國立交通大學

資訊學院　資訊學程

碩士論文

利用階層式支持向量機演算法建立應用於行動化視覺搜尋之影像字彙樹

Using Hierarchical SVM Algorithm to Construct a Vocabulary Tree for Mobile Visual Search Applications

研 究 生：蕭仁惠

指導教授：彭文孝　教授

中 華 民 國 九 十 九 年 七 月

利用階層式支持向量機演算法建立應用於行動化視覺搜尋之影像字彙樹

Using Hierarchical SVM Algorithm to Construct a Vocabulary Tree for Mobile Visual Search Applications

研 究 生: 蕭仁惠　　Student: Jen-Hui Hsiao
指導教授: 彭文孝 博士　　Advisor: Dr. Wen-Hsiao Peng

國 立 交 通 大 學
資訊學院 資訊學程

碩 士 論 文

A Thesis
Submitted to
College of Computer Science
National Chiao Tung University
In partial Fulfillment of the Requirements
For the Degree of
Master of Science
In
Computer Science

July 2010

Hsinchu Taiwan, Republic of China

中華民國九十九年七月

利用階層式支持向量機演算法建立應用於行動化視覺搜尋之影像字彙樹

研究生 ： 蕭仁惠　　指導教授:彭文孝 博士

國立交通大學 資訊學院 資訊學程碩士班

摘要

在 Mobile Visual Search 的系統中，scalable recognition 是其中一個很重要的環結，在之前的研究中，Vocabulary Tree 提供了一個很好的方式，能夠有效地進行影像的辨識。但是，在 Vocabulary Tree 的方法中，使用 Hierarchical K-means 演算法將所有影像的特徵點進行分類，由於 Hierarchical K-means 演算法是利用 K-means 演算法進行階層的分類，首先使用 K-means 演算法進行分類，分類的結果會受到選擇初始中心點的影響，造成分類隨著實驗變動，使得影像辨識的困難，再者在每一層分類上，分類的數目都是小於實際分類的數目，即使選擇分類中心點是實際分類的中心點，其他類別的特徵點就很容易被均勻分布在分類中心點，也造成影像分辨不正確的結果。

分類一群未知影像的特徵點是屬於 unsupervised machine learning，但是真正想要分類是影像而不完全是影像的特徵點，特徵點屬於那個影像是一個有效而且有用的資訊；於是在建立 Vocabulary Tree 的每一層分類時，本論文加入了特徵點的影像資訊，利用影像中心點，先將影像使用 K-means++演算法進行初步分類，再將初步分類的結果，使用 supervised machine learning 的 Support Vector Machine 演算法做真正特徵點的分類，使得影像的特徵點能夠更有效地分類，也提升了影像辨識的正確率。

Using Hierarchical SVM Algorithm to Construct a Vocabulary Tree for Mobile Visual Search Applications

Student: Jen-Hui Hsiao Advisor: Wen-Hsiao Peng

Degree program of Computer Science

National Chaio Tung University

Abstract

In the Mobile Visual Search system, scalable recognition is a very important part. On the previous study, a Vocabulary Tree provides a good way to effectively recognize image. However, in the Vocabulary Tree approach, Hierarchical K-means algorithm is used to classify all feature points of images and K-means algorithm is used at each level of classification. First, to use K-means algorithm, classification result is impacted by selection of initial class points and is changed with the experiment. Second, at each level classification, the number of classification is less than the number of actual classification, even if the classification result are actually the center of some images, feature points of other images can easily be distributed in the classification centers to cause images are not matching.

In this paper, K-means algorithm is not used to classify feature points of images at each level of a Vocabulary Tree. Centers of feature points on each image are used to group images at the beginning and then Support Vector Machine algorithm is used to classify feature points on each group of image. According to theory and experiment, Hierarchical SVM has outstanding result on matching rate and complexity of scalable recognition.

誌謝

回顧幾年的研究所生涯，首先，我要感謝我的指導教授—彭文孝 博士，給予我於學問研究上的指導。彭老師實事求是的精神，與深入剖析問題的態度，以及追根究柢與契而不捨的指導方式，已經成為我在學習與研究路上的典範與楷模。

有榮幸進入多媒體架構與處理實驗室，可以在這個優良的環境下不斷學習，又有熱心與親切的實驗室成員們的切磋與討論，是我在學士後時代最充實的時光。感謝我的學長姐們，引領我進入研究生的階段；感謝我的同學，不論是課業上或研究上，他們總是可以一針見血地提出問題的核心要點，給予最直接協助；感謝我的學弟吳思賢、王澤瑋、陳孟傑、與吳崇豪，在最後這一年內，給予許多無私的協助。

最後，我要感謝我的家人，在爭取碩士學位的路上，給予百分之百的支持，讓我免去許多後顧之憂與煩擾。感謝我的太太—黃靖芸，這幾年來辛苦地陪伴、體諒與關心。感謝我的上司、同事、與朋友們，是你們的支持，使我有信心取得這個學位，謝謝你們。

目錄

圖目錄

1. 序論

1.1. 研究動機與背景

近年來，手持裝置的普及率不斷上升，手持裝置上的應用也不斷推成出新，Mobile Visual Search 就是一個手持裝置上展新與熱門的應用程式。與一般電腦裝置比較起來，手持裝置的輸入裝置有很大的不同，對一般電腦而言，電腦鍵盤可以輕易地輸入文字，但由於體積的限制，在手持裝置上要輸入文字，並不是一件容易的事，於是要在手持裝置上進行以文字為基礎的資料搜尋並不方便；但相對於電腦裝置來說，相機已經是手持裝置上的基本配備，一張照片只要按一個按鈕就可以完成，於是基於手持裝置影像的資料搜尋，也成為了新的應用與熱門起來，包括了 Standford 大學[1][2][3][4]與 MIT 學院[5][6]，也有許多相關的研究。

Mobile Visual Search 不單是一個手持裝置的應用程式，而是一個結合手持裝置與雲端運算的應用，手持裝置所拍攝的照片影像利用網路傳回雲端伺服器，雲端伺服器搜尋既有的影像資料庫，找出最接近照片影像的影像，之後將找出影像的相關資訊傳回至手機上；在 Standford 大學的研究中[1][2][3][4]，他們在雲端伺服器建立一個 CD 封面的影像資料庫，並連結至 CD 內音樂的 Track，當使用者在 CD 商店選購 CD 時，可以馬上利用手持裝置將 CD 封面利用照相機拍下並傳回雲端伺服器，雲端伺服器利用影像搜尋的技術，找出相對映的 CD，將實際 CD 內的音樂 track 直接傳回手持裝置上，使用者可以透過試聽音樂 track 之後，在決定是否要購買 CD；另外，例如利用建築物照片找出建築物的介紹，利用街景照片找出位置或是附近商店資訊等等，也是常見 Mobile Visual Search 的應用。

在 Mobile Visual Search 的系統中，影像快速搜尋成為這個系統的主要關鍵之一，其中包括了影像特徵點的萃取與描述，以及建立以特徵點為基礎的影像資料庫這兩大部分；在特徵點的萃取與描述上，已經有許多這方面研究的論文，包括著名的 SIFT(Scale

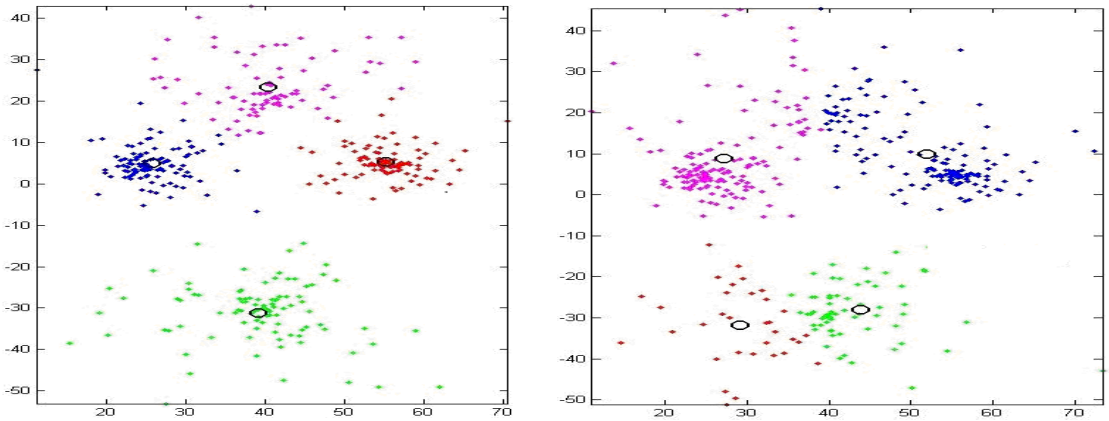
Invariant Feature Transform)與 SURF(Speed up Robust Features)演算法[7][8]，這兩個演算法不僅能夠正確地找出影像特徵點以及有很好的描述式，還具備了速度快的優點，本論文選擇使用 SURF 演算法做為影像特徵點的萃取與描述；但在建立影像資料庫上，國內外的研究比較少見，最有名的就是 D. Nist´er and H. Stew´enius [9]這篇論文，不論是 Standford 與 MIT 的研究中，都是選用[9]這篇論文所提出方式來建立影像資料庫的索引，以達成快速找到對應影像的目標；於是本論文針對了[9]這篇論文做了深入的研究，最後提出 Hierarchical SVM 演算法建立 Vocabulary Tree，來改善研究過程中一些發現[9]方法的問題，最後提高了比對正確率以及降低比對複雜度。

1.2. 背景知識

在[9]方法的研究中主要分成兩大部分，一是使用 Hierarchical K-means 演算法建立 Vocabulary Tree，二是將影像資料庫與搜尋影像特徵點歸類於 Vocabulary Tree 的葉節點上，然後找出影像資料庫中所有影像在葉節點上特徵點數目的比例，與搜尋影像在葉節點上特徵點數目比例最相近的影像為相似影像。

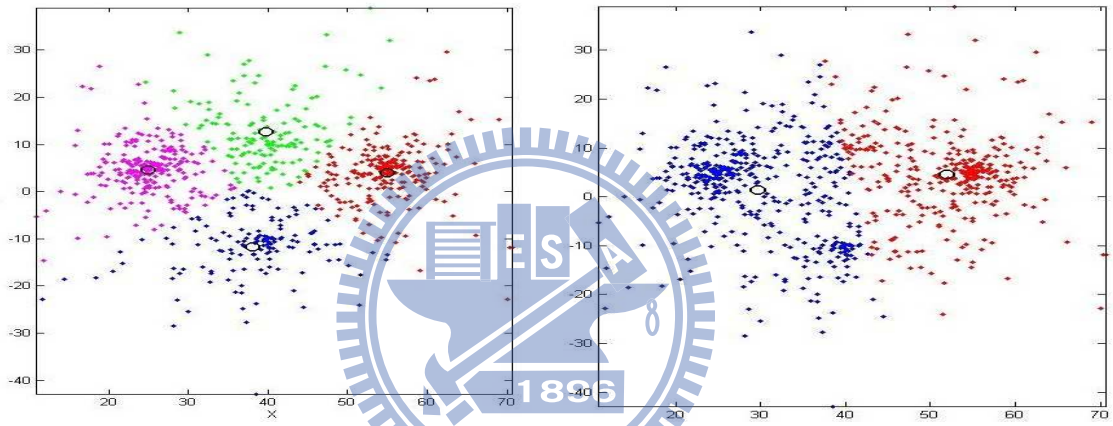
Hierarchical K-means 演算法的做法為使用 K-means 演算法將所有影像特徵點分成 K 類，然後將每一個分類的影像特徵點依序再分成 K 類，依此類推直到一個定義的階層數目 L，或是每一個類別中影像的特徵點數目已經小於 K 做為停止，最後將每層 K-means 演算法所產生的分類中心點串成階層樹，形成所謂的 Vocabulary Tree，在 Vocabulary Tree 上，最上面的節點為根節點，最下面的節點為葉節點。

影像資料庫所有影像的特徵點將會歸類至 Vocabulary Tree 上的葉節點，歸類的方式為從 Vocabulary Tree 的根節點開始，找尋距離最近的節點，依此類推往下尋找最接近節點，直到葉節點為止；同樣地方式將搜尋影像的所有特徵點歸類於葉節點，之後根據葉節點上特徵點的比例，找出比例最接近的影像就是相似影像。



(A) (B)

圖 1.1 K-means 演算法初始中心點不同的影響示意圖



(A) (B)

圖 1.2 K-means 演算法分類數目不同的影響示意圖

1.3. 問題陳述

使用[9]研究中的方法，證實可以提供很好的分類方式，達到快速比對的目的，但是在其方法中，使用 Hierarchical K-means 演算法將所有影像的特徵點進行階層式分類，主要的問題如下：

1. 首先，由於 Hierarchical K-means 演算法是利用 K-means 演算法在每個階層上進行分類，使用 K-mean 演算法進行分類，其分類的結果會受到選擇初始中心點的影響，造成分類隨著實驗變動，使得影像辨識的困難，如圖 1.1 所示。

   由圖 1.1 可以發現，一個四個類別的資料使用 K-means 演算法分成四類，有可能因

為初始值選擇的不同，有不同的分類結果；圖 1.1(A)顯示 K-means 演算法正確地將資料分成四類，但圖 1.1(B)因為初始點的不同，使得分類結果與原始資料大不相同。

2. 再者在每一層分類上，分類的類別數目都是小於實際分類的類別數目，即使選擇的分類中心點接近實際分類的中心點，其他類別的特徵點就很容易被均勻分割在不同分類，也容易造成影像分辨不正確的結果，如圖 1.2 所示。

   在圖 1.2(A)中，一個四個類別的資料使用 K-means 演算法分成兩類，即使所計算之後的中心點接近於紅色與紫紅色的中心點，原本綠色與藍色的資料卻被均勻地被分割在兩類別上如圖 1.2(B)所示，之後要利用綠色與藍色資料在兩類別的比例，來分辨與綠色或藍色資料的相似度，就容易產生辨識上的錯誤，辨識正確率的期望值只有 75%，造成辨識率不高的情況。

3. 最後，由於無法能夠將特徵點有效地分類，必需要利用葉節點上特徵點的比例來判斷相似影像，隨著分類數目與階層數目的增加，葉節點的數目也隨著增加，比對的複雜度也會相對的增加，另外，隨著影像數目的增加，也會使得比對複雜度的增加。

1.4. 貢獻

雖然，在分類一群未知影像的特徵點是屬於 unsupervised machine learning，但是事實上，真正想要分類的是影像本身，而不是想要分類影像的特徵點，特徵點屬於那個影像是一個有效而且有用的資訊；於是本論文的主要貢獻為建立了 Hierarchical SVM 演算法取代 Hierarchical K-means 演算法，在每一層分類時，將影像資訊做為指導特徵點分類的索引，然後使用 SVM 演算法將特徵點分類，使得影像的特徵點可以集中分類到某一節點上，於是之後在進行影像辨識時，不會因為有影像分類比例相近而分辨錯誤的問題，也因此經過 Vocabulary Tree 的搜尋之後就可以找到正確相似影像，使得整體的比對複雜度，不會因為分類數目以及階層數目的增加而增加。

1.5. 論文架構

除了本章說明了研究的動機與背景之外，後續的章節如下：

第二章為背景知識與相關研究，包括 Mobile Visual Search 的架構[1][2][3][4]、K-means 演算法[10]、K-means++演算法[11]、[9]的方法與 Support Vector Machine 演算法[16][17][18]。

第三章詳細說明如何使用本論文提出方法，包括使用 K-means++演算法將影像初步分類、如何應用 Hierarchical SVM 演算法建立 Vocabulary Tree 以及最後搜尋影像的比對。

第四章為相關的實驗結果。

第五章為結論與未來工作。

2. 背景知識與相關研究

2.1. Mobile Visual Search 架構

Mobile Visual Search 不僅是手持裝置上的應用程式，而是結合手持裝置與雲端運算的應用，Mobile Visual Search 的詳細架構如圖 2.1 所示。由圖 2.1 可知，Mobile Visual Search 可分為三大部分，第一部分是關於特徵點的萃取與描述，第二部分是關於影像資料庫的建立，最後一部分是關於手持裝置與雲端伺服器交換影像或特徵資料。

關於特徵點的萃取與描述，其中較廣為使用的演算法是由 David Lowe 所提出 SIFT 演算法[7]；SIFT 演算法不但能夠正確地找出影像的邊角，還具備尺度不變(scale invariant)與速度快的特性；除了 SIFT 演算法，SURF 演算法[8]是既 SIFT 演算法之後，另外一個廣被使用的演算法，與 SIFT 演算法比較起來，SURF 演算法不但具備 SIFT 演算法的優點，在特徵點的描述上，SURF 演算法只需要 64 維度的向量來描述特徵點，比 SIFT 演算法的 128 維度向量少了一半，之後不論是使用 Hierarchical K-means 演算法或是本論文提出的 Hierarchical SVM 演算法來分類，都可以大大降低計算量，於是本論文選擇使用 SURF 演算法來進行特徵點的萃取與描述，在下面的章節也簡單地介紹 SURF 演算法的內容。

