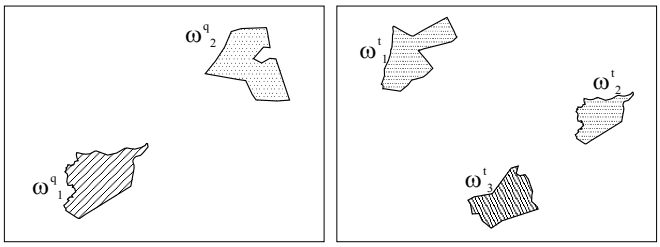
   Scan the y-axis from top to bottom.

   If there is a top or bottom boundary of the MER M_i,

   set $S^v = S^v\omega_i$.

END

Figure 5.8: The procedure to construct the *visual string* from a set of *visual keywords*



(a) (b)

Figure 5.9: An example of the *visual keyword* selection when two *visual strings* are compared. (a) the query image contains the *visual strings* $\mathbb{S}_q = (\omega_1^q\omega_2^q, \omega_1^q\omega_2^q)$ (b) the testing image contains the *visual strings* $\mathbb{S}_t = (\omega_1^t\omega_3^t\omega_2^t, \omega_1^t\omega_2^t\omega_3^t)$. When $\mathbb{S}_t$ is compared to $\mathbb{S}_q$, the *visual keywords* ω_3^t and ω_2^t are selected from $\mathbb{S}_t$.

Suppose $\mathbb{S}_q = (S_q^h, S_q^v)$ is a query *visual string* containing M *visual keywords*, and $\mathbb{S}_t = (S_t^h, S_t^v)$ is a testing *visual strings* containing N *visual keywords*. Then, the *visual strings* $\mathbb{S}_q$ and $\mathbb{S}_t$ can be represented by $\mathbb{U}^q = u_1^q u_2^q \cdots u_M^q$ and $\mathbb{U}^t = u_1^t u_2^t \cdots u_N^t$, respectively. In order to compare the query and testing *visual strings*, an element set $U_{I_M}^t(N) = \{u_{i_1}^t u_{i_2}^t \cdots u_{i_m}^t \cdots u_{i_M}^t\}$ is selected from $\mathbb{U}^t$, where $I_M = \{i_1, i_2, \cdots, i_m, \cdots, i_M\}$ is an index set of $\mathbb{U}^t$, and $u_{i_m}^t$ is an element matching with u_m^q in $\mathbb{U}^q$. In order to make sure that the spatial relations of the selected set $U_{I_M}^t(N)$ is similar as of $\mathbb{U}^q$, $U_{I_M}^t(N)$ has to satisfy (1) horizontal constraint: I_M is mono increasing, and (2) vertical constraint: for the elements in $U_{I_M}^t(N)$,

$$\text{sign}(v_{i_k}^t - v_{i_l}^t) = \text{sign}(v_k^q - v_l^q), \forall i_k, i_l \in I_M \tag{5.2.14}$$

Let $\mathbb{U}^q(m)$ and $\mathbb{U}^t(n)$ denote the first m and n elements in $\mathbb{U}^q$ and $\mathbb{U}^t$, respectively. Thus, $\mathbb{U}^q(m) = \mathbb{U}^q(m-1)u_m^q$ and $\mathbb{U}^t(n) = \mathbb{U}^t(n-1)u_n^t$. Suppose an element set $U_{I_{m-1}}^t(n-1) = \{u_{i_1}^t, u_{i_2}^t, \cdots, u_{i_k}^t, \cdots u_{i_{m-1}}^t\}$ contains $m-1$ elements selected from $\mathbb{U}^t(n-1)$ to compare with $\mathbb{U}^q(m-1)$, where $I_{m-1} =$

$\{i_1, i_2, \cdots, i_k, \cdots, i_{m-1}\}$ is an index set of $\mathbb{U}^t(n-1)$. Then, when $\mathbb{U}^q(m)$ and $\mathbb{U}^t(n)$ is compared , (5.2.14) can be computed as

$$\text{sign}(v_n^t - v_{i_k}^t) = \text{sign}(v_m^q - v_k^q), \forall k \in I_{m-1}, \tag{5.2.15}$$

and $U_{I_M}^t(N) = \{u_{i_1}^t u_{i_2}^t \cdots u_{i_M}^t\}$ is selected from $\mathbb{U}^t$ through the following recursive equation.

- $U_{I_m}^t(n) = \varepsilon$ for $m = 0$, i.e., the query string contains no elements.
- $U_{I_m}^t(n) = \phi$ for $n < m$.
- $U_{I_m}^t(n) = \begin{cases} U_{I_{m-1}}^t(n-1)u_n^t, & \text{if (5.2.15) is satisfied,} \\ U_{I_m}^t(n-1), & \text{otherwise.} \end{cases}$

On the first case, the query *visual string* contains no elements; therefore, we select zero element from $\mathbb{U}^t(n)$, i.e., $U_{I_0}^t(n) = \varepsilon, \text{for all } n$. On the second case, the length of the testing *visual string* is too short to select enough elements to compare with the query *visual string*. Hence, $U_{I_m}^t(n)$ is not exist, and $U_{I_m}^t(n) = \phi$. On the last case, based on $U_{I_m}^t(n-1)$ and $U_{I_{m-1}}^t(n-1)$, according to the newly attached element u_n^t of $\mathbb{U}^t(n)$, the computing of $U_{I_m}^t(n)$ may be divided into two cases: (A) u_n^t is selected to compare with the last element u_n^q in $\mathbb{U}^q(m)$, and therefore $U_{I_m}^t(n) = U_{I_{m-1}}^t(n-1)u_n^t$; (B) u_n^t is not selected, and then $U_{I_m}^t(n)$ is equal to the previously selected set $U_{I_m}^t(n-1)$. The case A occurs if (5.2.15) is satisfied; otherwise, the case B occurs.

After the *visual keyword* and *visual string* are generated, they are input to the *Visual Keyword based Retrieval Module* and the *Visual String based Retrieval Module*, respectively. Since the *visual keyword* is in a form of the mixture Gaussian distribution, the *Visual Keyword based Retrieval Module* is implemented

based on GPDNN shown in Section 3. GPDNN is built up to represent a query image, where each subnet corresponds to one of the *visual keywords* in the query image. In order to train a GPDNN to represent the designated image, we can select several *visual keywords* from different images as the training examples. Then, the learning rule shown in Section 3.2 is adopt to train the *Visual Keyword based Retrieval Module*. While a testing image is compared, its *visual keywords* are input to the *Visual Keyword based Retrieval Module* sequentially. Then, the input *visual keywords* are classified to their corresponding query *visual keyword* categories, and their distances are summed as the dissimilarity between the query and testing images.

Although GPDNN can measure the dissimilarity between two *visual keywords*, it is unable to present the spatial relation of the *visual keywords*. In order to compute and model the *visual string*, a *Visual String based Retrieval Module* is proposed in the following section.