在建立影像資料庫方面，[9]的方法，廣泛地被使用各種研究上；為了瞭解本論文所提出的 Hierarchical SVM 演算法，必須要對[9]的方法有深入地認識，本論文在下面的章節將會詳細說明[9]方法的細節，並探討其優缺點。

最後關於手持裝置與雲端伺服器的溝通，手持裝置可以直接將影像傳至雲端伺服器，另外也可以選擇將特徵點的描述經過壓縮之後，再傳給伺服器作比對；根據 Standford 大學的研究，如果在網路頻寬不夠的情況之下，選擇傳輸壓縮過的特徵點描述，是比較好的方式，也有許多的研究再探討這個部分[2][3][4]。

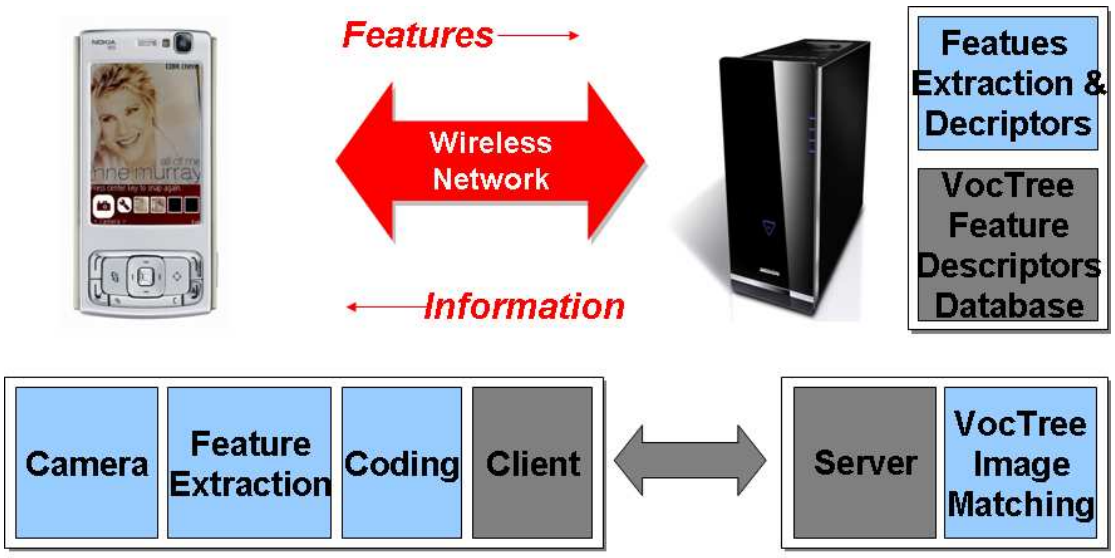


圖 2.1 Mobile Visual Search 架構圖（資料來源：[1]）

2.2. K-means 演算法

由於建立 Vocabulary Tree 是基於 Hierarchical K-means 演算法，Hierarchical K-means 演算法又是基於 K-mean 演算法[10]，首先必須要對 K-means 演算法有所了解。K-means 演算法的內容如下：

1. N 個資料點的集合 X 為式(1)：

$$X = \{x_1, ..., x_N \mid x_i \subset R^d\} \quad (1)$$

2. 將 X 分成 K 類，$K < N$。

3. 隨機從 N 的點中選出 K 個點做為初始中心點為式(2)：

$$c = \{c_1, ..., c_k\} \quad (2)$$

4. 對於 $i \in \{1, ..., K\}$，C_i 為所有 X 中的點，距離 c_i 比 c_j 短的集合，其中 $i \neq j$。

5. 對於 $i \in \{1, ..., K\}$，根據 C_i 中所有的點，重新計算 c_i 為式(3)：

$$c_i = \frac{1}{|C_i|} \sum_{x \in C_i} x \quad (3)$$

6. 重覆步驟 4, 直到 c_i 不再變動為止。

K-means 演算法[10]是由 Stuart Lloyd 在 1957 年所提出，並在 1982 年發表，原

本稱為 Lloyd 演算法，因為廣泛地在學術界被使用於解決 K-means 問題，所以就被稱為 K-means 演算法，大部分的人都誤認 K-means 演算法是解決 K-means 問題的演算法，但其實 K-means 演算法是 K-means 問題的一個近似解演算法(heuristic algorithm)。K-means 問題的定義如下：

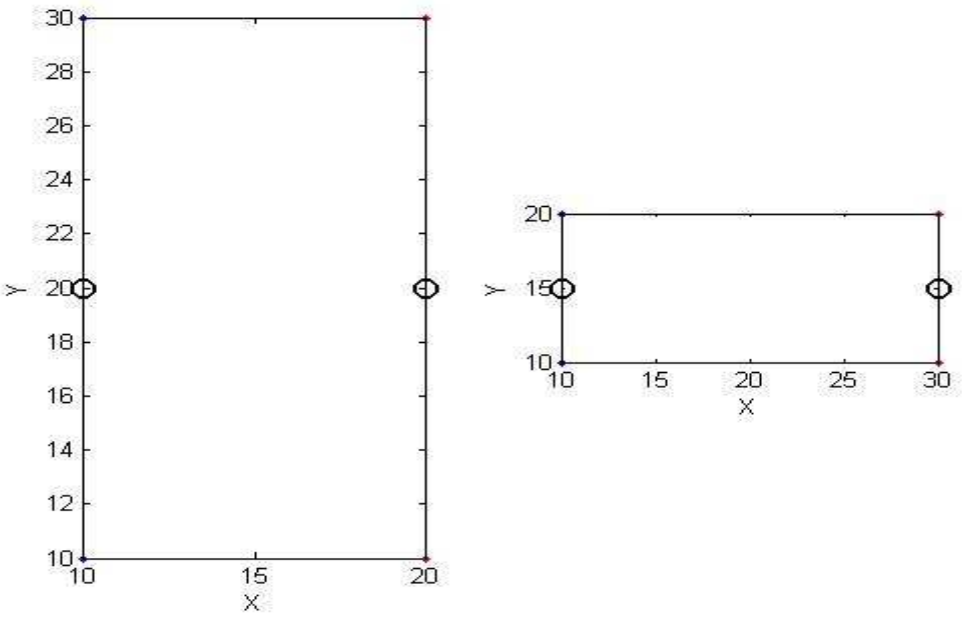
1. N 個資料點的集合 X 為式(4)：

$$X = \{x_1, ..., x_N \mid x_i \subset R^d\} \tag{4}$$

2. 將 X 分成 K 類的 S 集合，$S = \{s_1, ..., s_K \mid K < N\}$，使得所有分類的 sum of squares 能夠最小為式(5)：

$$\arg\min_s \sum_{i=1}^{K} \sum_{x_j \in s_i} |x_j - u_i|^2, u_i = \frac{1}{|x_j|} \sum_{x_j \in s_i} x_j \tag{5}$$

根據許多的研究顯示，K-means 的問題是一個 NP-Hard 的問題[12][13]，於是有許多的 heuristic algorithm 來解決 K-means 的問題[10][11][14][15]，K-means 演算法就是其中之一。事實上，K-means 演算法是利用 local optimum 的方式來解決 K-means 問題，回顧之前 K-means 的演算法可以發現，K-means 演算法是找出已經分類好的最佳解，而不是整體的最佳解；在這樣地情況下，選擇初始點的好壞，決定 K-means 演算法結果的好壞。以圖 2.2 為例，在圖 2.2(A)中，長方形(10, 10)、(15, 10)、(10, 20)、(15, 20)，選擇最下方兩個點進行 K-means 演算法之後，收斂至長方形長的邊；但在圖 2.2(B)中，長方形(10, 10)、(20, 10)、(10, 15)、(20, 15)，選擇最下方兩個點進行 K-means 演算法之後，收斂至長方形長短的邊，但是不論是圖 2.2(A)或是圖 2.2(B)，真正的最佳解應該是短的邊。因此初始點的選擇對 K-means 演算法有很大的影響，也會對所建立的 Vocabulary Tree 產生影響。



(A) (B)

圖 2.2 K-means 演算法收斂中心點

2.3. K-means++演算法

針對 K-means 演算法的問題，David Arthur 與 Sergei Vassilvitskii 於 2007 年提出了 K-means++演算法[11]，提供了一個比較好的方式，選擇 K-means 演算法的初始點，在 David Arthur 的研究中，證明了 K-means++演算法是 $O(\log k)$ 近似於最佳解。K-means++演算法的內容如下：

1. N 個資料點的集合 X 為式(6)：

$$X = \{x_1, ..., x_N \mid x_i \subset R^d\} \quad (6)$$

2. 將 X 分成 K 類，$K < N$。

3. 從集合 X 中，隨機選擇一個點做為第一個中心點 C_1。

4. 從集合 X 中，根據 $\frac{D(x)^2}{\sum_{x \in X} D(x)^2}$ 的機率，選擇下一個中心點，其中 $D(x)$ 代表 x 到最接中心點的距離。

5. 重覆步驟 4, 直到找到 K 個中心點為式(7)：

$$c = \{c_1, ..., c_k\} \quad (7)$$

6. 對於 $i \in \{1, ..., K\}$，C_i 為所有 X 中的點，距離 c_i 比 c_j 短的集合，其中 $i \neq j$。

7. 對於 $i \in \{1, ..., K\}$，根據 C_i 中所有的點，重新計算 c_i 為式(8)：

$$c_i = \frac{1}{|C_i|} \sum_{x \in C_i} x \quad (8)$$

8. 重覆步驟 6, 直到 c_i 不再變動為止。

在 K-means++演算法中，除了步驟 3 到 5 之外，其他的步驟與 K-means 演算法完全一樣。原本 K-means 演算法是使用隨機的方式來選擇初始中心點，但是在 K-means++演算法的步驟 4 中，是以資料與最接近初始中心點的距離為機率來選擇初始中心點，也就是說距離現有初始中心點遠的資料，有比較大的機會被挑選到成為下一個初始中心點，於是初始中心點會比較分散，不會集中於相近的資料，而得到比 K-means 演算法較佳的結果。

之後本論文的方法將先使用影像中心點進行初步分類，在分類時，一是希望相似影像分成同類，二是希望不同類別的影像可以越遠越好，於是選用 K-means++演算法來進行影像初步分類，以達到更好的分類結果。

2.4. Scalable recognition with vocabulary tree

本節詳細說明[9]所提出的方法，包括使用 Hierarchical K-means 演算法建立影像資料庫的 Vocabulary Tree，以及建立影像資料庫的 Vocabulary Tree 之後，如何利用 Vocabulary Tree 來做影像比對，最後也將詳細說明 Hierarchical K-means 演算法的問題。

2.4.1. Hierarchical K-means 演算法

Hierarchical K-means 演算法的內容如下：

1. 一個包含了 N 張影像的影像資料庫為集合 T 為式(9)：

$$T = \{I_1, ..., I_N\} \quad (9)$$

2. 每張影像是由 m_i 個特徵點所構成 I_i 為式(10)：

$$I_i = \{f^i{}_1, \ldots, f^i{}_{m_i} \mid f^i{}_j \subset R^d\} \tag{10}$$

其中對於影像 I_i 與 I_j 來說，m_i 與 m_j 可以不相等，d 決定於使用的特徵點描述式，由於本論文使用 SURF 演算法的特徵點描述式，所以 d 會等於 64。

3. 將所有影像的特徵點形成集合 F 為式(11)：

$$F = \{f^1{}_{1,\ldots,} f^1{}_{m_1}, \ldots, f^N{}_{1,\ldots,} f^N{}_{m_N}\} \qquad N_f = \sum_{i=1}^{N} m_i \tag{11}$$

4. 使用 K-means 演算法將 F 分成 M 類得到集合 K 與集合 K 的中心點集合為 C，其中 c_i 為 K_i 的中心點為式(12)與式(13)：

$$K = \{K_1, \ldots, K_M \mid M < N_f\} \qquad C = \{c_1, \ldots, c_M\} \tag{12}$$

$$K_i = \{I'_j \mid j \subset (1 \ldots N)\} \qquad I'_i = \{f^i{}_j \mid j \subset (1, \ldots, m_i)\} \tag{13}$$

假設 K_i 中有 N' 的影像數目，而且每 K_i 中的 I'_i 影像有 m'_i 個特徵點。於是 K_i 的特徵點集合為為式(14)：

$$F_{K_i} = \{f^1{}_{1,\ldots,} f^1{}_{m'_1}, \ldots, f^{N'}{}_{1,\ldots,} f^{N'}{}_{m'_{N'}}\} \quad N_{K_i} = \sum_{j=1}^{N'} m'_j \quad c_i = \frac{1}{N_{K_i}} \sum_{j=1}^{N'} \sum_{k=1}^{m'_j} f^j{}_k \tag{14}$$

5. 將 $F_{K_i}, \forall i = 1, \ldots, M$ 重覆步驟 4，直到 $N_{K_i} < M$ ，或者是重覆步驟 4 的次數，超過了一個定義的階層次數 L 。

6. 所有不同層的 c_i 將構成 Vocabulary Tree，c_i 是 Vocabulary Tree 上面的一個節點，如果一個節點下面的沒有其他的節點，這個節點將形成葉節點，影像中所有的特徵點將被指定葉節點上。

以圖 2.3 為例，說明建構 Vocabulary Tree 的流程。

1. 一開始使用 K-means 演算法，將所有影像的特徵點分為三個類別，如圖 2.3 左上方圖所示。

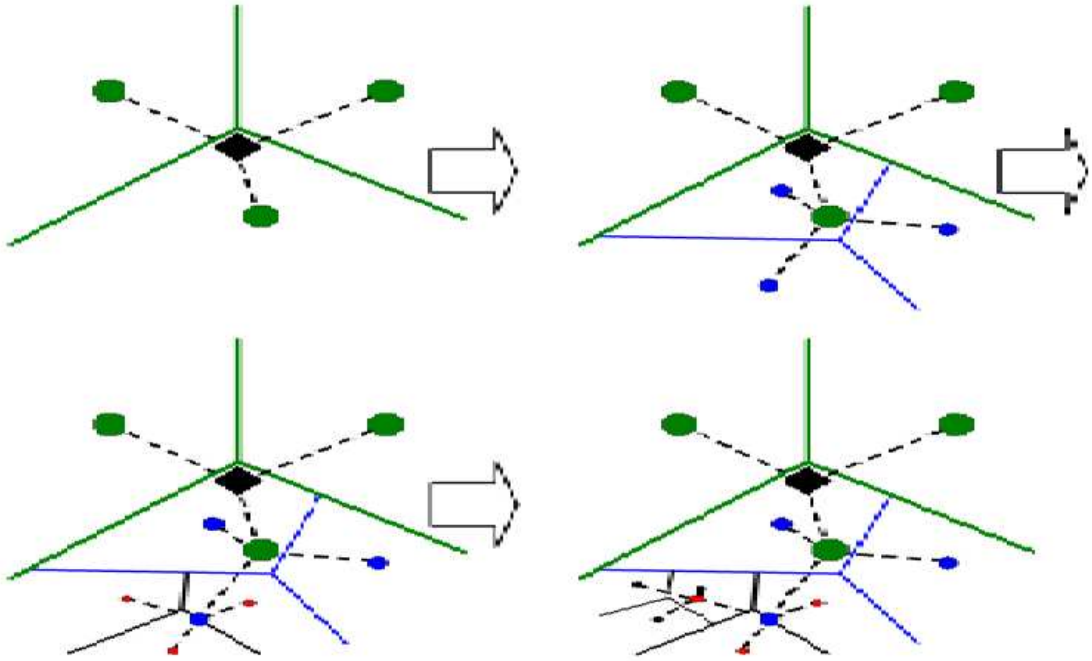


圖 2.3 建構 Vocabulary Tree 的流程（資料來源：[9]）

2. 以在最下方的分類為例，將所有屬於這個分類的特徵點再利用 K-means 演算法分成三類，如圖 2.3 右上方圖所示。
3. 依此類推，將分類好的類別再使用 K-means 演算法分類，如圖 2.3 下方圖所示。
4. 每個階層 K-means 演算法計算之後的中心點形成 Vocabulary Tree，每個中心點是 Vocabulary Tree 的節點，最下面的節點是葉節點，所有的特徵點最後將被指定葉節點。

2.4.2. 影像比對

建立好 Vocabulary Tree 之後，就可以利用 Vocabulary Tree 做影像比對，在 scalable recognition with vocabulary tree 中，是利用計算葉節點上特徵點的數量比例為分數，來做影像辨識的根據。特徵點數量比例的分數是利用 TF(Term Frequency)-IDF(Inversed Document Frequency)來計算，然後算出搜尋影像與所有影像資料庫在每一葉節點上 TF-IDF 的誤差和，最小的誤差和為就是 match 的影像。

在計算 TF-IDF 之前，首先必須先將搜尋影像的特徵點，全部指定到 Vocabulary

Tree 的葉節點之中；指定方式是由 Vocabulary Tree 的最上層到最下層，首先在最上層找尋與特徵點距離最近的節點，然後往這個節點下一層，再找尋最近的節點，依次類推一直找到葉節點為止，最後所有搜尋影像的特徵點，都會被指定到特定的葉節點；之後就可以根據葉節點所擁有特徵點的數量，來計算 TF-IDF。下面是計算 TF-IDF 分數的方法：

1. 假設 Vocabulary Tree 的葉節點的集合為VL為式(15)：

$$VL = \{LT_1, ..., LT_T \mid M \le T \le M^L\} \tag{15}$$

2. 影像I_i在LT_i的 Term Frequency 定義為影像I_i在LT_i特徵點的數目除以影像I_i的特徵點的數目為式(16)：

$$TF_{I_i|LT_i} = \frac{m_{I_i|LT_i}}{m_{I_i}} \tag{16}$$

3. 每一個葉節點LT_i來說，其 Inversed Document Frequency 定義為所有影像數目除以在葉節點影像數目取 log 為式(17)：

$$IDF_{LT_i} = \log \frac{N}{N_{LT_i}} \tag{17}$$

4. 影像I_i在LT_i的 TF-IDF 定義為$W_{I_i|LT_i}$為式(18)：

$$W_{I_i|LT_i} = \frac{m_{I_i|LT_i}}{m_{I_i}} * Log \frac{N}{N_{LT_i}} \tag{18}$$

搜尋影像q在LT_i的 TF-IDF 定義為$W_{q|LT_i}$為式(19)：

$$W_{q|LT_i} = \frac{m_{q|LT_i}}{m_q} * Log \frac{N}{N_{LT_i}} \tag{19}$$

5. 搜尋影像對影像I_i的分數為式(20)：

$$S(q, I_i) = \left| \frac{W_q}{|W_q|} - \frac{W_{I_i}}{|W_{I_i}|} \right| \tag{20}$$

6. 最後計算出所有影像與搜尋影像的$S(q, I_i)$，最小 S 的影像就是根搜尋影像最相似

地影像。

2.4.3. 分析

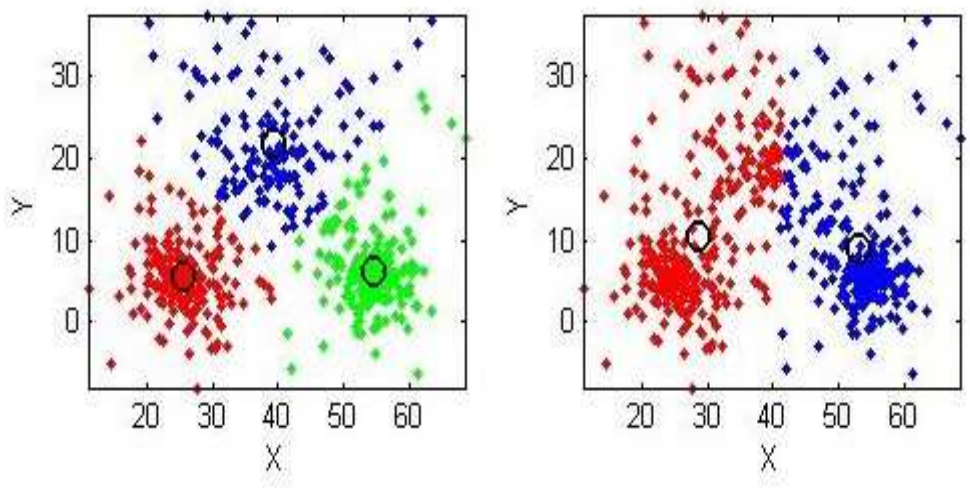
Hierarchical K-means 演算法提供了一個有效的方式建立 Vocabulary Tree，加速之後的影像比對，但是由於 Hierarchical K-means 演算法是基於 K-means 演算法，而 K-means 演算法又不見得每次都是最佳解，所以計算的結果常常隨著每次實驗有所不同。

即使有一個最佳解的演算法來處理 K-means 問題，Hierarchical K-means 演算法還是有一個的問題存在；回顧 Hierarchical K-means 演算法中，在步驟 4 使用 K-means 演算法進行分類時，Hierarchical K-means 演算法將 F 分成 M 類別，此時，M 將遠小於真正影像類別，即使選擇的初始點都在部分影像的中心點上，其他影像的特徵點有很大的機率散落於不同的類別，造成之後再分類或比對的困難，尤其是在分類遠小於真正類別的情況。下面是利用一個簡單的範例來說明，分類的問題：

1. 假設一個實際有三群混在一起的資料要進行分類。
2. 如果使用 K-means 演算法將其三類，雖然可能分類的結果每次不同，但大部分都可以得到很好的結果如圖 2.4(A)所示。由圖 2.4(A)可知，K-means 演算法可以正確地將資料分成三群，也就是說如果使用 K-means 演算法，經過重覆計算找出最小誤差的情況之下，K-means 演算法可以接近最佳解。
3. 但是如果將上面三群的資料分成兩群，得到的結果如圖 2.4(B)所示。由圖 2.4(B)可知，即使在最佳解的情況之下，原本圖 2.4(A)的藍色資料被均分到另外綠色與紅色資料之中，而造成藍色資料分配不佳。

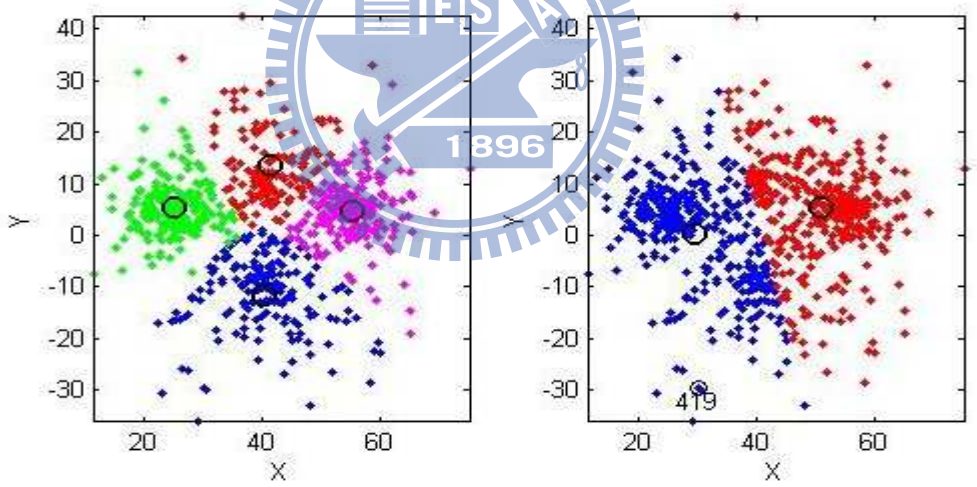
再利用另外一個範例說明分類與比對的問題：

1. 假設一個實際有四群混在一起的資料要進行分類。
2. 使用 K-means 演算法將其四類，雖然可能分類的結果每次不同，但大部分都可以得到很好的結果如圖 2.5(A)所示。



(A) (B)

圖 2.4 K-means 演算法分類範例 I



(A) (B)

圖 2.5 K-means 演算法分類範例 II

3. 但是如果將上面四群的資料分成兩群，得到的結果如圖 2.5(B)所示。由圖 2.5(B)可知，原本圖 2.5(A)的藍色與紅色資料被均分到另外綠色與紫紅色資料之中，之後如果想要利用 TF-IDF 分辨這兩個被分散的資料，時常會有錯誤的情況發生，於是當分類的數目越小於真正的群組數目，越容易發生這樣地現象。

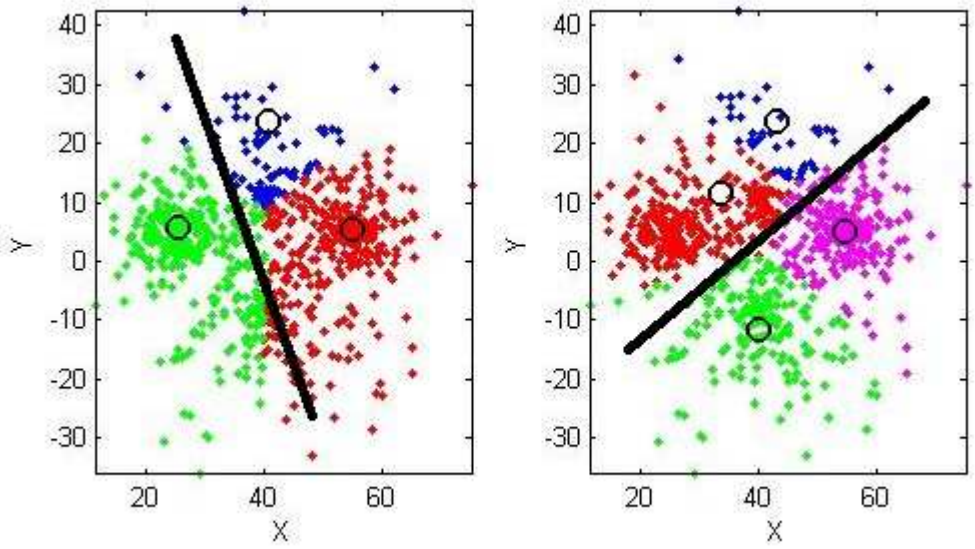


圖 2.6 分類假想示意圖

如果想要將上面三群或四群資料分成兩類，所希望的結果如圖 2.6 所示。也就是說，其實希望能夠將同類的點盡量分在同一群，這樣相同資料要進行比對比較容易，也比較不會有錯誤，另外要再進行分類也能夠比較正確地分類；但是，這樣地結果是因為已經知道了資料有三群與四群，在完全不知道真正群組的情況下，只能使用 K-means 演算法來進行分類。

事實上，K-means 演算法是屬於 unsupervised machine learning 的方式，也就是在不知道群組的情況進行分類，如果可以知道群組資訊，使用 supervised machine learning 的方式，例如 SVM(Support Vector Machine)，KNN(K-nearest neighbor)，或是 Neural Network，都可以得到比 K-means 演算法更好的結果，因為多了群組的資訊，可以使得分類器能夠正確的分類。以上面三群分兩群或四群分兩群的例子來說，使用 K-means 演算法是得到最小誤差的解，但是想要的結果並不是最小誤差，於是即使是最佳解的 K-means 演算法，也無法得到好的分類結果。

但是問題在於進行影像特徵點的分類，群組資訊在那裡呢？再回顧 Hierarchical K-means 演算法，在每一層都是使用 K-means 演算法將特徵點分類，但是真正的目的希望分類影像而非影像的特徵點；也就是說雖然是分類影像的特徵點，但希望相同影像的特徵點希望能夠盡量分在相同的類別中。於是在特徵點上加上了影像的資訊之後，就可

以將原本使用 Unsupervised machine learning 的 K-means 演算法，改成 supervised machine learning 的演算法；因此本論文選擇了 Support Vector Machine 這個 supervised machine learning 的演算法取代 K-means 演算法，最後形成了 Hierarchical SVM 演算法取代 Hierarchical K-means 演算法建立 Vocabulary Tree。

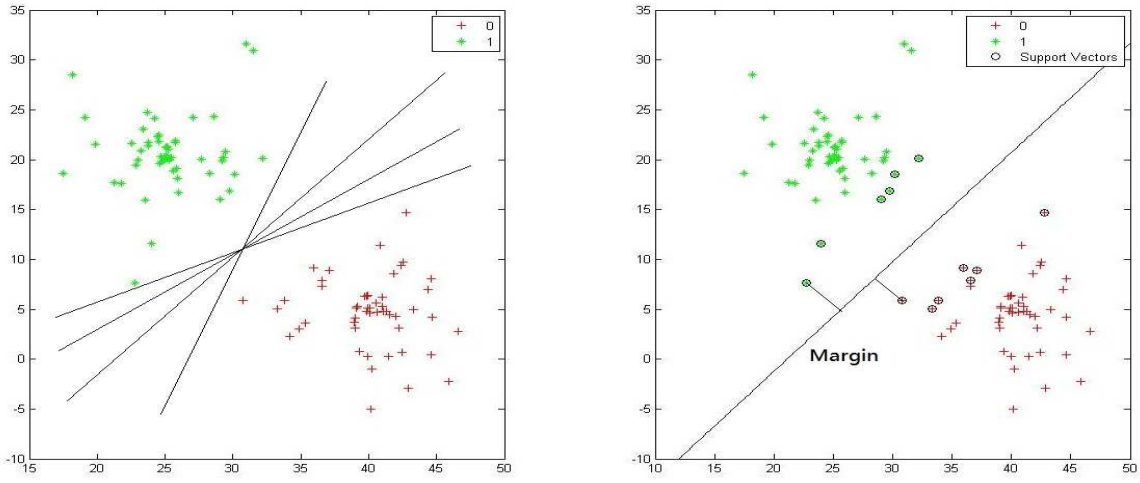
2.5. Support Vector Machine

2.5.1. Support Vector Machine 介紹

支持向量機(Support Vector Machines)[16][17][18]，簡稱 SVM)是一種以統計學習理論為基礎，發展出來的機器學習系統，屬於有監督學習的方法(supervised machine learning)，廣泛的應用於統計分類以及回歸分析中；SVM 原為處理二元類別分類，運算兩類別樣本空間之最佳分割超平面（Optimal Separate Hyperplane)，確立最小分類錯誤。

以圖 2.7(A)為例，在兩群資料中，找尋一個 Hyperplane，來分割這兩群資料，但是可能有 N 個或無限多個的 Hyperplane 存在，於是要找一個 Hyperplane 如圖 2.7(B)所示，使得這個 Hyperplane 到這兩群資料邊界的距離能夠最大。

由於 Support Vector Machine 廣泛地被使用在各種研究上，到今天除了二類別線性的 Support Vector Machine，也發展出不同種類非線性與多類別的 Support Vector Machine[19]。下面的章節，本論文先介紹二類別線性的 Support Vector Machine，之後再延伸至二類別非線性與多類別的 Support Vector Machine。



(A)　　(B)

圖 2.7 Support Vector Machine 分類

2.5.2. 二類別線性 Support Vector Machine

二類別線性 Support Vector Machine 就是要在兩類別中，找出 Hyperplane，使得超平面到兩類別邊界的距離達到最大。其內容如下：

1. 一個包含兩類別資料的集合 X 定義為式(21)：

$$X=\{(x_1,y_1),(x_2,y_2)...,(x_m,y_m)\} \quad x_i\in R^d \quad y_i\in\{+1,-1\} \tag{21}$$

2. 假設可以將 y_i 為+1 與-1 區分的平面為式(22)：

$$w\cdot x+b=0 \tag{22}$$

其中 W 為法向量(Normal Vector)，b 偏移量(bias)。

3. 於是對於所有的 x_i 會滿足式(23)與式(24)：

$$w\cdot x_i+b\geq+1 \quad \text{for} \quad y_i=+1 \tag{23}$$

$$w\cdot x_i+b\leq-1 \quad \text{for} \quad y_i=-1 \tag{24}$$

合並上面兩式可以得到式(25)：

$$y_i(w\cdot x_i+b)-1\geq 0 \quad \forall i \tag{25}$$

4. 邊界的距離為$\frac{2}{|w|}$要使其最大化，於是超平面將滿足式(26)與式(27)：

$$\text{Minimum } \frac{|w|^2}{2} \tag{26}$$

$$\text{Subject to } y_i(w\cdot x_i+b)-1\geq 0 \quad \forall i \tag{27}$$

5. 使用 Langrange multiplier method 來解上面式子。滿足式(26)與式(27)的 Lagrange equation 定義為式(28)：

$$L_p(w,b,a)=\frac{1}{2}|w|^2-\sum_{i=1}^{m}a_i[y_i(w\cdot x_i+b)-1] \quad a_i\in R \tag{28}$$

要求L_p的最小值。

6. 對w與b取偏微分為零以求取L_p的最小值為式(29)與式(30)：

$$\frac{\partial L_p(w,b,a)}{\partial w}=w-\sum_{i=1}^{m}a_iy_ix_i=0 \quad w=\sum_{i=1}^{m}a_iy_ix_i \tag{29}$$

$$\frac{\partial L_p(w,b,a)}{\partial b}=\sum_{i=1}^{m}a_iy_i=0 \tag{30}$$

7. 將上面兩式帶回為L_p得到式(31)：

$$L_D(a)=\sum_{i=1}^{m}a_i-\frac{1}{2}\sum_{i,j=1}^{m}a_ia_jy_iy_jx_i\cdot x_j \tag{31}$$