5.2.5 Visual String based Retrieval Module

The schematic diagram of a *Visual String based Retrieval Module* (VSRM) for a query *visual string* $\mathbb{U}^q$ is shown in Fig. 5.10. The matching system is composed of M submodules, each of which corresponds to an element in $\mathbb{U}^q$. The submodules are sequentially connected according to the orders of the element appeared in $\mathbb{U}^q$. Each submodule contains three components: (1) a VKNN(m), abbreviated from *visual keyword neural network*, adopts the same structure as the subnet shown in Fig. 3.1 to compute the difference between the *visual keyword* ω_m^q and a testing *visual keyword*, (2) a *vertical spatial comparator*, which

checks the similarity of the vertical spatial relations of the *visual keywords*, and (3) a storage, which saves the intermediate computing results. The details of the recursive computation is described as follows.

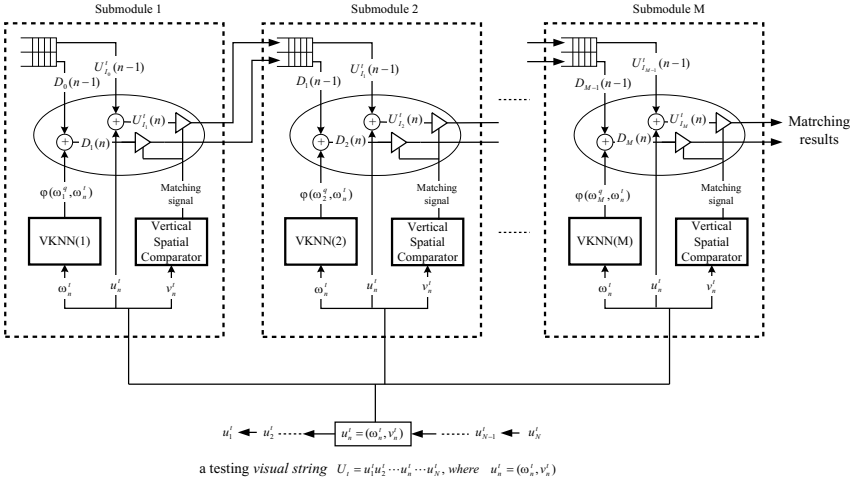


Figure 5.10: The schematic diagram of the *Visual String based Retrieval Module* (VSRM) for a *visual string* $\mathbb{U}^q = u_1^q u_2^q \cdots u_M^q$. The VSRM consists of M submodules, sequentially connected depending on the order of the elements appeared in $\mathbb{U}^q$. Each of submodules contains (1) a VKNN(m), abbreviated from *visual keyword neural network*, adopts the same structure as the subnet shown in Fig. 3.1 to compute the difference between the *visual keyword* ω_m^q and a testing *visual keyword*, (2) a *vertical spatial comparator*, which checks the similarity of the vertical spatial relations of the *visual keywords*, and (3) a storage, which saves the intermediate computing results.

When a tested *visual string* $\mathbb{U}^t = u_1^t u_2^t \cdots u_N^t$ is given to the VSRM, the elements in $\mathbb{U}^t$ are sequentially input to each submodule. Suppose an element $u_n^t = (\omega_n^t, v_n^t)$ is input to the m^{th} submodule. Then, the VKNN(m) measures the difference between ω_n^t and ω_m^q, and the *vertical spatial comparator* performs (5.2.15) to check whether the spatial relations of the *visual keywords* in $\mathbb{U}^q(m)$

and in $\mathbb{U}^q(n)$ are the same. Suppose the last intermediate computing results in the storage of the m^{th} submodule are the element set $U^t_{I_{m-1}}(n-1)$, which is selected from $\mathbb{U}^t(n-1)$, and the similarity degree between the $\mathbb{U}^q(m-1)$ and $\mathbb{U}^t(n-1)$, which is denoted as $D_{m-1}(n-1)$. Then, the new intermediate computing results, $D_m(n)$ and $U^t_{I_m}(n)$, are computed by summing the output of VKNN(m) and $D_{m-1}(n-1)$ and by appending v^t_n to $U^t_{I_{m-1}}(n-1)$, respectively. If the *vertical spatial comparator* indicates that the vertical spatial relations are similar, the new intermediate computing results are feed forward to the next submodule. Otherwise, the new intermediate computing results are aborted. When the last *visual keyword* u^t_N is input, the output of the M^{th} submodule is the element set $U^t_{I_M}(N)$ whose spatial relation is similar to $\mathbb{U}^q$.

5.3 Experiments

In this section, two experiments of image query and retrieval will be discussed. In order to measure the image retrieving performance of the *visual keyword* and *visual string* based matching system, we have implemented a *Neural Networks based Image Retrieval System* (NNIRS) at "http:// 140.113.216.78/ ImageQuerySystem". The database for training and testing of NNIRS is from the COREL Gallery $1,000,000$, which contains about 60000 general purpose color images. In the first part of experiment (see section 5.3.1), the retrieving performance of VKRM is compared with some leading algorithms [3]. On considering the spatial relation of the *visual keywords* in an image, VSRM is used in the second part of experiment (see section 5.3.2).

As shown in Section 5.2.3, a *visual keyword* represents a homogenous region which is expanded from a reference point assigned by a user. For each image in the dataset, its *visual keywords* are generated automatically. The first region is produced by randomly place a reference points in the image, and uses the expanding method (see Section 5.2.3) to produce a region which covers a homogenous color/texture area. Then, the second region can be produced again by placing another reference point on the area which is not in the previously partitioned region(s). This process continues until the whole image is covered by the homogenous regions. At this stage, *visual keywords* of the image are then generated, and their corresponding *visual string* can be produced by using the method introduced in Section 5.2.4. In the following section, the image retrieving performance of the *visual keyword* is presented.

5.3.1 Visual keyword evaluation

To qualitatively evaluate the accuracy of NNIRS based on the *visual keyword* method, I picked four query images with different semantics, namely horses, elephants, flowers, and people. For each query example, the precision of the query results depending on the relevance of the image semantics is examined. The retrieval results of these four examples are illustrated in Fig. 5.11. For each block of images, the query image is on the upper-left corner, and the rest images are the top 20 query results. As we can see, most of query results match with the query images in the cases of horse or elephant.