於是$a_i\geq 0$為L_p限制式，因此可將求解L_p的最小值的問題，轉換成L_D最大值的對偶問題(dual problem) 為式(32)與式(33)：

$$\text{Maximum } L_D(a)=\sum_{i=1}^{m}a_i-\frac{1}{2}\sum_{i,j=1}^{m}a_ia_jy_iy_jx_i\cdot x_j \tag{32}$$

$$\text{Subject to } \sum_{i=1}^{m}a_iy_i=0 \quad \forall a_i \tag{33}$$

8. 由上式可解出a_i並帶回上面式子求的w，使用由 Fletcher 提出的 KKT 條件，可以解出b，最後可以得到一個決策函數(decision function)來判斷x'屬於那一個類別如式(34)。

$$D_f(x')=\text{sgn}(\sum_{i\in SV}a_iy_ix_i\cdot x'+b) \tag{34}$$

2.5.3. 二類別非線性 Support Vector Machine

雖然線性 Support Vector Machine 提供了一個好的方式來分類，但是常常要能夠找到一個線性 Hyperplane 不是件容易的事情。於是 Boser et al.[19]提出了將低維度的原始資料，利用一個映射函數 $\phi(x)$ 投影到高維度的空間，再從高維度空間的找出 Hyperplane 進行分類。其內容如下：

1. 經過 $\phi(x)$ 轉換之後，超平面將滿足式(35)與式(36)：

$$\text{Minimum } \frac{|w|^2}{2}+C\sum_i \xi_i \quad \xi_i \geq 0 \tag{35}$$

$$\text{Subject to } y_i(w\cdot\phi(x_i)+b)-1+\xi_i \geq 0 \quad \forall i \tag{36}$$

2. 轉換 Lagrange 的對偶問題(dual problem)為式(37)：

$$\text{Maximum } L_D(a)=\sum_{i=1}^{m} a_i-\frac{1}{2}\sum_{i,j=1}^{m} a_i a_j y_i y_j k(x_i\cdot x_j) \quad k(x_i,x_j)=\phi(x_i)\cdot\phi(x_j)$$

$$\text{Subject to } \sum_{i=1}^{m} a_i y_i = 0 \quad \forall C \geq a_i \geq 0 \tag{37}$$

3. 最後可以得到一個決策函數(decision function)來判斷 x' 屬於那一個類別如式(38)。

$$D_f(x')=\text{sgn}(\sum_{i\in SV} a_i y_i k(x_i,x')+b) \tag{38}$$

其中 $K(x_i,x_j)$ 稱為 Kernel 函數。當使用非線性 Support Vector Machine 時，必須先選擇 Kernel 函數。目前比較常見的 Kernel 數如下：

1. linear kernel

$$k(x_i,x_j)=x_i\cdot x_j \tag{39}$$

2. polynomial kernel

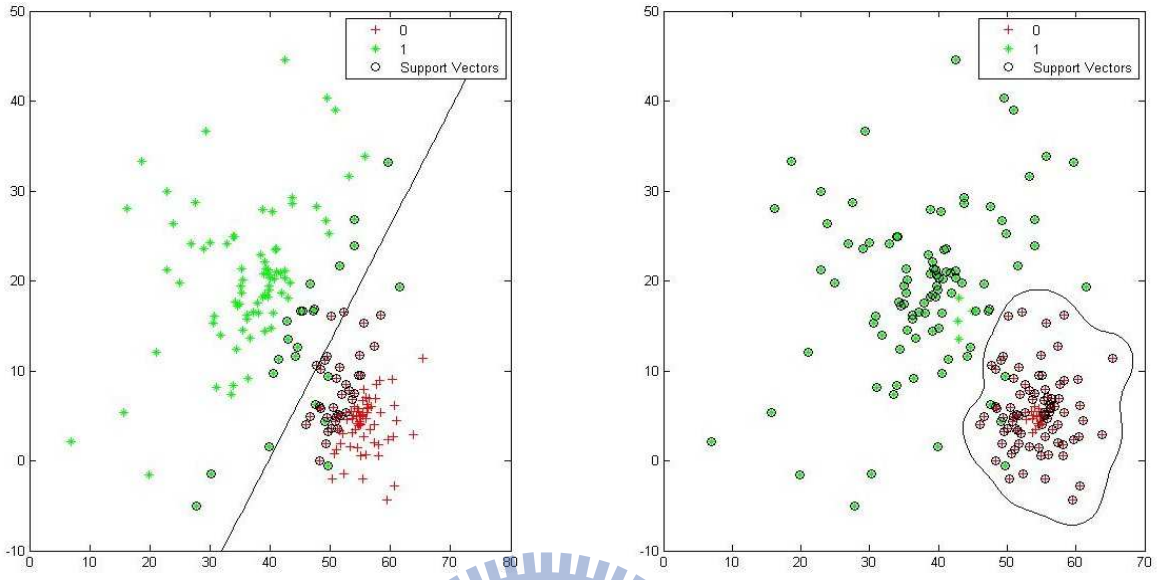
$$k(x_i,x_j)=(\gamma(x_i\cdot x_j)+k)^d, r\geq 0, k\geq 0, d\geq 0 \tag{40}$$

3. multi-layer perception(MLP kernel)

$$k(x_i,x_j)=\tanh(\gamma(x_i\cdot x_j)+k), r\geq 0, k\geq 0 \tag{41}$$

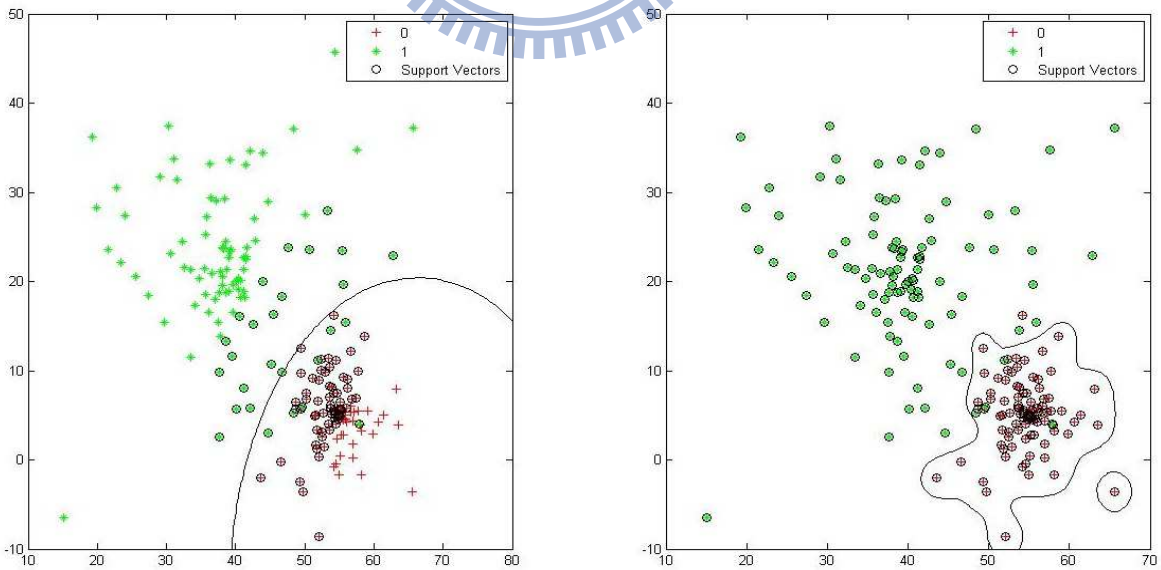
4. Radial Basis function(RBF kernel)

$$k(x_i, x_j) = \exp(-\frac{|x_i - x_j|^2}{\sigma^2}) \quad (42)$$



（A）線性　　（B）RBF（$\sigma = 0.3$）

圖 2.8 Support Vector Machine 分類範例 I



（A）RBF（$\sigma = 2$）　　（B）RBF（$\sigma = 0.2$）

圖 2.9 Support Vector Machine 分類範例 II

圖 2.8 的例子為線性與非線性的不同，圖 2.8(B)使用 RBF kernel 函數能夠得到更佳的分類。圖 2.9 的例子為 RBF Sigma 不同的結果，Sigma 越小分類越精確，但由於分類的區域越小，之後判斷錯誤的機率也會提高。

本論文之後的實驗將使用非線性的 RBF Kernel 函數。

2.5.4. 多類別 Support Vector Machine

Support Vector Machine 是一個二元的分類器，如要解決多類別(Multi-class)分類的方式，必需結合多個二分法來達成多分類的目標。目前 Support Vector Machine 常見處理多類別問題大致上可分為 One-against-all method 與 One-against-one method 這兩種方式。

1. One-against-all method

假設要分成 K 個類別，此方式會產生 K 個 SVM。其中第 i 個 SVM 產生的方式，是將第 i 類別的資料標記成+1，其他類別的所有資料則標記成-1，而得到 $D^i{}_f$ 的決策函數為式(43)

$$D^i{}_f(x') = \mathrm{sgn}(\sum_{j \in SV} a^i{}_j y^i{}_j k(x^i{}_j, x') + b^i) \qquad (43)$$

某一個測試資料 x' 經過每一類 SVM 的決策函數可以得到一個決策值，最大決策值的類別則為 x' 的類別。

2. One-against-one method

假設要分成 K 個類別，此方式會產生 $\frac{k \cdot (k-1)}{2}$ 個 SVM。也就是說任兩個類別產生一個 SVM，而得到 $D^{ij}{}_f$ 的決策函數為式(44)

$$D^{ij}{}_f(x') = \mathrm{sgn}(\sum_{k \in SV} a^{ij}{}_k y^{ij}{}_k k(x^{ij}{}_k, x') + b^{ij}) \quad i \neq j \qquad (44)$$

對測試資料 x'，one-against-one 採取投票策略(voting strategy)，如果決策函數 $D^{ij}{}_f$ 決定為 i 類別，則 i 類別多一票，反之 j 類別多一票，最高票的類別則為 x' 的類別，如果有相同最高票的類別，則隨機選擇一個最高票的類別為 x' 的類別。

2.6. SURF 演算法

SURF 演算法包括了兩個部分，一是快速 Hessian 特徵點偵測，二是 SURF 特徵點描述子。

2.6.1. 快速 Hessian 特徵點偵測

首先 SURF 演算法定義積分影像(Integral Image)I_Σ，假設$X=(x,y)$代表影像上的像素點，$I(X)$為X點上的像素值，積分影像定義為從原點到 X 點的區域中，所有像素值的和，如式(45)所示。

$$I_\Sigma(X)=\sum_{i=0}^{x}\sum_{j=0}^{y}I(x,y) \tag{45}$$

將影像上所有像素點的積分影像計算之後，要計算一個矩形區域像素值的和，可以利用矩形上左上方與右下方像素點的積分影像和，減去左下方與右上方像素點的積分影像和，如圖 2.10 所示。

在平面上 discriminant of Hessian Matrix 可以用來偵測區域的極點代表特徵點，將其延伸至尺度空間(scale space)，Hessian Matrix 與 discriminant 分別為式(46)與(47)。

$$H(X,\sigma)=\begin{pmatrix}L_{xx}(X,\sigma) & L_{xy}(X,\sigma)\\ L_{xy}(X,\sigma) & L_{yy}(X,\sigma)\end{pmatrix} \tag{46}$$

$$\det(H)=L_{xx}L_{yy}-(L_{xy})^2 \tag{47}$$

其中L_{xx}與L_{yy}為$I(X)$在影像上對x方向上的二次微分，也就是 Laplacian of Gaussian，同樣地L_{xy}為$I(X)$在影像上對x方向上的一次微分與y方向上的一次微分。

Lowe[7]使用了 Difference of Gaussian 來近似 Laplacian of Gaussian 以簡化計算的複雜度；同樣地概念，Bay[8]提出 Box Filter 的方式來近似 Laplacian of Gaussian，如圖 2.11 所示。

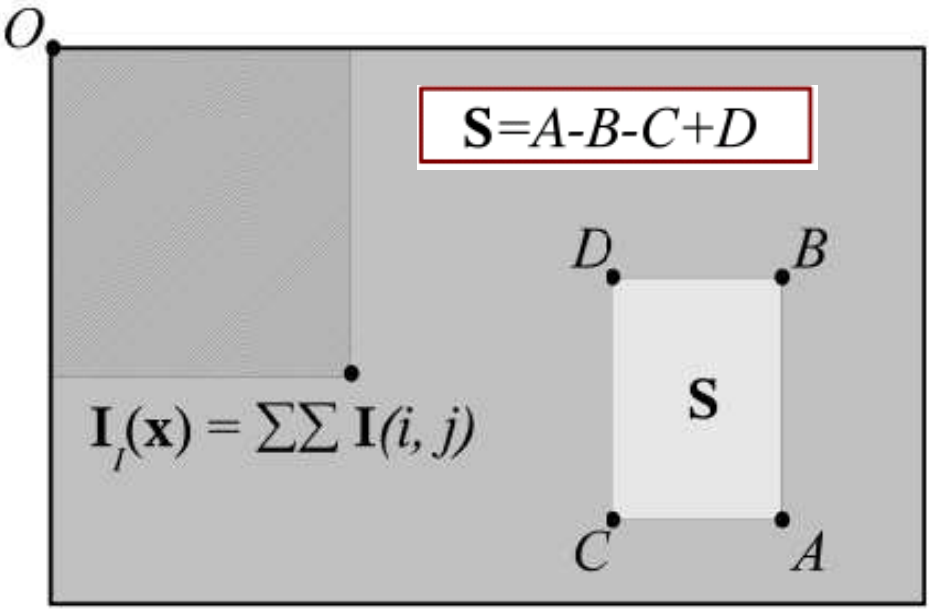


圖 2.10 積分影像與區域像素值的和(資料來源：[8])

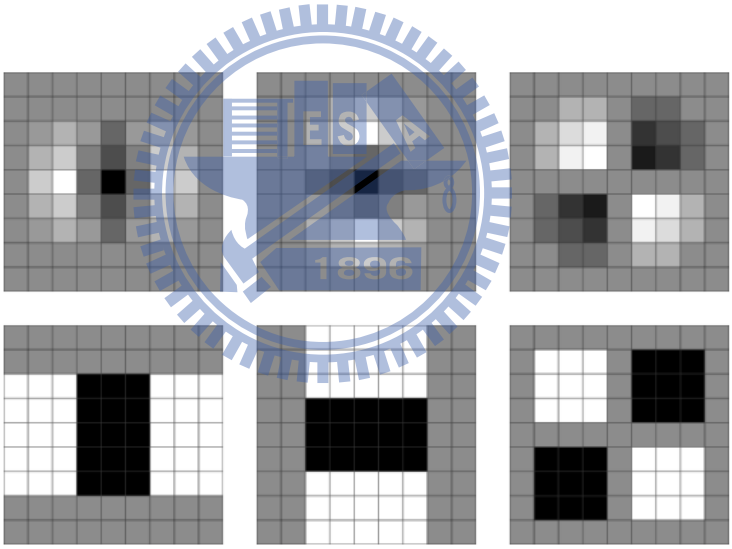


圖 2.11 使用 Box Filter 來近似 Laplacian of Gaussian （資料來源：[22])

原本 Laplacian of Gaussian 的計算，可以使用圖 2.11 上半部L_{xx}、L_{yy}與L_{xy}的 mask 跟影像做 convolution 來求得，Bay[8]將其簡化成圖 2.11 下半部D_{xx}、D_{yy}與D_{xy}的 mask，其中D_{xx}與D_{yy}黑色部分為 2、D_{xy}黑色部分為 1、其他白色部分為-1、灰色部分為 0，原本也可以跟影像做 convolution 來求得，但是可以使用上面的積分影像快速求得。