In order to evaluate the performance of the *visual keyword* query, 10 categories, each of which contains 100 images, are selected [3] from the COREL Gallery $1,000,000$. These categories are *Africa*, *Beach*, *Building*, *Buses*, *Dinosaurs*, *Elephants*, *Flowers*, *Horses*, *Mountains*, and *Food*. The mean and variance of the precision rate are computed for each category. Give a query image q belonging to a category C, the first N retrievals of the NNIRS contains n_q correct candidates, and then the precision rate for the query image q are defined as

$$Precision_q = \frac{n_q}{N},$$

and the mean of the precision rate of the category C is computed as

$$\overline{P}_C = \frac{1}{W} \sum_{q \in C} Precision_q,$$

where W is the number of the images in the category C. The mean and variance and the average rank, are also used. Let r_q^i is the rank (position) of image i in

the retrieved images, and the average rank for the query image q are defined as

$$Rank_q = \frac{1}{W} \sum_{i \in C} r_q^i.$$

Then, for a category C, its mean rank is computed as

$$\overline{R}_C = \frac{1}{W} \sum_{q \in C} Rank_q.$$

The retrieving performance of the *visual keyword* method is compared to the IRM [3] and UFM [4] methods. The average precision and average rank for each category of the IRM [3] and the *visual keyword* are shown in Table 5.1.

As we can see that the retrieving performance by the *visual keyword* method is 49.4%, which is comparable to 46.8% of the IRM method. Here, the rank and precision performance for each category of the UFM method are not shown since their numerical results are not available in [4]. By comparing the overall average precision for 10 categories for the UFM method shown in [4], the *visual keyword* method is still comparable to 47.7% of the UFM method.

This experimental result indicates that without considering the local content of the image, i.e., query by all the *visual keywords* in the image, the retrieving performance of NNIRS is comparable to (and even slightly better than) that of the antecedent research. It is emphasis that the content of the image is abundant. Thus, the unexpected retrieval result may be obtained when including the uninterested or unconcerned *visual keywords* in the query.

An example is shown in Fig. 5.12, where (a) is the query image assigned to the *Mountains* category. With respect to the designated image of *Mountains*, some *visual keywords* in (a) are unconcerned, such as the river, tree, horse, etc. The retrieval result of query by all the *visual keywords* in the image is shown

Table 5.1: Performance of two different retrieving methods, *visual keyword* (VK) and IRM, on the different categories of images. The rank and precision performance of VK were presented by its mean/variance values.

	precision		rank	
category	VK	IRM	VK	IRM
Africa	0.532/0.15	0.475	148.62/54.67	178.2
Beach	0.350/0.13	0.325	267.52/94.12	242.1
Building	0.277/0.10	0.33	299.73/74.51	261.8
Buses	0.550/0.19	0.36	137.05/76.94	260.73
Dinosaurs	0.932/0.05	0.981	52.67/4.66	49.7
Elephants	0.447/0.10	0.4	197.08/49.47	197.7
Flowers	0.505/0.14	0.402	199.00/84.41	298.4
Horses	0.684/0.11	0.719	102.05/43.56	92.5
Mountains	0.270/0.10	0.342	324.44/83.09	230.4
Food	0.390/0.13	0.34	234.24/80.06	271.4
Average	0.494/0.19	0.468/0.22	196.24/83.03	208.29/80.91

in (b), and whose precision rate is 15%. As we can see, some unexpected images related to the unconcerned *visual keywords* are retrieved. With respect to the interested *visual keywords* (illustrated as a union of the elliptic shapes in (a)), the retrieval result is shown in (c), and the precision rate of the retrieval result is enhanced from 15% to 32%. More examples can be found at "http:// 140.113.216.78/ NNIRS/ results". Hence, the retrieving performance can be improved by selecting the interested *visual keywords* corresponding to the designated image to query the system.

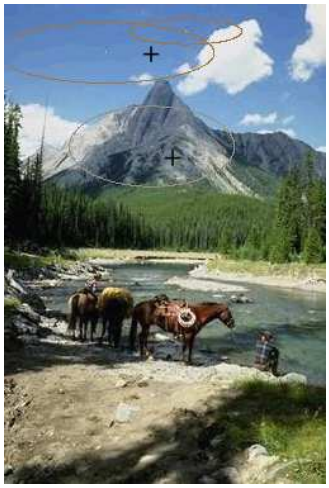
(a) Horses: 19 matches out of 20

(b) Elephants: 19 matches out of 20

(c) Flowers: 20 matches out of 20

(d) People: 14 matches out of 20

Figure 5.11: Query results by using the VKRM in the NNIRS. For each block of images, the query image is on the upper-left corner. There are two numbers below each image. From left to right they are: the ID of the image in the database and the value of the *visual keyword* measure between the query image and the matched image.



(a)

(b)

(c)

Figure 5.12: The retrieval results of query by all the *visual keywords* and by the interested *visual keywords*. (a) shows the query image and the interested *visual keyword*, illustrated as a union of the elliptic shapes. The relevant images are shown in (b) with respect to all the *visual keywords*, and (c) with respect to the interested *visual keywords* in (a). The images are shown in order of the similarity from top to bottom and left to right.

5.3.2 Visual string evaluation

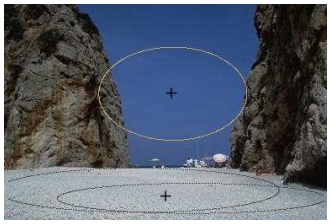
The *visual string* evaluation is different from the *visual keyword* evaluation since the spatial relation of the *visual keywords* is involved in the *visual string*. Although each of 10 categories mentioned above is formed by images with the same semantics, the spatial relation of regions in the image is not concerned in each category. Thus, the average precision and average rank for each of 10 category are not suitable to evaluate the retrieving performance of the *visual string* method.

Seven query examples are selected for the *visual string* evaluation. For each image shown in Fig. 5.13, a *visual string* is generated to represent the spatial relation of the selected *visual keywords*, shown as a union of the elliptic shapes. The description of the corresponding *visual string* is shown below each image in Fig. 5.13.

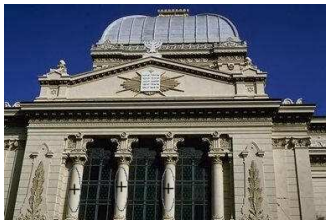
The retrieval results of query examples in Fig. 5.13 are shown from Fig. 5.14 to Fig. 5.20. In these figures, part (a) and (b) illustrate the relevant feedbacks of the NNIRS with respect to *visual keywords* and *visual string*. For each query example, the precision of the query results depending on the relevance of the image semantics is examined. It is admitted that the relevance of the *visual string* depends on the standpoint of the user. Thus our relevance criteria, specified from Fig. 5.14 to Fig. 5.20, may be quite different from those used by a user of the system. In each retrieval case, the top 20 images are shown in order of the similarity from top to bottom and left to right. More retrieval results can be found at "http:// 140.113.216.78/ NNIRS/ results".

Fig. 5.21 illustrates the number of interested images depended on the

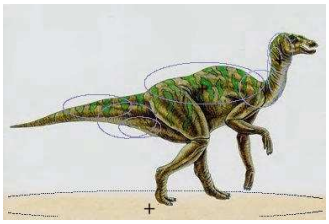
number of retrieved images with respect to the different query examples. For each query example, the retrieval performance based on the *visual keyword* query and the *visual string* query are depicted as the solid line and dash line, respectively. As we can see, in each query example, the dash line is on the left-upper side of the solid one. That means, the interested images are in the fore part of the retrieved images when the *visual string* is used to query the system. Hence, query by *visual string* can significantly improve the hit rate of finding the interested images while the spatial relation of *visual keywords* is concerned.



(a) Blue sky and white sand beach



(b) Stelas



(c) Dinosaur stands on the sienna ground.

(d) Elephant stands on the sand in front of trees.

(e) Flower with pink and white petal margins

(f) Brown horse stands on the left side of the white horse in the grass.

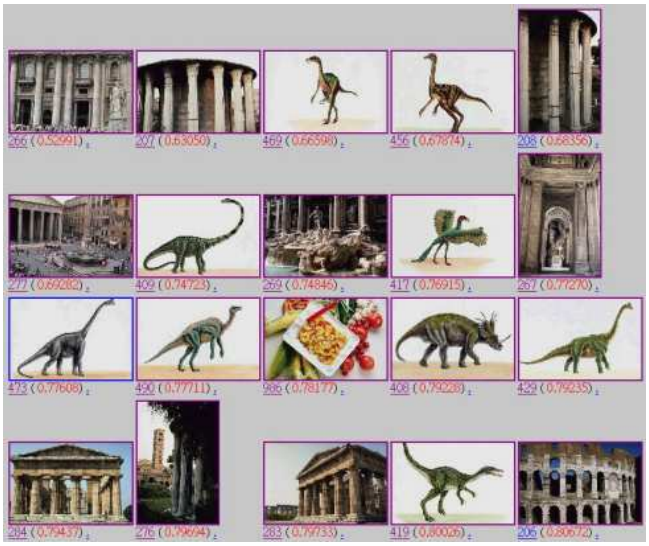
(g) Blue sky and mountain with snow

Figure 5.13: Seven query images are selected for the *visual string* evaluation experiments. In each image, several *visual keywords* are generated using the proposed methods in Section 5.2.3. Each *visual keyword* and its corresponding reference point are shown as a union of the elliptic shapes and the "+" marks, respectively. A *visual string* is generated to represent the spatial relation of the selected *visual keywords* in an image as described in Section 5.2.4.

(a) 7 matches out of 20

(b) 11 matches out of 20

Figure 5.14: The retrieval results for the query *visual string* shown in Fig. 5.13(a). The relevant images are shown in (a) with respect to only the *visual keywords*, and (b) with respect to the *visual string*. The images are shown in order of the similarity from top to bottom and left to right.

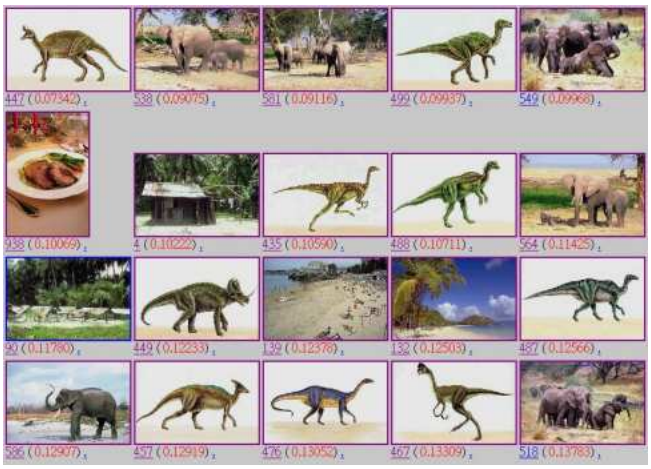


(a) 10 matches out of 20

(b) 11 matches out of 20

Figure 5.15: The retrieval results for the query *visual string* shown in Fig. 5.13(b). The relevant images are shown in (a) with respect to only the *visual keywords*, and (b) with respect to the *visual string*. The images are shown in order of the similarity from top to bottom and left to right.

(a) 0 matches out of 20



(b) 9 matches out of 20

Figure 5.16: The retrieval results for the query *visual string* shown in Fig. 5.13(c). The relevant images are shown in (a) with respect to only the *visual keywords*, and (b) with respect to the *visual string*. The images are shown in order of the similarity from top to bottom and left to right.

(a) 8 matches out of 20

(b) 9 matches out of 20

Figure 5.17: The retrieval results for the query *visual string* shown in Fig. 5.13(d). The relevant images are shown in (a) with respect to only the *visual keywords*, and (b) with respect to the *visual string*. The images are shown in order of the similarity from top to bottom and left to right.

(a) 8 matches out of 20

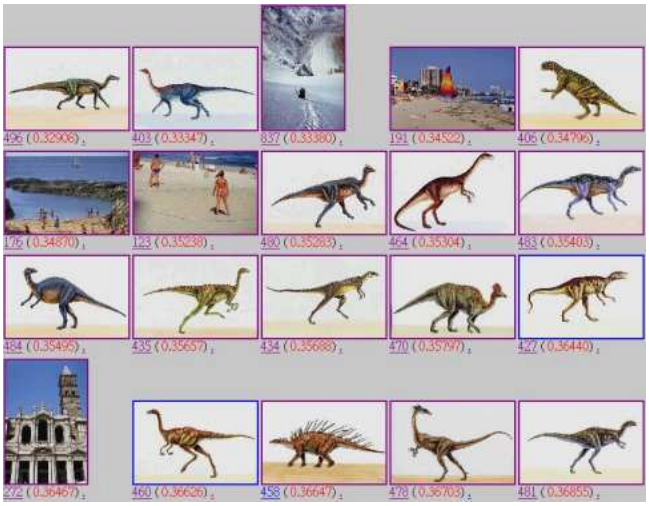
(b) 13 matches out of 20

Figure 5.18: The retrieval results for the query *visual string* shown in Fig. 5.13(e). The relevant images are shown in (a) with respect to only the *visual keywords*, and (b) with respect to the *visual string*. The images are shown in order of the similarity from top to bottom and left to right.

(a) 3 matches out of 20

(b) 9 matches out of 20

Figure 5.19: The retrieval results for the query *visual string* shown in Fig. 5.13(f). The relevant images are shown in (a) with respect to only the *visual keywords*, and (b) with respect to the *visual string*. The images are shown in order of the similarity from top to bottom and left to right.



(a) 0 matches out of 20

(b) 10 matches out of 20

Figure 5.20: The retrieval results for the query *visual string* shown in Fig. 5.13(g). The relevant images are shown in (a) with respect to only the *visual keywords*, and (b) with respect to the *visual string*. The images are shown in order of the similarity from top to bottom and left to right.