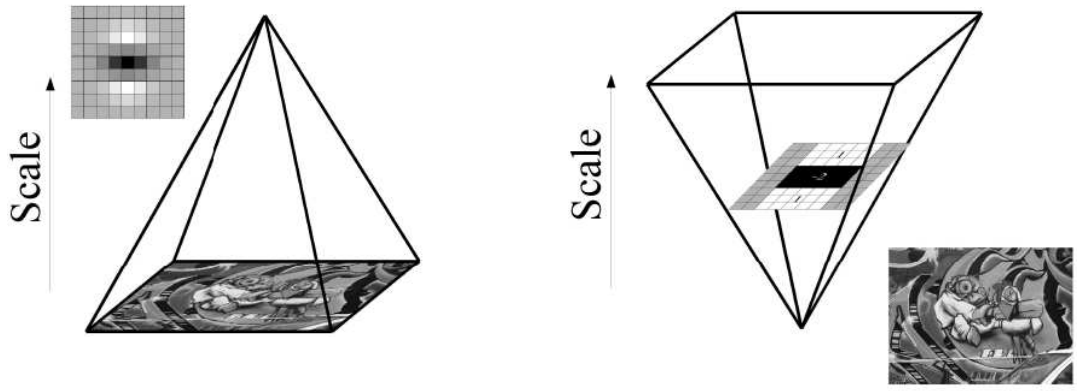


圖 2.12 Lowe[7]與 Bay[9]尺度空間示意圖（資料來源：[22]）

由於 Bay[8]使用 D_{xx}、D_{yy} 與 D_{xy} 來近似 L_{xx}、L_{yy} 與 L_{xy}，於是近似 discriminant of Hessian Matrix 定義為式(48)。

$$\det(H_{approx}) = D_{xx}D_{yy} - (0.9D_{xy})^2 \tag{48}$$

求得在平面空間與尺度空間近似 discriminant of Hessian Matrix 的極值，其所在像素位置就是要尋找的特徵點。

尺度空間的建立與 Lowe[7]的方式不同，Lowe[7]是將影像縮小構成金字塔的尺度空間，Bay[8]使用放大 Box filter 構成倒金字塔的尺度空間，如圖 2.12 所示。

2.6.2. SURF 特徵點描述式

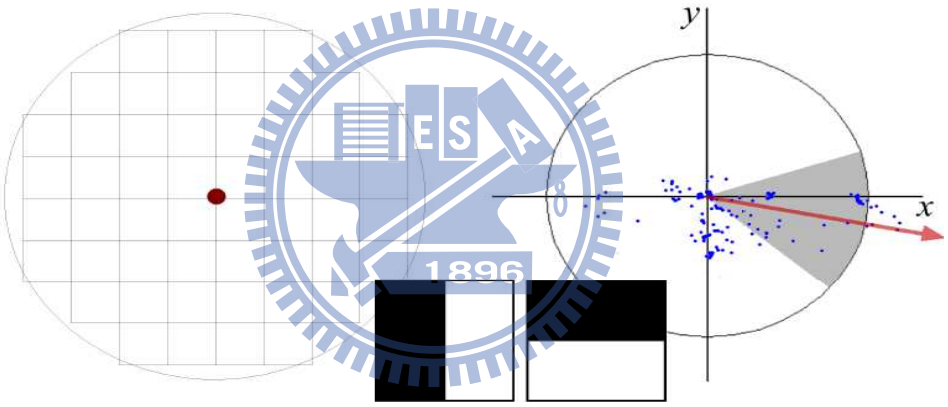
找出影像中所有的特徵點之後，接下來計算特徵點的描述式來表示特徵點。SURF 演算法計算的特徵點描述式有兩個步驟，一是計算特徵點的方向，二是根據特徵點的方向建構矩形區域，計算矩形區域的像素值變化量為描述式。

特徵點方向的計算，是以特徵點為中心建構一個 6σ 為半徑的圓，計算在圓中所有點 x 與 y 的 Haar 小波響應，其中 Haar 小波響應區域為 4σ 的矩形區域；之後在圓上以 60 度為窗口，將所有的 x 與 y 的 Haar 小波響應加其來形成一個向量代表方向，最後最大的向量和為特徵點方向，如圖 2.13 所示。

圖 2.3 中，圖 2.13(A)表示建構 6σ 的圓；圖 2.13(B)為計算 x 與 y 的 Haar 小波響

應，其中黑色部分為 1 白色部分為-1，事實上就是近似一階微分的值也就是代表斜率或是方向，也可以使用 integral image 來計算；圖 2.13(C)表示每 60 度計算出向量和，最大的向量和為特徵點的方向。

最後特徵點描述式的計算，如圖 2.14 所示。根據特徵點的方向，建構一個 20σ 大小的矩形區域，並將這個矩形區域分成 4x4 的子區域；每個子區域以 5x5 為大小做取樣，在每個取樣點上，計算對於特徵點垂直與水平方向的 Haar 小波響應為 d_x 與 d_y ，並乘以一個高斯濾波，將子區域中計算所有取樣點 d_x 與 d_y 的和與絕對值和，得到 $\sum d_x$ 、$\sum d_y$ 、$\sum |d_x|$ 與 $\sum |d_y|$ 四個值，所有 4x4 子區域形成 64 個值的向量，為特徵點的描述式。



(A) (B) (C)

圖 2.13 計算特徵點方向(資料來源：[8])

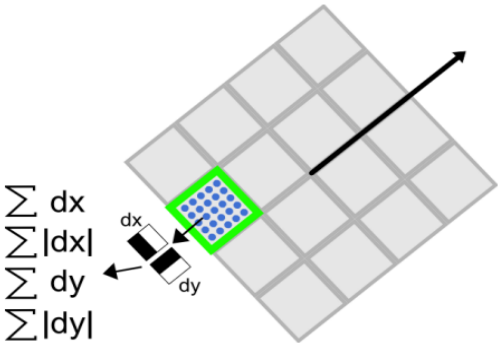


圖 2.14 特徵點描述式的計算(資料來源：[22])

3. 基於 Hierarchical SVM 演算法建立 Vocabulary Tree

3.1. 簡介

本論文所提出的方法主要也是先利用 hierarchical 的方式建立 vocabulary tree，然後使用所建立的 Vocabulary Tree 加速影像比對。

在建立每一層 Vocabulary Tree 的分類時，本論文捨棄了使用 K-means 演算法，主要的步驟更改為：

1 在建立 Vocabulary Tree 的每一層分類時，本論文加入了特徵點的影像資訊，首先計算影像特徵點的平均值為影像中心點，之後將影像中心點使用 K-means++演算法[11]進行初步分類，使得相近的影像歸於同類。

2 再將初步分類的結果，將所有相同類別中所有影像的特徵點，設定為相同的索引值，使用 supervised machine learning 的 Support Vector Machine 演算法[16][17][18]進行特徵點真正的分類。

由於是使用 Hierarchical 的方式而且主要使用 SVM 來分類，於是將這個方式稱為 Hierarchical SVM 演算法建立 Vocabulary Tree。

Vocabulary Tree 建立之後，就可以在每一層的 Vocabulary Tree 節點上，使用 SVM 的決策函數將搜索影像做分類，從 Vocabulary Tree 最上層的節點開始，利用節點上的決策函數，將搜尋影像的特徵點做分類，其中特徵點數目最多的分類，將分類上所有搜尋影像的特徵點再往下一層做分類，直到沒有下一層節點的葉節點為止，葉節點上，最多特徵點的影像就是最相似的影像。

下面小節，將詳細說明 Hierarchical SVM 演算法與影像比對。

3.2. Hierarchical SVM 演算法

本小節詳細說明 Hierarchical SVM 演算法，包括了詳細步驟、流程以及實際的範例。首先是 Hierarchical SVM 的詳細步驟如下：

1. 一個包含了 N 張影像的影像資料庫為集合 T 為式(49)：

$$T = \{I_1, ..., I_N\} \tag{49}$$

2. 每張影像是由 m_i 個特徵點所構成 I_i 為式(50)：

$$I_i = \{f^i_1, ..., f^i_{m_i} \mid f^i_j \subset R^d\} \tag{50}$$

其中對於影像 I_i 與 I_j 來說，$m_i \neq m_j$，d 決定於使用的特徵點描述式，由於本論文使用 SURF 演算法的特徵點描述式，所以 d 會等於 64。

3. 將所有影像的特徵點形成集合 F 為式(51)：

$$F = \{f^1_{1,...,} f^1_{m_1}, ..., f^N_{1,...,} f^N_{m_N}\} \quad N_f = \sum_{i=1}^{N} m_i \tag{51}$$

4. 對於每張影像 I_i，其中心點 C_i 定義為所有特徵點的平均值為式(52)：

$$C_i = \frac{1}{m_i} \sum_{j=1}^{m_i} f^i_j \tag{52}$$

5. 所有影像的中心點形成集合 C 為式(53)

$$C = \{C_i, ..., C_N\} \tag{53}$$

6. 使用 K-means++演算法將 C 集合分成 M 類形成集合 K，集合 K 是由中心點 C_i 所對應的每張影像 I_i 所組成為式(54)與式(55)：

$$K = \{K_1, ..., K_M \mid M < N\} \tag{54}$$

其中

$$K_i = \{I_j \mid j \subset (1, ..., N)\} \qquad \sum_{i=1}^{M} K_i = N \qquad K_i \cap K_j = \phi, \forall i \neq j \tag{55}$$

假設 K_i 中的影像數目為 N'_i，每個影像的特徵點數目為 m'_i，於是 K_i 的特徵點集合為式(56)：

$$F_{K_i} = \{ f^1_{1,\ldots,} f^1_{m'_1}, \ldots, f^{N'_i}_{1,\ldots,} f^{N'_i}_{m'_{N'_i}} \} \quad (56)$$

7. 根據F_{k_i}建立類別索引集合Y_{K_i}以及SVM分類初始資料集合X_{K_i}為式(57)：

$$Y_{K_i} = \{ y^1_{j_1,\ldots,} y^1_{j_{m_1}}, \ldots, y^{N'_i}_{j_1,\ldots,} y^{N'_i}_{m_{N'_i}} \mid \forall y = i \}$$

$$X_{K_i} = \{ (f^1_1, y^1_1)_{,\ldots,} (f^1_{m'_1}, y^1_{m'_1}), \ldots, (f^{N'_i}_1, y^{N'_i}_1)_{,\ldots,} (f^{N'_i}_1, y^{N'_i}_{m'_{N'_i}}) \} \quad (57)$$

8. 使用多類別與非線性的SVM演算法將$X_{K_i}, \forall i = 1,\ldots,M$進行分類，可以得到M類別的集合W，以及SVM的特徵函數為式(58)與式(59)：

$$W = \{ W_1, \ldots, W_M \mid M < N \} \quad (58)$$

其中

$$W_i = \{ I'_j \mid j \subset (1 \ldots N) \} \quad I'_i = \{ f^i_j \mid j \subset (1, \ldots, m_i) \} \quad Nw = \sum_{i=1}^{M} m'_i \quad (59)$$

而且SVM的決策函數為式(60)：

$$D_f(f') = \mathrm{sgn}(\sum a^j_i y^j_i k(f^j_i, f') + b) \quad (60)$$

9. 如果$\frac{m'_i}{m_i}$小於50%，I'_i將會被忽略，而得到W'_i為式(61)：

$$W'_i = \{ I'_j, \ldots, I'_{N'''} \mid N''' \le N'' \} \quad (61)$$

10. 如果N'''大於M，則重覆步驟4，或者是重覆步驟4的次數，小於一個定義的階層次數L。

11. 將所有的SVM特徵函數構成Vocabulary Tree，每一個特徵函數為Vocabulary Tree上面的節點，最下面的節點稱為葉節點。

其中所有步驟的流程如下：

1. 1-3為影像資料庫的定義。
2. 4-8為建構Vocabulary Tree每一層的步驟，包括了3-6使用K-means++演算法做影像初步分類，以及7-8使用SVM演算法進行特徵點真正分類並得到決策函數。

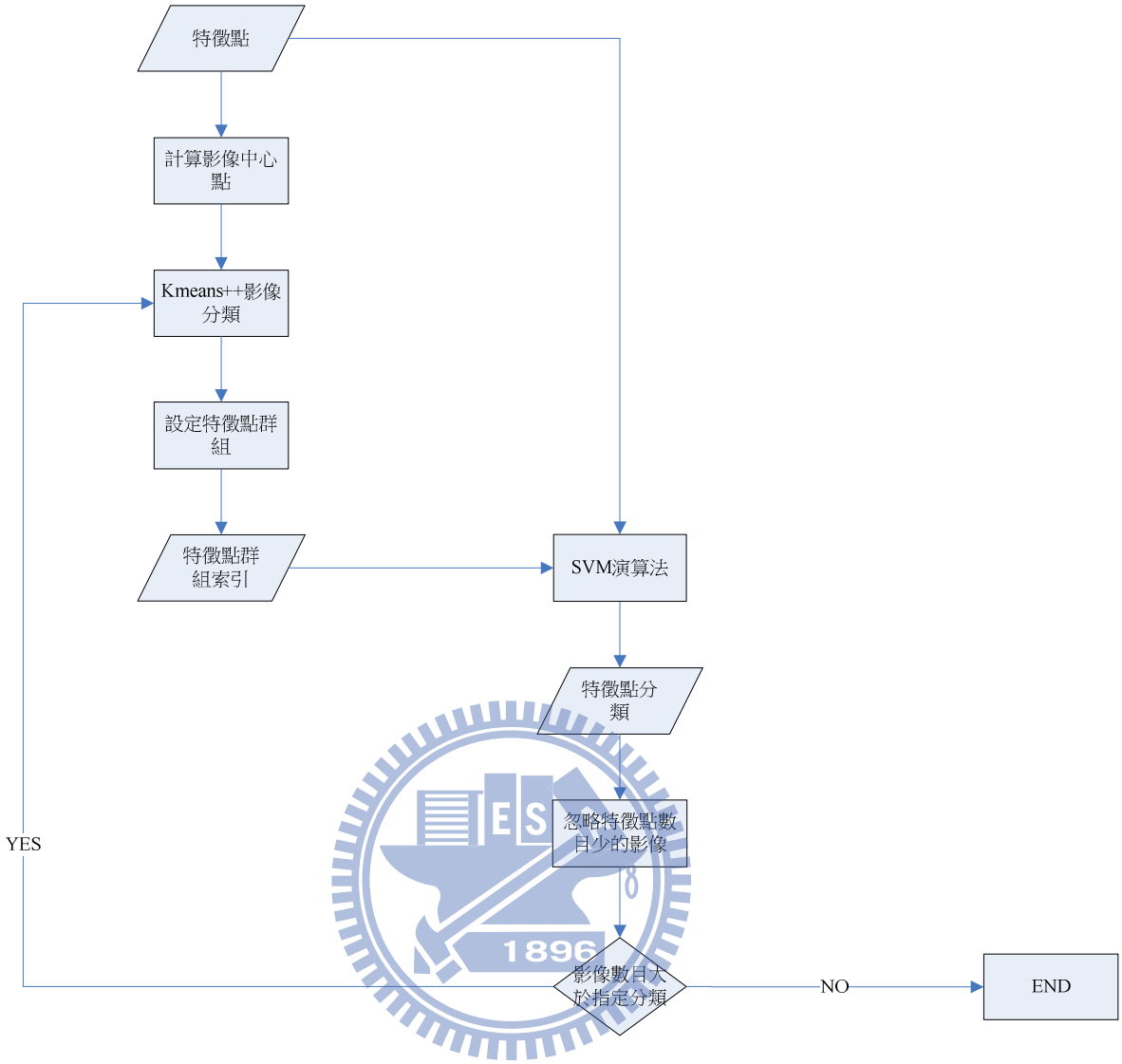


圖 3.1 Hierarchical SVM 演算法的流程

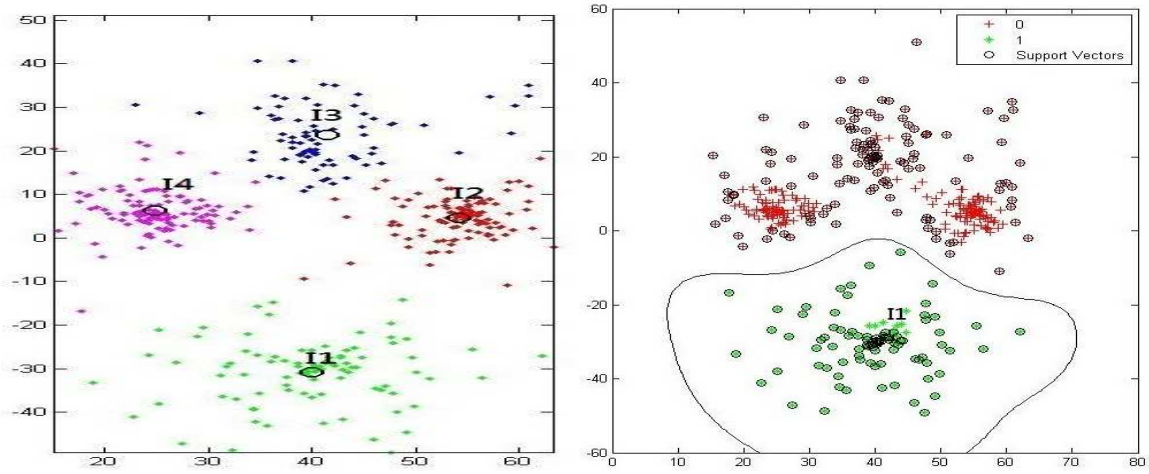
3. 9-10 為判斷是否要往下一層繼續分類，如果是則重覆 4-8。

4. 11 將每一層的特徵函數建構成 Vocabulary Tree，每一個特徵函數為 Vocabulary Tree 上面的節點，最下面的節點稱為葉節點。

整個流程如圖 3.1 的流程圖所示。

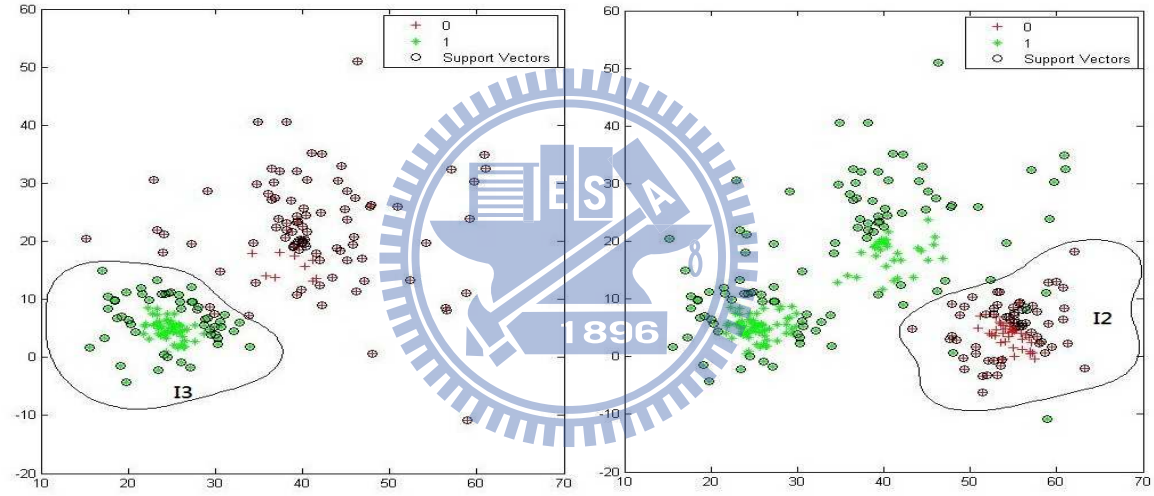
下面使用一個實際的範例來說明 Hierarchical SVM 演算法。將以上小節四類別混合的資料為例子來說明：

1. 圖 3.2(A)的範例為四個影像的例子，四個影像為 I_1、I_2、I_3與 I_4，特徵點假設為二維，圖上的黑色圓圈為中心點，設定每層 Hierarchical SVM 的類別數目為 2 以及階層數目為 2。



(A) (B)

圖 3.2 Hierarchical SVM 演算法分類範例 I



(A) (B)

圖 3.3 Hierarchical SVM 演算法分類範例 II

2. 經由影像分類之後，I_2 、I_3與I_4為一類，I_1獨自為一類，經過 SVM 分類之後，得到圖 3.2(B)的結果。
3. 根據上面 9-10 的判斷，I_2 、I_3與I_4重覆 4-8 的步驟再進行分類。
4. 經由影像分類之後，I_3與I_4為一類，I_2獨自為一類，經過SVM分類之後，得到圖3.3(A)的結果。
5. 根據上面 9-10 的判斷，I_3與I_4重覆 4-8 的步驟再進行分類。
6. 經由影像分類之後，I_3與I_4各為一類，經過 SVM 分類之後，得到圖 3.3(B)的結果。

7. 最後使用圖 3.2(B)、圖 3.3(A)與圖 3.3(B) SVM 所計算的決策函數做為 Vocabulary Tree 的節點，建立 Vocabulary Tree 以利之後的比對。

3.3. 影像比對

使用 Hierarchical SVM 演算法建立 Vocabulary Tree 之後，就可以利用建立的 Vocabulary Tree 進行影像比對。其詳細步驟如下：

1. 搜尋影像是由 m 個特徵點所構成 S 為式(62)：

$$S=\{f^s_1,...,f^s_m \mid f^s_j \subset R^d\} \tag{62}$$

搜尋影像使用 SURF 演算法的找出特徵點與特徵點描述式，d 等於 64，並將 S 代入 Vocabulary Tree，從 Vocabulary Tree 最上層的根節點開始。

2. 使用節點上 SVM 的決策函數找出搜尋影像所有特徵點的決策值，也就是特徵點所分到的類別。

$$D_f(f^s_k)=\mathrm{sgn}(\sum a^j_i y^j_i k(f^j_i, f^s_k)+b) \tag{63}$$

3. 計算相同類別上，搜尋影像特徵點的數目，最大數目類別上的特徵點形成集合 S'

$$S'=\{f^{s'}_i,...,f^{s'}_{m'} \mid f^{s'}_j \subset R^d, 1\le i\le m'\le m\} \tag{64}$$

4. 如果最大類別的節點下面還有其他節點，將 S' 重覆步驟 2。
5. 反之最大類別的節點下面沒有其他節點，此時最多的特徵點將落於葉節點，如果葉節點上只有一張影像，則此張影像為相似影像，反之如果葉節點上有多張影像，使用 SVM 演算法再將影像分類，並使用分類之後的結果來判斷正確的相似影像。

本論文的比對方式與[9]方法有很大的不同，[9]方法使用 TF-IDF 的計算，但是本論文的方式由於分類比較正確，所以只須要將找出最多特徵點的葉節點，最後在找出葉節點上最相似的影像；如果葉節點上的影像只有一張影像，此張影像就是最相似的影像，反之必須判斷最相似影像，判斷的方式可以還是 SVM 演算法，或是 nearest neighbor、K-means 演算法等等，也就是說將所有葉節點上的影像在使用 SVM，nearest neighbor、或 K-means 演算法再分類，之後就可以正確地判斷出相似影像，由於本論文之前已經使

用了 SVM 演算法來做的分類，為了方便起見，影像比對也使用 SVM 演算法來分類。

3.4. 分析

3.4.1. 比對正確率

由於 Hierarchical SVM 演算法能夠有效地將影像分類，所以可以得到更好的比對正率。下面利用一個範例舉例來說明。

還是以一個實際有四群混在一起的資料為例子，將此資料使用 Hierarchical K-means 演算法與 Hierarchical SVM 演算法建立 Vocabulary Tree，不論是 Hierarchical K-means 演算法或 Hierarchical SVM 演算法，每層分類數目為 2 以及階層數目為 1；假設 Hierarchical K-means 演算法剛好將兩群資料分成兩半，Hierarchical SVM 演算法將兩群資料切開如圖 3.4 所示，其中圖 3.4(A)為 Hierarchical K-means 演算法的結果，圖 3.4(B)為 Hierarchical SVM 演算法的結果。

如果相似於 I1 與 I4 的資料，利用 Hierarchical K-means 演算法所建立的 Vocabulary Tree 進行計算 TF-IDF 進行比對，比對的正確率會是 100%；但是如果是相似於 I2 與 I3 的資料用同樣的方法來進行比對，可能就會有誤判的情況，所以正確率將是 50%，因此比對正確率的期望值將會是 75%。

同樣地情況，使用 Hierarchical SVM 演算法來比對，由於每個分類上有兩個明顯的資料，在比對之前還必須將資料分開才能分辨，於是使用 SVM 演算法再分類，如圖 3.5 所示。

因此不論是 I1、I2、I3 與 I4 的資料來比對，比對正確率的期望值將會是 100%，也因此 Hierarchical SVM 演算法可以得到更好的比對正確率。

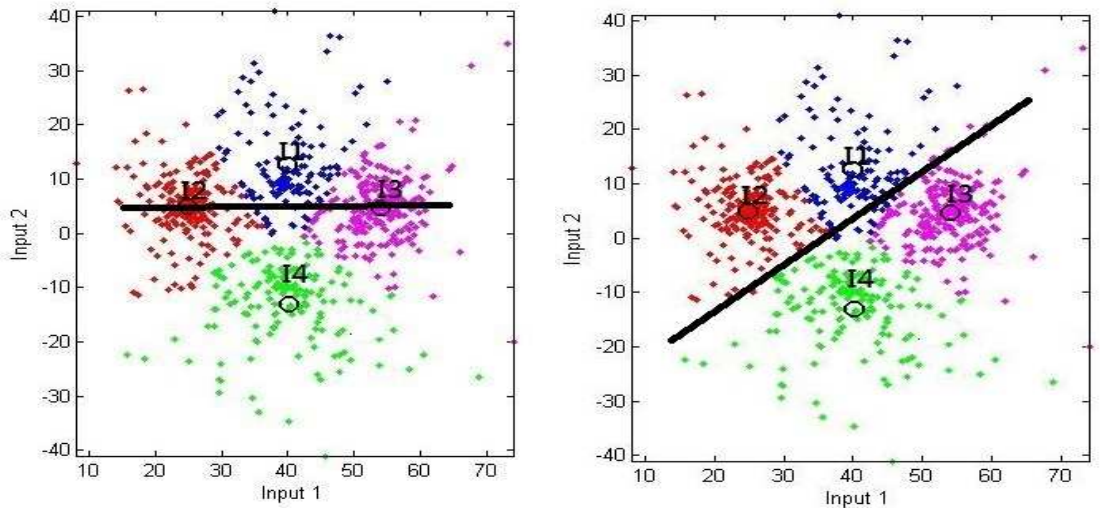


圖 3.4 Hierarchical K-means 與 Hierarchical SVM 分類示意圖

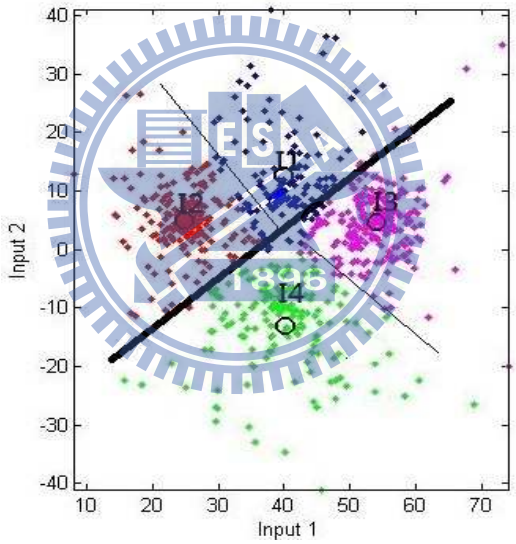


圖 3.5 Hierarchical SVM 葉節點再分類示意圖

3.4.2. 葉節點上比對複雜度比較

本小節將分析[9]與本論文方法的比對複雜度。首先，不論是[9]還是本論文的方法，都是將所有要比對的特徵點，找到 Vocabulary Tree 的葉節點上，在本論文的方法中，只取每層最多特徵點的節點往下尋找，也許在尋找葉節點會有一點優勢，但為了比較方便起見，假設尋找葉節點的複雜度兩個方法是一樣的，於是對於複雜度的比較來說，在

[9]方法是誤差和的計算，而本論文則是分類的比較。

[9]方法的比對方式是計算誤差和為分數，將葉節點上比對影像特徵點的 TF-IDF，減去在葉節點所有影像的 TF-IDF 形成所謂的誤差，然後將每張影像所有葉節點的誤差值加起來形成誤差和，誤差和最小的是就是相似的影像；如果將計算誤差的複雜度定義為 C_{error} ，以及每層分類的數目為 K 與階層數目為 L ，於是[9]方法的複雜度為比對影像葉節點數目的期望值 $E(X)$，乘以每個葉節點上影像數目的期望值 $E(N_{LT})$ 再乘以 C_{error} ，如式(65)所示。

$$Complexity_{[9]} = E(X) \cdot E(N_{LT}) \cdot C_{error} \tag{65}$$

當分類數目 K 與階層數目 L 很小的情況下，使用 Hierarchical K-means 演算法會使得 $E(X)$ 剛好等於葉解點的數目 K^L ，以及 $E(N_{LT})$ 剛好等於影像的數目 N ，於是比對的複雜度為式(66)所示。

$$Complexity_{[9]} = K^L \cdot N \cdot C_{error} \tag{66}$$

本論文的比對方式是找到最多特徵點所在的葉節點，如果葉節點上只有一個影像，此影像就是相似影像，此時複雜度可以視為 0，但是如果有一個以上的影像，就必須進行影像比對；假設使用 SVM 演算法分類，所有的特徵點必須進行 SVM 特徵函數運算，如果將運算 SVM 特徵函數的複雜度定義為 C_{SVM}，於是本論文方法的複雜度為比對影像特徵點數目的期望值 $E(N_f)$ 乘以每個葉節點上影像數目的期望值 $E(N'_{LT})$ 再乘以 C_{SVM} ，如式(67)所示。

$$Complexity = E(N_f) \cdot E(N'_{LT}) \cdot C_{SVM} \tag{67}$$

其中如果分類可以做的很好的情況之下，可以將 $E(N'_{LT})$ 估計為影像數目 N 除以葉節點數目 K^L ，於是複雜度為式(68)所示。

$$Complexity = N_f \cdot \frac{N}{K^L} \cdot C_{SVM} \tag{68}$$

經過式(66)與式(68)的整理之後，可以得到當葉節點數目大於特徵點數目的平方根

的情況下，使用 Hierarchical K-means 演算法的葉節點比對複雜度會大於 Hierarchical SVM 演算法，如式(69)所示。

$$K^L > \sqrt{\frac{N_f \cdot C_{SVM}}{C_{error}}} \tag{69}$$

根據式(69)，葉節點數目 K^L 只要大於 $\sqrt{\frac{N_f \cdot C_{SVM}}{C_{error}}}$ 之後，本論文在葉節點上面的比對複雜度就會小於[9]方法，而且當葉節點數目 K^L 超過影像數目 N 的一定比例之下，本論文在葉節點上之的比對複雜度將會是 0，這並不是代表本論文的整體地比對複雜度降低，只是比對的複雜度轉移到 Vocabulary Tree 上了，相反地使用[9]方法，葉節點數目 K^L 的增加，除了 Vocabulary Tree 上的比對複雜度增加，在葉節點的複雜度也跟著增加，使得整體的複雜度都增加，其最大的原因就是因為使用[9]方法無法有效地將影像正確地分類，葉節點數目 K^L 的增加，無法使得葉節點上的影像數目有效地減少所造成的，而為了提昇比對正確率，葉節點數目 K^L 的增加勢在必行，最後本論文將會有比較好的比對正確率與複雜度。

最後，再利用一個範例來說明，有 60 張影像每張影像平均有 600 個特徵點的資料，要比對的影像也是有 600 個特徵點，如果分類數目是為 4 而且階層數目為 2，此時無論如何是[9]或是本論文方法，都會有 16 個葉節點；使用 Hierarchical K-means 演算法做分類時，不論是影像資料庫還是比對影像來說，影像無法集中於單一節點上，於是葉節點數目將是 16 而且每個葉節點上的影像數目為 60，在此假設 C_{error} 與 C_{SVM} 皆為 1，於是 Hierarchical K-means 演算法的複雜度為式(70)。

$$Complexity_{[9]} = 16 \cdot 60 \cdot 1 \cdot 60 = 57600 \tag{70}$$

反之使用 Hierarchical SVM 演算法，假設影像可以被分類的很平均，於是每個葉節點上的影像數目可以視為 60 除以 16 為 3.75，於是 Hierarchical SVM 演算法的複雜度為式(71)。

$$Complexity_{[SVM]} = 600 \cdot 3.75 \cdot 1 \cdot 60 = 135000 \tag{71}$$

但是如果將分類數目增加為 1，此時葉節點數目將變成 25 而且每個葉節點上的影像數目還是為 60，每個葉節點上的影像數目則降為 60 除以 25 近似於 2.4，於複雜度將分別變成 90000 與 86400，可見當分類數目或階層數目增加，[9]方法的複雜度將倍數增加而本論文倍數減少；以這個例子來說，當葉節點的數目超過 25 之後，本論文的複雜度將小於[9]方法。

事實上如果能夠很有效地分類的情況下，理論上本論文的方法只需要 60 個葉節點，來分別儲存 60 個影像，而且此時一但找到葉節點之後，其影像就是相似的影像，比對的複雜度為 0，所以當分類數目增加的情況，本論文的方法葉節點比對複雜度將接近於 0。

另外，上面的範例有 600 乘以 60 也就是 36000 個特徵點，根據本論文的實驗結果，要使用 Hierarchical K-means 演算法得到比較好的辨識率，葉節點數目將遠大於 25，也因此 Hierarchical SVM 演算法的比對複雜度將小於 Hierarchical K-means 演算法。

4. 實驗結果

4.1. 實驗環境

在本章節中，依照[9]與本論文所提出的 Hierarchical SVM 演算法，實作程式進行模擬實驗，並比較與分析兩方法的實驗結果。

首先，實驗的開發環境如下：

1. 程式語言為 C++，使用 Microsoft Visual Studio 2008 做為開發工具。
2. 測試電腦為 Intel Core i5-520M 2.4GHZ、4G RAM 以及作業系統為 Windows 7。