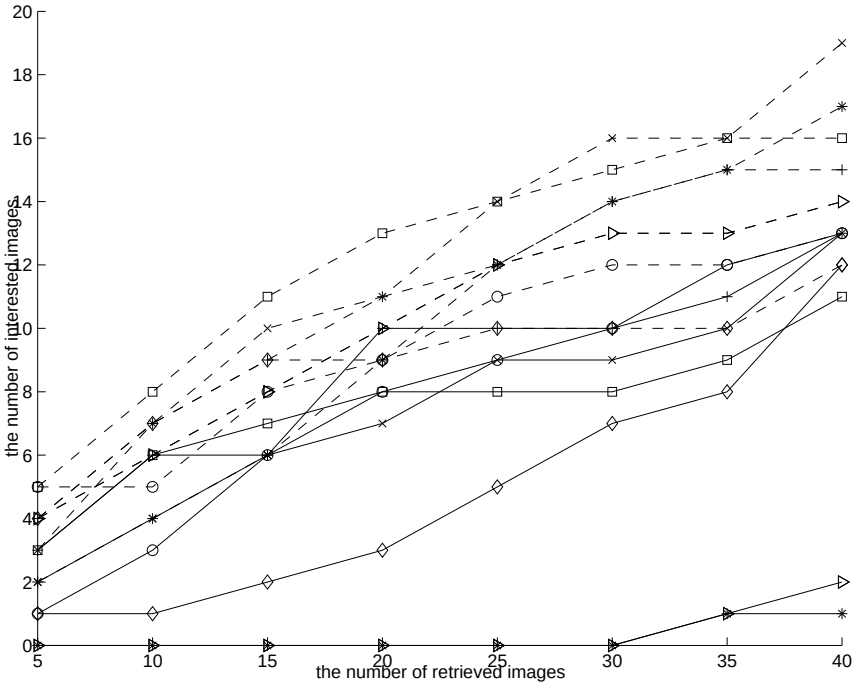


Figure 5.21: The retrieving performance with respect to the query examples shown in Fig. 5.13 from (a) to (g) are depicted as '+', '×', '∗', '○', '□', '◇', and '▷', respectively. In each query example, the dash and solid lines are used to shown the retrieval performance based on the *visual keyword* and *visual string*, respectively.

Chapter 6

Conclusions and Future work

In this thesis, *Self-growing Probabilistic Decision based Neural Network* (SPDNN) is first introduced. The ISLUG training scheme is proposed to tune the decision boundary of SPDNN to improve the classification accuracy. Furthermore, the *Generalized Probabilistic Decision based Neural Network* (GPDNN), a generalized version of SPDNN, is proposed to handle the general case that the data are in the form of the distributions instead of the numerical quantities. In order to verify the accuracy of the proposed SPDNN and GPDNN, the applications of handwritten character recognition and content-based image retrieval are involved.

In the handwritten character recognition application, an SPDNN-based Handwritten Chinese major hybrid character recognition system is developed. This recognition system performs pre-classification, character recognition, and personal adaptation. The experiment results indicate the follows: (1) the mixed Gaussian-based discrimination function permits SPDNN to learn the character decision boundary precisely, and (2) the self-growing rules allow a small number of Gaussian clusters to be sufficient to represent the character image distribu-

tion.

In the content-based image retrieval application, a *Neural Networks based Image Retrieval System* (NNIRS) is developed based on GPDNN and is implemented at "http:// 140.113.216.78/ imagequerysystem". The database of NNIRS is from the COREL Gallery $1,000,000$, which contains about 60000 general purpose color images. Two novel image representation concepts for CBIR are proposed: (1) the *visual keyword* and (2) the *visual string*. The *visual keyword* describes the visual characteristics (color, texture, and shape features) of a homogenous region, while the *visual string* represents the spatial relationship of the regions in an image. The experiment results show the follows: (1) the retrieving performance by *visual keyword* method is 49.4%, which is comparable the 46.8% of IRM method [3] and 47.7% of the UFM method [4], and (2) query by *visual string* can significantly improve the hit rate of finding the interested images while the spatial relation of *visual keywords* is concerned.

The experiment results shown that SPDNN and GPDNN are suitable for handwritten character recognition and content-based image retrieval, respectively. The following description provides some further research topics in the future.

NNIRS over Internet

Since there is a a huge thesaurus on the Internet, the ability of the NNIRS can be extended in the future to mine the World Wide Web to find the interested images. The NNIRS over Internet can be designed as shown in Fig. 6.1. In this system, the robot travels the Internet periodically in the background. while a

new image is found, the robot will sent it back to the system for image indexing. Then, the URL of the new image is recorded in the local database. The user can query the NNIRS to retrieve the interesting images, and download the images via URL.

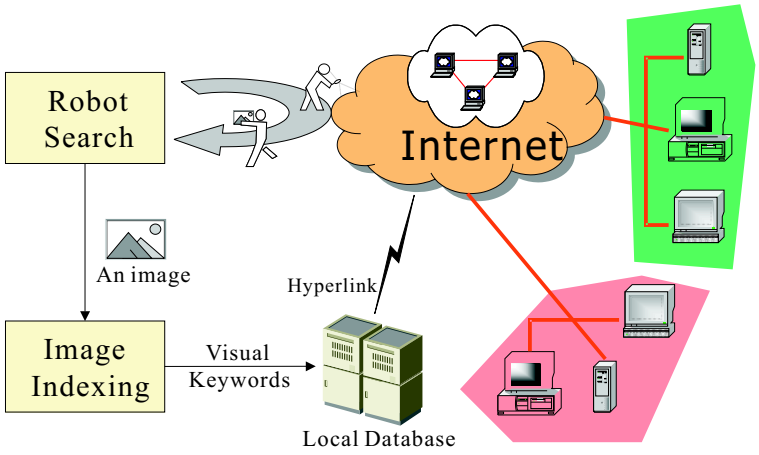


Figure 6.1: The flowchart of the NNIRS over Internet. In the robot search, images in the Internet will be checked periodically. While a new image is found, it will be indexed by *visual keyword* and *visual string*. The user can query the NNIRS to retrieve the interesting images, and download the images via the images' URL.

Structured Visual Keyword Search

As discussed in this thesis, the *visual keywords* in database are sequentially compared to the query *visual keywords*. Thus, while the database is grown up, such as the case of NNIRS over Internet, the speed of the image retrieval will be slow down. Hence, to structuralize the *visual keywords* in the database for further query or browsing will be the next research topic to improve the query speed of NNIRS over Internet.

Content-based Video retrieval

In the recent years, more and more researches focus on the content-based video retrieval (CBVR). How to define the content of a video is a kernel problem in CBVR. The video can be considered as a sequence of the images. Since an image can be indexed by the proposed *visual keywords*, it is an interesting research topic to extend the *visual keyword* with the time information to represent the content of a video.

Appendix A

Proof

Lemma 3.1.1: Suppose Suppose $\mathcal{P}_a$ and $\mathcal{P}_b$ are two mixture Gaussian distributions defined in (3.1.5) and (3.1.5), respectively. The product moment of $\mathcal{P}_a$ and $\mathcal{P}_b$,

$$\mathcal{F}(\mathcal{P}_a, \mathcal{P}_b) = \sum_{r_a=1}^{R_a} \sum_{r_b=1}^{R_b} P(\theta_{r_a}) P(\theta_{r_b}) \mathcal{G}(\theta_{r_a}, \theta_{r_b}),$$

where

$$\mathcal{G}(\theta_{r_a}, \theta_{r_b}) = \frac{\exp\left\{-\frac{1}{2}\sum_{d=1}^{D} \frac{(\mu_{r_b(d)} - \mu_{r_a(d)})^2}{\sigma^2_{r_b(d)} + \sigma^2_{r_a(d)}}\right\}}{\sqrt{(2\pi)^D \prod_{d=1}^{D} (\sigma^2_{r_b(d)} + \sigma^2_{r_a(d)})}}.$$

Proof.

$$\begin{aligned}
\mathcal{F}(\mathcal{P}_a, \mathcal{P}_b) &= \int_{R^D} \mathcal{P}_a \mathcal{P}_b d\mathbf{z} \\
&= \int_{R^D} \left(\sum_{r_a=1}^{R_a} P(\theta_{r_a}) p(\mathbf{z} \mid \theta_{r_a}) \sum_{r_b=1}^{R_b} P(\theta_{r_b}) p(\mathbf{z} \mid \theta_{r_b}) \right) d\mathbf{z} \\
&= \sum_{r_a=1}^{R_a} \sum_{r_b=1}^{R_b} \left(P(\theta_{r_a}) P(\theta_{r_b}) \int_{R^D} p(\mathbf{z} \mid \theta_{r_b}) p(\mathbf{z} \mid \theta_{r_a}) d\mathbf{z} s \right).
\end{aligned}$$

Let $M(\theta_{r_b} \mid \theta_{r_a})$ be the expected value of the $p(\mathbf{z} \mid \theta_{r_b})$ under the Gaussian distribution $p(\mathbf{z} \mid \theta_{r_a})$,

$$\begin{aligned} M(\theta_{r_b} \mid \theta_{r_a}) &= \int_{R^D} p(\mathbf{z} \mid \theta_{r_b}) p(\mathbf{z} \mid \theta_{r_a}) d\mathbf{z} \\ &= \underbrace{\int \cdots \int}_{D} \{p(z_1, \cdots, z_D \mid \theta_{r_b}) \\ &\cdot \quad p(z_1, \cdots, z_D \mid \theta_{r_a})\} \, dz_1 \cdots dz_D. \end{aligned} \tag{A.0.1}$$

Since the elements in visual feature vector are mutually independent, the following joint probability densities can be expressed in product forms:

$$\begin{aligned} p(z_1, \cdots, z_D \mid \theta_{r_b}) &= \prod_{d=1}^{D} p(z_d \mid \theta_{r_b}), \text{ and} \\ p(z_1, \cdots, z_D \mid \theta_{r_a}) &= \prod_{d=1}^{D} p(z_d \mid \theta_{r_a}). \end{aligned} \tag{A.0.2}$$

Substitute (A.0.2) into (A.0.1),

$$M(\theta_{r_b} \mid \theta_{r_a}) = \prod_{d=1}^{D} M_d(\theta_{r_b} \mid \theta_{r_a}). \tag{A.0.3}$$

Each of the product terms, $M_d(\theta_{r_b} \mid \theta_{r_a})$, can be derived as follows:

$$\begin{aligned}
M_d(\theta_{r_b} \mid \theta_{r_a}) &= \int_{-\infty}^{\infty} p(z_d \mid \theta_{r_b}) p(z_d \mid \theta_{r_a}) dz_d \\
&= \frac{1}{2\pi\sigma_{r_a(d)}\sigma_{r_b(d)}} \\
&\quad \cdot \int_{-\infty}^{\infty} \exp\left\{ -\frac{1}{2} \left(\frac{(z_d - \mu_{r_a(d)})^2}{\sigma^2_{r_a(d)}} \right.\right. \\
&\quad \left.\left. + \frac{(z_d - \mu_{r_b(d)})^2}{\sigma^2_{r_a(d)}} \right) \right\} dz_d \\
&= \frac{1}{2\pi\sigma_{r_a(d)}\sigma_{r_b(d)}} \\
&\quad \cdot \int_{-\infty}^{\infty} \exp\left\{ -\frac{1}{2} \left[\left(\frac{1}{\sigma^2_{r_a(d)}} + \frac{1}{\sigma^2_{r_b(d)}} \right) z_d^2 \right.\right. \\
&\quad -2 \left(\frac{\mu_{r_a(d)}}{\sigma^2_{r_a(d)}} + \frac{\mu_{r_b(d)}}{\sigma^2_{r_b(d)}} \right) z_d \\
&\quad \left.\left. + \left(\frac{\mu^2_{r_a(d)}}{\sigma^2_{r_a(d)}} + \frac{\mu^2_{r_b(d)}}{\sigma^2_{r_b(d)}} \right) \right] \right\} dz_d.
\end{aligned}$$

Let

$$\begin{aligned}
\mathcal{A} &= \int_{-\infty}^{\infty} \exp\left\{ -\frac{1}{2} \left(\frac{1}{\sigma^2_{r_a(d)}} + \frac{1}{\sigma^2_{r_b(d)}} \right) \right. \\
&\quad \left. \cdot \left(z_d - \frac{\sigma^2_{r_b(d)}\mu_{r_a(d)} + \sigma^2_{r_a(d)}\mu_{r_b(d)}}{\sigma^2_{r_a(d)} + \sigma^2_{r_b(d)}} \right)^2 \right\} dz_d.
\end{aligned}$$

Then

$$\begin{aligned} M_d(\theta_{r_b} \mid \theta_{r_a}) &= \frac{\mathcal{A}}{2\pi\sigma_{r_a(d)}\sigma_{r_b(d)}} \\ &\quad \cdot \exp\left[-\frac{1}{2}\left(\frac{\mu^2_{r_a(d)}}{\sigma^2_{r_a(d)}} + \frac{\mu^2_{r_b(d)}}{\sigma^2_{r_b(d)}}\right)\right. \\ &\quad \left. + \frac{1}{2}\frac{\left(\frac{\mu_{r_a(d)}}{\sigma^2_{r_a(d)}} + \frac{\mu_{r_b(d)}}{\sigma^2_{r_b(d)}}\right)^2}{\left(\frac{1}{\sigma^2_{r_a(d)}} + \frac{1}{\sigma^2_{r_b(d)}}\right)}\right] \\ &= \frac{\mathcal{A}}{2\pi\sigma_{r_a(d)}\sigma_{r_b(d)}} \\ &\quad \cdot \exp\left[-\frac{1}{2}\frac{(\mu_{r_a(d)} - \mu_{r_b(d)})^2}{\sigma^2_{r_a(d)} + \sigma^2_{r_b(d)}}\right]. \end{aligned} \tag{A.0.4}$$