再者使用的 Open Source library 有：

1. OpenCV[22]：OpenCV 是 Intel 開發的影像處理軟體，本論文使用 OpenCV 處理影像的讀寫。
2. OpenSURF[21]：OpenSURF 實作了 SURF 演算法，本論文用於萃取影像的特徵點以及特徵點的描述式。
3. K-means++ library[23]：David Arthur 實作 K-means++演算法，本論文用於影像的初步分類。
4. LibSVM[20]：台灣大學林智仁(Chih-Jen Lin)教授實做 SVM 演算法的函式庫，本論文用於 SVM 分類。

最後，本論文實驗所採用的影像資料庫為方法[9]所提供的影像資料庫[24]，影像資料庫有 2550 個類別，相同類別中有四張不同的相似影像，四張影像分別是相同物體不同角度與大小的影像。

實驗時從相同類別選取一張影像做為搜尋影像，另外選取一張影像構成 training 影像資料庫，使用[9]或本論文的方法建立 Vocabulary Tree，然後利用 Vocabulary Tree 搜尋與搜尋影像相同類別的影像，如果搜尋後的影像與搜尋影像屬於相同類別，則影像比對為正確，實際正確比對類別數目除以實驗選取類別數目則為比對的正確率。

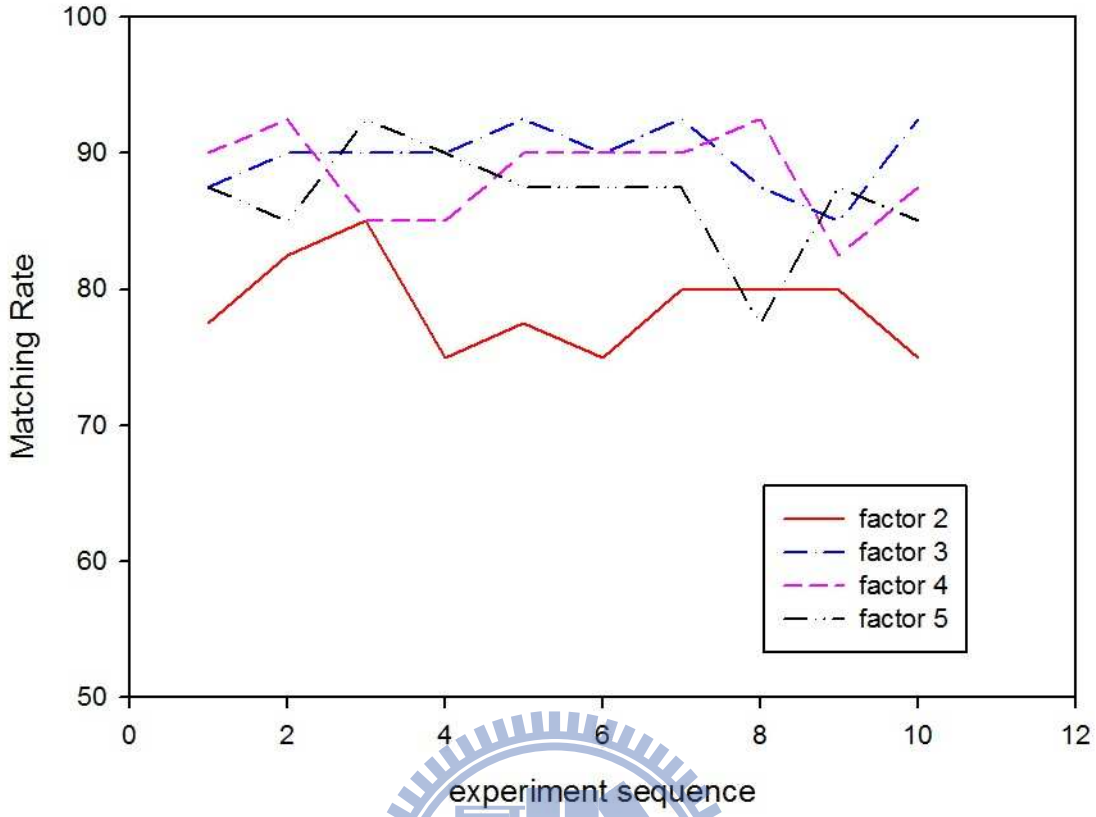


圖 4.1 實驗一 40 種影像不同實驗次數與不同分類數目的比對正確率

4.2. 方法[9]的實驗結果與探討

實驗一本論文將觀察比對正確率是否受到 K-means 演算法因為選取的初始值不同，而每次收斂結果不同的影響。

實驗一的方式是從[24]的影像資料庫中選擇 40 個影像類別，並挑選影像類別中的第一張影像為搜尋影像，另外選取一張影像為 training 影像，根據[9]所提出的方法建立 Vocabulary Tree，Vocabulary Tree 每層的分類數目 2、3、4、5 而且階層數目為 4，重覆做實驗 10 次，所得到的結果如圖 4.1 所示；其中，橫軸代表實驗的次數，縱軸代表比對正確率，每一條曲線是每層不同分類數目在不同實驗次數下的比對正確率。

實驗一的結果可以發現，即使每次實驗輸入搜尋與 training 影像都是相同的影像，但比對正確率卻隨著不同次的實驗而改變，而且不論是增加 Vocabulary Tree 的分類數目或階層數目，都無法改善比對正確率變動的問題。其真正原因就是本論文一開始

所提出 K-means 演算法的問題，因為每次的收斂結果受到選擇中心點而不同，使得建立的 Vocabulary Tree 每次不同，造成比對正確率不同；或者是說，並沒有一個 K-means 問題的演算法可以得到一個唯一最佳解，而造成實驗每次變動這樣的結果。

這樣地結果不但影響建構一個穩定的 Mobile Visual Search 系統，也增加分析 Vocabulary Tree 的困難度，更無法有效地提升比對的正確率；在下面小節根據本論文方法所做的實驗結果，可以看出本論文的方法有效地改善了這個問題。

實驗二本論文將觀察 Vocabulary Tree 分類數目與階層數目的大小，對於比對正確率的影響，一般來說，提高 Vocabulary Tree 分類數目或階層數目，比對正確率也應該跟隨著提高。

實驗二的方式包括了兩個實驗，分別為從[24]的影像資料庫中選擇 20 與 40 個影像類別，並挑選影像類別中的第一張影像為搜尋影像，另外選取一張影像為 training 影像，根據[9]所提出的方法建立 Vocabulary Tree，Vocabulary Tree 的每層分類數目從 2 到 6 而且階層數目從 2 到 10，所得到的結果分別如圖 4.2 與 4.3 所示；其中橫軸代表 Vocabulary Tree 的階層數目，縱軸代表比對率，每一條曲線是代表不同的階層數目，在不同分類數據下的比對正確率。

根據實驗二的結果可以發現，不論是 Vocabulary Tree 的分類數目或是階層數目增加，在一開始增加的時候，都能夠很快提高比對的正確率，以 40 個影像類別的例子來說，原本分類數目與階層數目都為 2 時，其比對正確率只有 40%，不論是分類數目或是階層數目增加 1 時，其比對正確率分別變成 72.5%與 77.5%，都有 30%以上的成長，但是，隨著分類數目與階層數目再增加，比對正確率卻無法快速的提昇，有些情況反而因為分類數目與階層數目的增加，比對正確率有下降的情況，何況比對正確率也受到 K-means 演算法每次不同分類結果的波動，因此隨著影像數目的增加比對正確率也隨著下降，更無法透過分類數目與階層數目的增加而一直提高的比對正確率。

這樣地結果就是本論文之前所提出每分類數目小於真正分類的影響，因為分類時，有些影像被分散在不同的類別上，造成這些被分散在不同類別的影像，無法正確地被辨識，之後的實驗也顯示，本論文的方法有效地改善了這個問題。

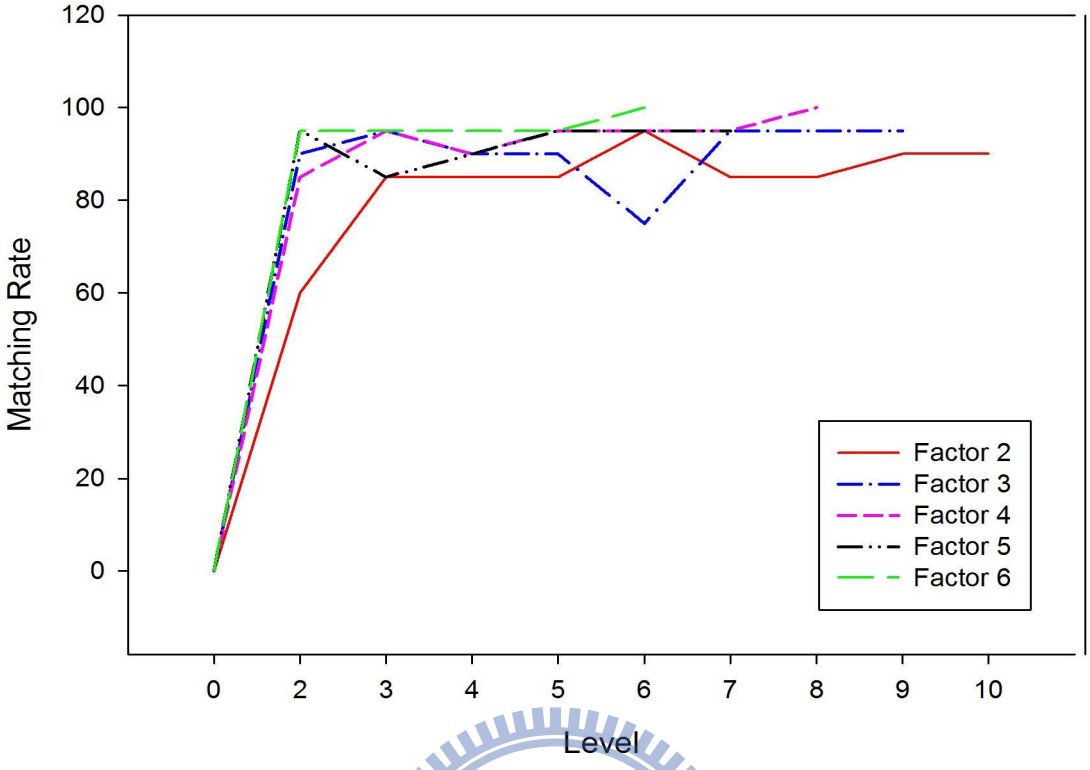


圖 4.2 實驗二之一 20 種影像不同分類數目與不同階層數目的比對正確率

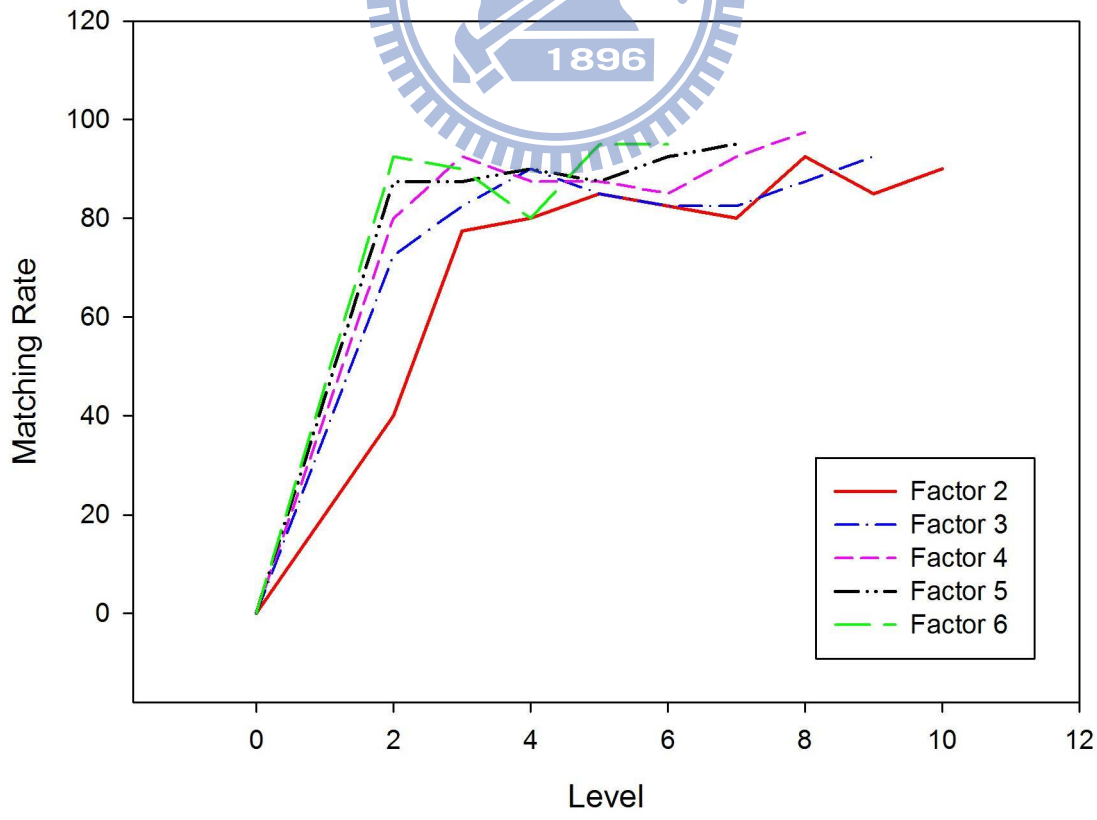


圖 4.3 實驗二之二 40 種影像不同分類數目與不同階層數目的比對正確率

另外，在本論文在進行分類數目或是階層數目實驗的同時，發現如果修改方法[9]中 IDF 的定義，以及計算比對分數的方式，可以達到更好的比對結果，於是下面說明本本論文對於 IDF 定義與計算比對分數方式的修改，並提供實驗結果做為比較。

方法[9]使用 TF-IDF 做為計算比對相似度的基礎，這樣的概念是根據文件搜尋而來；對於文件搜尋來說，IDF 的概念是如果文字在不同文章出現的比率很高，表示這個文字並不重要，因為無法使用這個字來分辨文章，因此 IDF 定義為式(72)。

$$IDF_{LT_i} = \log \frac{N}{N_{LT_i}} \tag{72}$$

對於方法[9]來說，Vocabulary Tree 就是形成影像資料庫的文字，將 TF-IDF 概念用於影像的 Vocabulary Tree，是一很好的做法，但是根據本論文實驗的結果顯示，如果修改 IDF 的定義，在分類數目或是階層數目較小的情況下，就可以得到較高的比對正確率；原因是當 Vocabulary Tree 上葉節點的影像數目等於影像資料的影像數目時，也就是 $N = N_{LT_i}$ 的情況下，IDF_{LT_i} 將等於零，於是在葉節點上所有的比對分數將會是零，因此在葉節點上特徵點多的影像會難以辨識；當分類數目或是階層數目較小的情況下，許多的葉節點將會是影像數目等於影像資料的影像數目，於是本論文將 IDF 的定義改為式(73)。

$$IDF_{LT_i} = \frac{N}{N_{LT_i}} \tag{73}$$

修改 IDF 的定義之後，實驗三本論文觀察新的 IDF 定義如何影響比對正確率；不但希望在分類數目或是階層數目較小的情況下，就可以得到較高的比對正確率，在分類數目或是階層數目較高的情況下，也能夠與舊的 IDF 定義有相同程度的比對正確率。

實驗三的方式包括了使用新與舊的 IDF 定義，首先從[24]的影像資料庫中選擇 40 個影像類別，並挑選影像類別中的第一張影像為搜尋影像，另外選取一張影像為 training 影像，根據[9]所提出的方法建立 Vocabulary Tree，Vocabulary Tree 的每層分類數目從 3 到 5 而且階層數目從 2 到 10，在計算 TF-IDF 時，分別使用新的與舊的 IDF 定義，所得到的結果分別如圖 4.4 所示；其中，橫軸代表 Vocabulary Tree 的階層數目，縱軸代表比對率，每一條曲線是代表不同的階層數目，在不同分類與 IDF 定義下的比對

正確率。

在圖 4.4 的實驗結果中，藍色曲線代表使用新的 IDF 定義，反之紅色曲線代表使用舊的 IDF 定義，由圖上的結果可以發現，當分類數目或是階層數目較小的情況下，新 IDF 定義的比對正確率比舊 IDF 定義的比對正確率要高出許多，而且分類數目或是階層數目較高的情況下，新 IDF 定義的比對正確率與舊 IDF 定義的比對正確率也大致一樣，如此一來就可以降低分類數目或是階層數目，而減少計算量以及比對時間，於是本論文的實驗都採用本論文所提出 IDF 的定義。