Let $s_d = z_d - \frac{\sigma^2_{r_b(d)}\mu_{ra(d)} + \sigma^2_{r_a(d)}\mu_{r_b(d)}}{\sigma^2_{r_a(d)} + \sigma^2_{r_b(d)}}$; thus

$$\begin{aligned} \mathcal{A}^2 &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \exp\left\{-\frac{1}{2}\left(\frac{1}{\sigma^2_{r_a(d)}} + \frac{1}{\sigma^2_{r_b(d)}}\right)\right. \\ &\quad \left. \cdot(s_d^2 + t_d^2)\right\} ds_d dt_d. \end{aligned}$$

Let $s_d = \rho_d \cos\alpha_d$ and $t_d = \rho_d \sin\alpha_d$, then

$$\begin{aligned} \mathcal{A}^2 &= \int_0^{\infty} \rho_d \int_0^{2\pi} \exp\left\{-\frac{1}{2}\left(\frac{1}{\sigma^2_{r_a(d)}} + \frac{1}{\sigma^2_{r_b(d)}}\right)\right. \\ &\quad \left. \cdot\rho_d^2\right\} d\rho_d d\alpha_d \\ &= 2\pi\left(\frac{\sigma^2_{r_a(d)}\sigma^2_{r_b(d)}}{\sigma^2_{r_a(d)} + \sigma^2_{r_b(d)}}\right). \end{aligned}$$

Hence,

$$\mathcal{A} = \sqrt{2\pi\left(\frac{\sigma^2_{r_a(d)}\sigma^2_{r_b(d)}}{\sigma^2_{r_a(d)} + \sigma^2_{r_b(d)}}\right)}. \tag{A.0.5}$$

Substitute (A.0.5) into (A.0.4);

$$M_d(\theta_{r_b} \mid \theta_{r_a}) = \frac{\exp(-\frac{1}{2}\frac{(\mu_{r_b(d)} - \mu_{r_a(d)})^2}{\sigma^2_{r_a(d)} + \sigma^2_{r_b(d)}})}{\sqrt{2\pi}\sqrt{\sigma^2_{r_a(d)} + \sigma^2_{r_b(d)}}}. \tag{A.0.6}$$

Finally, substitute (A.0.6) into (A.0.3), and

$$\begin{aligned} M(\theta_{r_b} \mid \theta_{r_a}) &= \frac{\exp(-\frac{1}{2}\sum_{d=1}^{D} \frac{(\mu_{r_b(d)}-\mu_{r_a(d)})^2}{\sigma^2_{r_a(d)}+\sigma^2_{r_b(d)}})}{\sqrt{(2\pi)^D \prod_{d=1}^{D}(\sigma^2_{r_a(d)}+\sigma^2_{r_b(d)})}} \\ &= \mathcal{G}(\theta_{r_b} \mid \theta_{r_a}). \end{aligned}$$

□

Bibliography

[1] Tze-Fen Li and Shiaw-Shian Yu, "Hand-printed chinese character recognition using the probability distribution feature," *Int. J. of Pattern Recognition and Artificial Intelligence*, vol. 8, no. 5, pp. 1241–1258, 1994.

[2] Y.H. Tseng, C.C. Kuo, and H.S. Lee, "Speeding up chinese character recognition in an automatic document reading system," *Pattern Recognition*, vol. 31, no. 11, pp. 1601–1612, 1998.

[3] James Ze Wang, Jia Li, and Gio Wiederhold, "SIMPLIcity: Semantics-sensitive integrated matching for picture LIbraries," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 9, pp. 947–963, 2001.

[4] Y. Chen and J.Z. Wang, "A region-based fuzzy feature matching approach to content-based image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 9, pp. 1252–1267, 2002.

[5] Christopher M. Bishop, *Neural Networks for Pattern Recognition*, Oxford University Press, 1995.

[6] S. Watanabe, *Pattern Recognition: Human and Mechanical*, New York: Wiley, 1985.

[7] G. Nagy, "State of the art in pattern recognition," *Proceedings of the IEEE*, vol. 56, pp. 336–362, 1992.

[8] L.N. Kanal, "Patterns in pattern recognition: 1968-1974," *IEEE Trans. Information Theory*, vol. 20.

[9] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*, New York: John Wiley and Sons, 1973.

[10] Luc Devroye, Laszlo Gyorfi, and Gabor Lugosi, *A Probabilistic Theory of Pattern Recognition*, Berlin: Springer-Verlag, 1996.

[11] Anil K. Jain, Robert P. W. Duin, and Jianchang Mao, "Statistical pattern recognition: A review," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 4–37, 2000.

[12] Tuevo Kohonen, Gyorgy Barna, and Chrisley Ronald, "Statistical pattern recognition with neural networks: benchmarking studies," in *Proceedings of the IEEE Conference on Neural Networks*, 1988, pp. 61–68.

[13] J.A. Anderson, A. Pellionisz, and E. Rosenfeld, *Neurocomputing 2: Directions for Research*, MIT Press, Cambridge Mass., 1990.

[14] B. D. Ripley, *Networks and Chaos X Statistical and Probabilistic Aspects*, chapter Statistical aspects of neural networks, pp. 40–123, Chapman and Hall, 1993.

[15] Shang-Hung Lin, S.Y. Kung, and L.J. Lin, "Face recognition/detection by probabilistic decision-based neural networks," *IEEE Transactions on Neural Networks*, vol. 8, no. 1, pp. 114–132, 1997.

[16] Patrick M. Kelly, Michael Cannon, and Julio E. Barros, "Efficiency issues related to probability density function comparison," in *SPIE Storage and Retrieval for Still Image and Video Databases IV*, 1996, vol. 2670, pp. 42–49.

[17] Simone Santini and Ramesh Jain, "Similarity measures," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 9, pp. 871–883, 1999.

[18] N. Vasconcelos and A. Lippman, "Probabilistic retrieval: new insights and experimental results," in *Workshop on CAIVL, CVPR '99*, Denver, 1999.

[19] D. Comaniciu, P. Meer, and D. Tyler, "Dissimilarity computation through low rank corrections," *Pattern Recognition Letters*, vol. 24, no. 1-3, pp. 227–236, 2003.

[20] S.Y. Kung and J.S. Taur, "Decision-based hierarchical neural networks with signal/image classification applications," *IEEE Transactions on Neural Networks*, vol. 6, no. 1, pp. 170–181, 1995.

[21] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of the Royal Statistical Society Series B*, vol. 39, no. 1, pp. 1–38, 1977.

[22] H.S. Baird, "Feature identification for hybrid structural/statistical pattern classification," *Computer Vision, Graphics, and Image Processing*, vol. 42, pp. 318–333, 1988.

[23] R.M. Bozinovic and S.N. Srihari, "Off-line cursive word recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 68–83, 1989.

[24] D.J. Burr, "Designing a handwriting reader," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-5, pp. 554–559, 1983.

[25] F.H. Cheng and W.H. Hsu, "Research on chinese ocr in taiwan," *Int. Journal of Pattern Recognition and Artificial Intelligence*, vol. 5, no. 1, pp. 139–164, 1991.

[26] Y. Le Cun, B. Boser, J.S. Denker, D. Henderson, R.R. Howard, W. Hubbard, and L.D. Jackel, "Handwritten zip code recognition with multi-layer networks," in *Proc. 10th Int. Conf. Pattern Recognition*, 1990, pp. 35–40.

[27] S. Mori, C.Y. Suen, and K. Yamamoto, "Historical review of ocr research and development," *Proceedings of IEEE*, vol. 80, no. 7, pp. 1029–1058, 1992.

[28] P. R. Krishnaiah, L. N. Kanal, and Eds., *Handbook of Statistics*, chapter Optical character recognition, pp. 621–649, North Holland, 1982.

[29] G. Nagy, "Chinese character recognition: A twenty-five year retrospective," in *Proc. 12th Int. Conf. Pattern Recognition*, 1988, pp. 163–167.

[30] C.Y. Suen, M. Berthod, and S. Mori, "Automatic recognition of handprinted character - the state of the art," *IEEE Proceedings*, vol. 68, pp. 469–487, 1980.

[31] N. Arica and F. T. Yarman-Vural, "Optical character recognition for cursive handwriting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 6, pp. 801–813, 2002.

[32] X. Tang and F. Lin, "Video-based handwritten character recognition," in *Proc. Intl Conf. Acoustics, Speech, and Signal Processing*, 2002, pp. 3748–3751.

[33] J.-X. Dong, C.Y. Suen, and A. Krzyzak, "High accuracy handwritten chinese character recognition using support vector machine," in *Proceedings of International Workshop on Artitifical Neural Networks on Pattern Recognition*, 2003, pp. 39–45.

[34] Cheng-Chin Chiang, Tze Cheng, and Shiaw-Shien Yu, "An iterative rule-based character segmentation method for chinese documents," in *Proc. of Int. Conf. on Chinese Computing'96*, 1996.

[35] H.C. Fu and K.P. Chiang, "Recognition of handwritten chinese characters by multi-stage neural network classifiers," in *Proc. of 1995 IEEE Int. Conf. Neural Networks*, 1995.

[36] F. Kimura, K.Takashina, S. Tsuruoka, and Y. Miyake, "Modified quadratic discrimininant functions and application to chinese character recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, no. 1, pp. 149–153, 1987.

[37] L.T. Tu, Y.S. Lin, C.P. Yeh, I.S. Shyu, J.L. Wang, K.H. Joe, and W.-W. Lin, "Recognition of hand-printed chinese characters by feature matching," in *Proc. of 1991 First National Workshop on Character Recognition*, 1991, pp. 166–175.

[38] Jonathan J. Hull, "A database for handwritten text recognition research," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 6, pp. 550–554, 1994.

[39] Ju-Wei Tai, "Some research achievements on chinese character recognition in china," *Int. J. of Pattern Recognition and Artificial Intelligence*, vol. 5, no. 1-2, pp. 199–206, 1991.

[40] Yin Xia, "Research report on interactive self-learning system of handwritten chinese characters," Tech. Rep., Department of Computer Science, QingHua University, Nov. 1989.

[41] H.C. Fu, S.C. Chuang, Y.Y. Xu, W.H. Su, and K.T. Sun, "A personal adaptive module for unconstrained handwritten chinese characters recognition," in *Proc. of the International Symposium on Multi-technology Information Processing*, 1996.

[42] Jianbo Shi and Jitendra Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 888–905, 2000.

[43] W.Y. Ma and B.S. Manjunath, "Edgeflow: a technique for boundary detection and image segmentation," *IEEE Transactions on Image Processing*, vol. 9, no. 8, pp. 1375–1388, 2000.

[44] James Ze Wang, Jia Li, Robert M. Gray, and Gio Wiederhold, "Unsupervised multiresolution segmentation for images with low depth of field," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 1, pp. 85–90, 2001.

[45] S. K. Chang, E. Jungert, and Y. Li, "Representation and retrieval of symbolic pictures using generalized 2d strings," *University of Pittsburgh*, vol. PA 15260, 1988.

[46] J. R. Smith and C.-S. Li., "Image classification and querying using composite region templates," *Journal of Computer Vision and Image Understanding*, 1999.

[47] Christos Faloutsos, Ron Barber, Myron Flickner, Jim Hafner, Wayne Niblack, Dragutin Petkovic, and William Equitz, "Efficient and effective querying by image content," *Journal of Intelligent Information Systems*, vol. 3, no. 3/4, pp. 231–262, 1994.

[48] Amarnath Gupta and Ramesh Jain, "Visual information retrieval," *Communications of the ACM*, vol. 40, no. 5, pp. 70–79, 1997.

[49] A. Pentland, R. Picard, and S. Sclaroff, "Photobook: content-based manipulation of image databases," in *Storage and Retrieval for Image, Video Databases II*, San Jose, CA, Feb. 1994, SPIE, vol. 2185.

[50] John R. Smith and Shih-Fu Chang, "Visualseek: A fully automated content-based image query system," in *ACM Multimedia*, 1996, pp. 87–98.

[51] W. Y. Ma and B. S. Manjunath, "Netra: a toolbox for navigating large image databases," in *Proc. IEEE Int'l Conf. Image Processing*, 1997, pp. 568–571.

[52] Chad Carson, Megan Thomas, Serge Belongie, Joseph M. Hellerstein, and Jitendra Malik, "Blobworld: A system for region-based image indexing and retrieval," in *Third International Conference on Visual Information Systems*, June 1999, pp. 509–516.

[53] S.Y. Lee, M.C. Yang, and J.W. Chen, "2d b-string: a spatial knowledge representation for image database systems," in *Proc. ICSC'92 Second Int. Computer Sci. Conf.*, 1992, pp. 609–915.

[54] John C. Russ, *The Image Processing Handbook*, Ron Powers, 3 edition, 1998.

[55] Pietro Perona and Jitendra Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, no. 7, pp. 629–639, July 1990.

[56] G. Wyszecki and W. Stiles, *Color Science: Concepts and Methods, Quantitative Data and Formulae*, Wiley, second edition, 1982.

[57] J. G. Daugman, "Complete discrete 2d gabor transforms by neural networks for image analysis and compression," *IEEE Transactions on ASSP*, vol. 36, pp. 1169–1179, July 1988.

[58] B. S. Manjuath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 837–842, August 1996.