最後，方法[9]計算影像相似度是使用 TF-IDF 的誤差和為分數，最小誤差和的影像為相似影像，誤差和分數的計算如式(74)所示。

$$S(q,I_i)=\left|\frac{W_q}{|W_q|}-\frac{W_{I_i}}{|W_{I_i}|}\right| \tag{74}$$

但是根據本論文的實驗結果，當每層分類數目或是階層數目較高的情況下，由於許多影像特徵點被分散在不同節點，使得誤差和的準確性不高，由於在文件搜尋中計算 TF-IDF 之後，常使用餘弦相似度（cosine similarity）來進行相似度的計算，於是本論文採用了餘弦相似度來計算影像相似度的分數，使得分類數目或是階層數目較高的情況下，可以維持一定程度的比對正確率，餘弦相似度的定義如式(75)所示。

$$similarity(q,I_i)=\left|\frac{W_q}{|W_q|}\cdot\frac{W_{I_i}}{|W_{I_i}|}\right| \tag{75}$$

當兩個影像越接近時，餘弦相似度所計算出來的結果越接近 1，反之越接近 0，也就是說，餘弦相似度越大影像就是越相似的影像。

將分數計算修改為餘弦相似度之後，實驗四本論文觀察誤差和與餘弦相似度如何影響比對正確率；不但希望在分類數目或是階層數目較小的情況下，就可以得到相同的比對正確率，在分類數目或是階層數目較高的情況下，餘弦相似度也能夠維持一定程度的比對正確率。

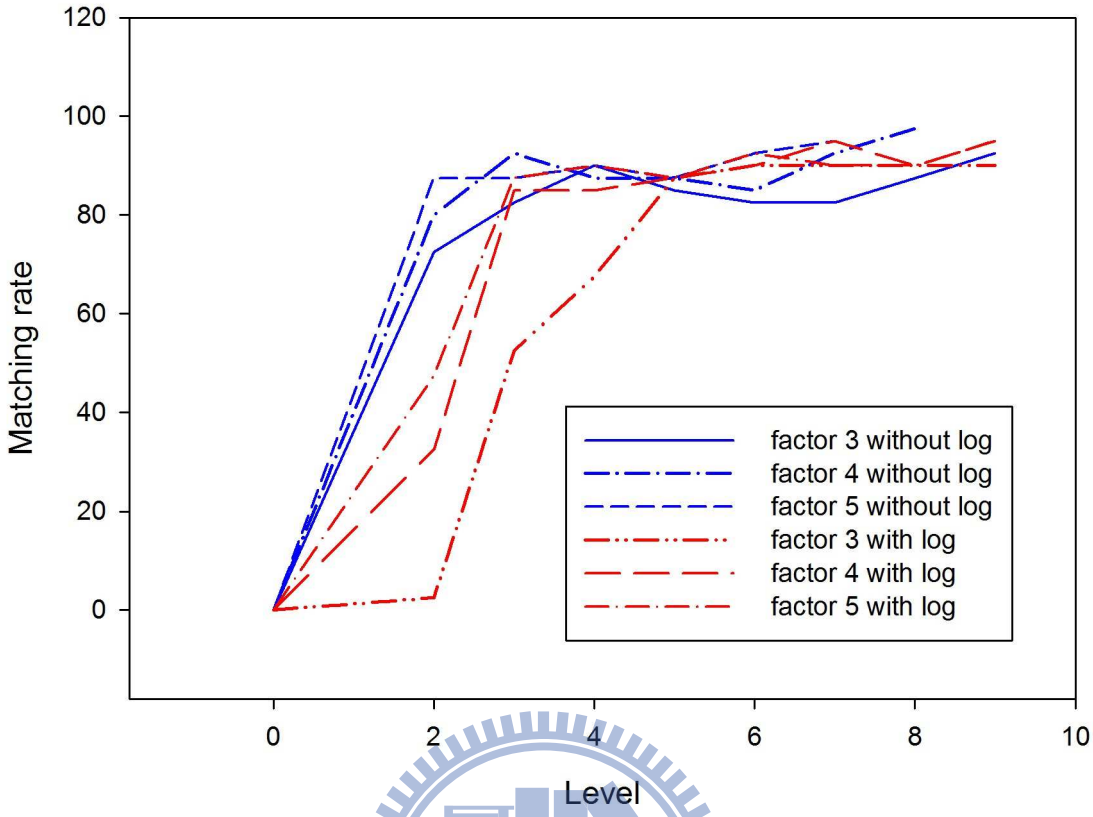


圖 4.4 實驗三 40 種影像不同 IDF 定義的比對正確率

實驗四的方式包括了使用誤差和與餘弦相似度來計算比對分數，首先從[24]的影像資料庫中選擇 40 個影像類別，並挑選影像類別中的第一張影像為搜尋影像，另外選取一張影像為 training 影像，根據[9]所提出的方法建立 Vocabulary Tree，Vocabulary Tree 的每層分類數目從 3 到 5 而且階層數目從 2 到 9，在計算比對分數時，分別使用誤差和與餘弦相似度，所得到的結果分別如圖 4.5 所示；其中，橫軸代表 Vocabulary Tree 的階層數目，縱軸代表比對正確率，每一條曲線是代表不同的階層數目，在不同分類數目與使用誤差和以及餘弦相似度的比對正確率。

在圖 4.5 的實驗四結果中，藍色曲線代表使用餘弦相似度來計算比對分數，反之紅色曲線代表使用誤差和來計算比對分數，由圖上的結果可以發現，當分類數目或是階層數目較小的情況下，餘弦相似度的比對正確率與誤差和的比對正確率要相差不多甚至於還要高，但是在分類數目或是階層數目較高的情況下，餘弦相似度的比對正確率並不會下降反而還有上升，反之使用誤差和的比對正確率甚至降為零，如此一來如果使用誤

差和必須要找到最佳的分類數目與階層數目，而增加計算的複雜度，於是本論文的實驗都採用餘弦相似度來計算比對分數。

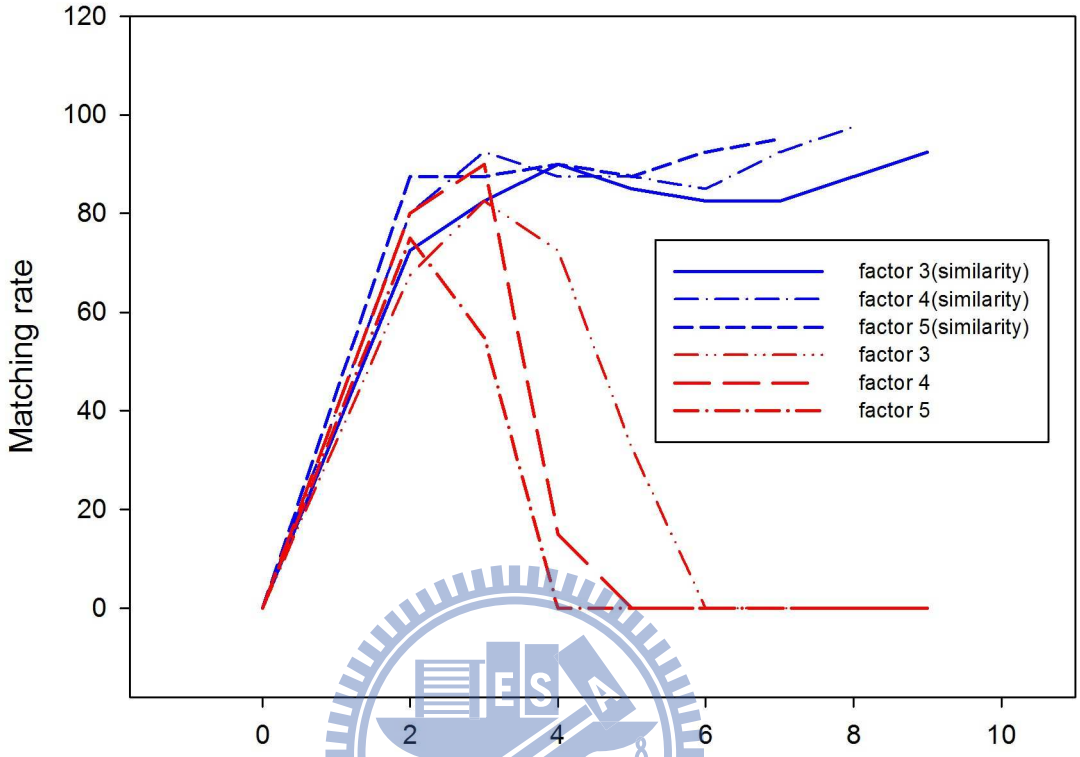


圖 4.5 實驗四 40 種影像使用誤差和以及餘弦相似度的比對正確率

4.3. Hierarchical SVM 演算法的實驗結果與探討

由於比對正確率受到 Hierarchical K-means 演算法的影響導致每次實驗結果不同，實驗五本論文將觀察比對正確率是否受到 Hierarchical SVM 演算法的影響，也產生每次不同實驗比對正確率不同的結果。

實驗五的方式為從[24]的影像資料庫中選擇 20 個影像類別，並挑選影像類別中的第一張影像為搜尋影像，另外選取一張影像為 training 影像，根據 Hierarchical SVM 演算法建立 Vocabulary Tree，Vocabulary Tree 每層的分類數目 2、3、4、5 而且階層數目為 4，重覆做實驗 5 次，所得到的結果如圖 4.6 所示；其中，橫軸代表實驗的次數，縱軸代表比對正確率，每一條曲線是每層不同分類數目在不同實驗次數下的比對正確

率，除了將 Hierarchical K-means 演算法更改為 Hierarchical SVM 演算法之外，其他的步驟跟第一個實驗幾乎是一模一樣的。

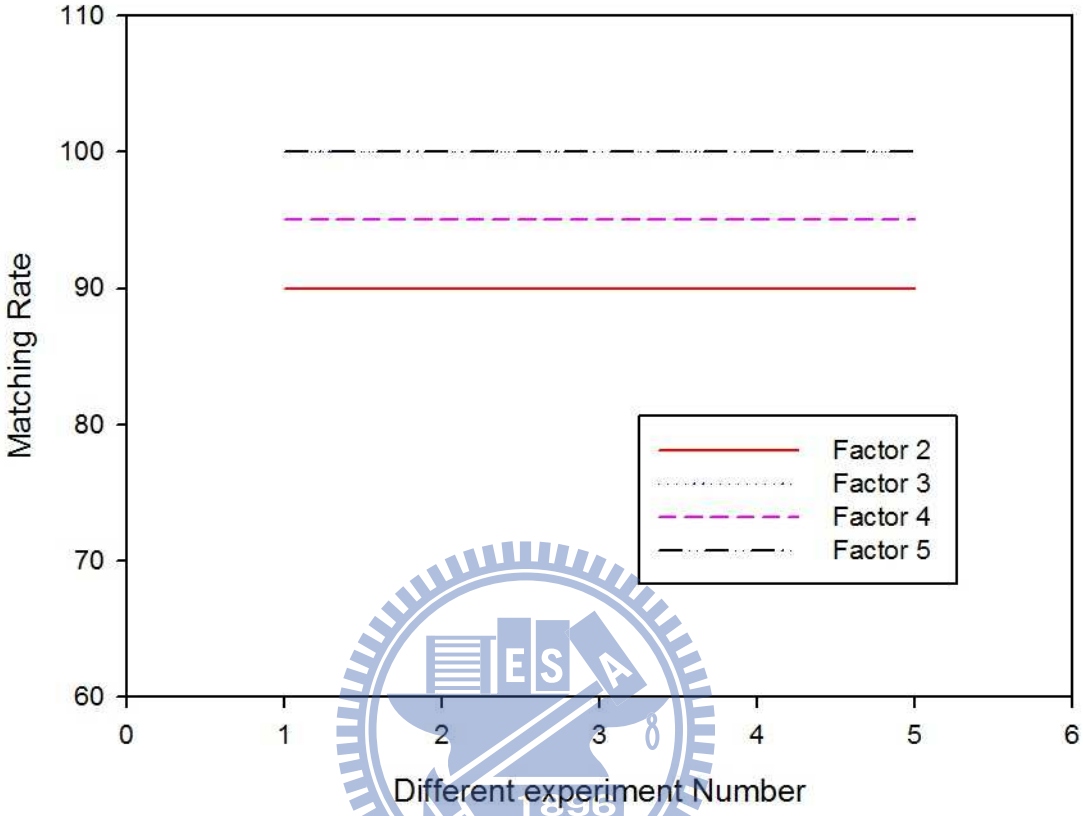


圖 4.6 實驗五 20 種影像不同實驗次數與不同分類數目的比對正確率

由實驗五的結果可以發現，本論文所提出的 Hierarchical SVM 演算法，完全不會受到實驗次數而影響到比對的正確率，相同的輸入可以得到相同的輸出，如此一來可以建構一個穩定的 Mobile Visual Search 系統，也比較容易分析比對正確率的問題。

實驗六本論文將觀察使用 Hierarchical SVM 演算法所建立的 Vocabulary Tree，分類數目與階層數目的大小對於比對正確率的影響，與實驗二做一個比較。

實驗六的方式與實驗二雷同包括了兩個實驗，分別為從[24]的影像資料庫中選擇 20 與 40 個影像類別，並挑選影像類別中的第一張影像為搜尋影像，另外選取一張影像為 training 影像，根據本論文 Hierarchical SVM 演算法建立 Vocabulary Tree，Vocabulary Tree 的每層分類數目從 2 到 5 而且階層數目從 2 到 5，所得到的結果分別如圖 4.7 與 4.8 所示；其中橫軸代表 Vocabulary Tree 的階層數目，縱軸代表比對正確

率，每一條曲線是代表不同階層數目在不同分類數據下的結果。

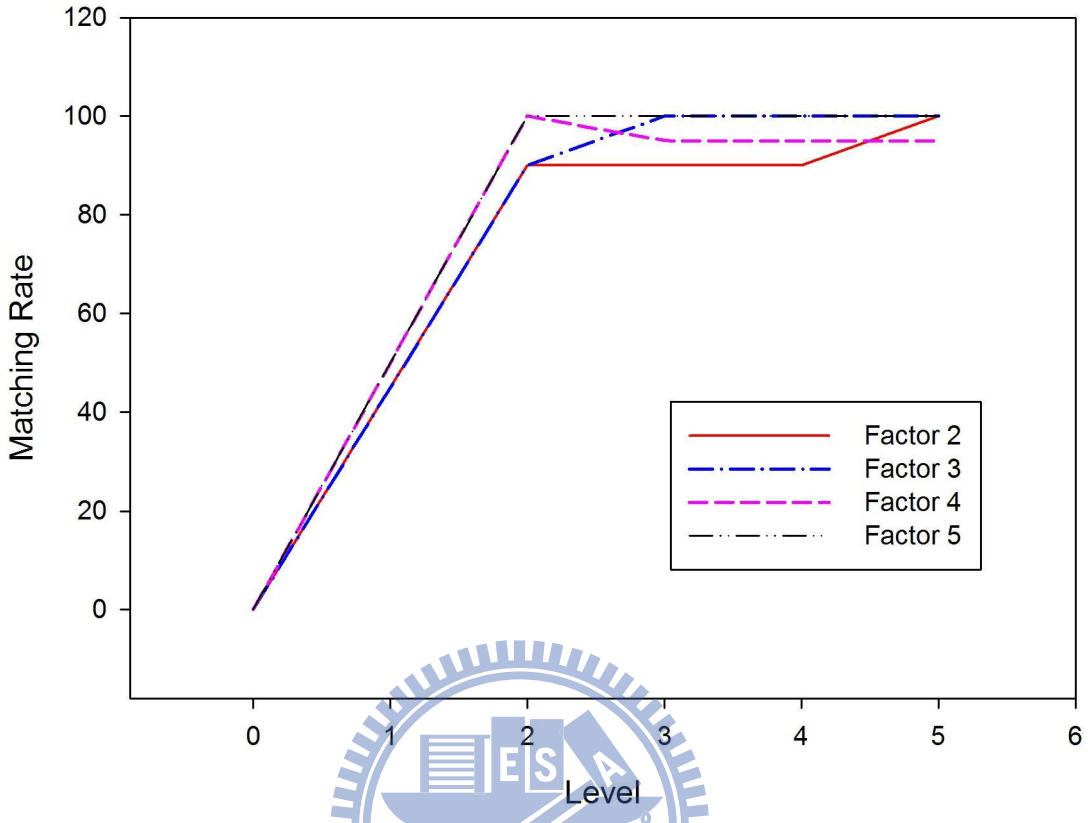


圖 4.7 實驗六之一 20 種影像不同分類數目與不同階層數目的比對正確率

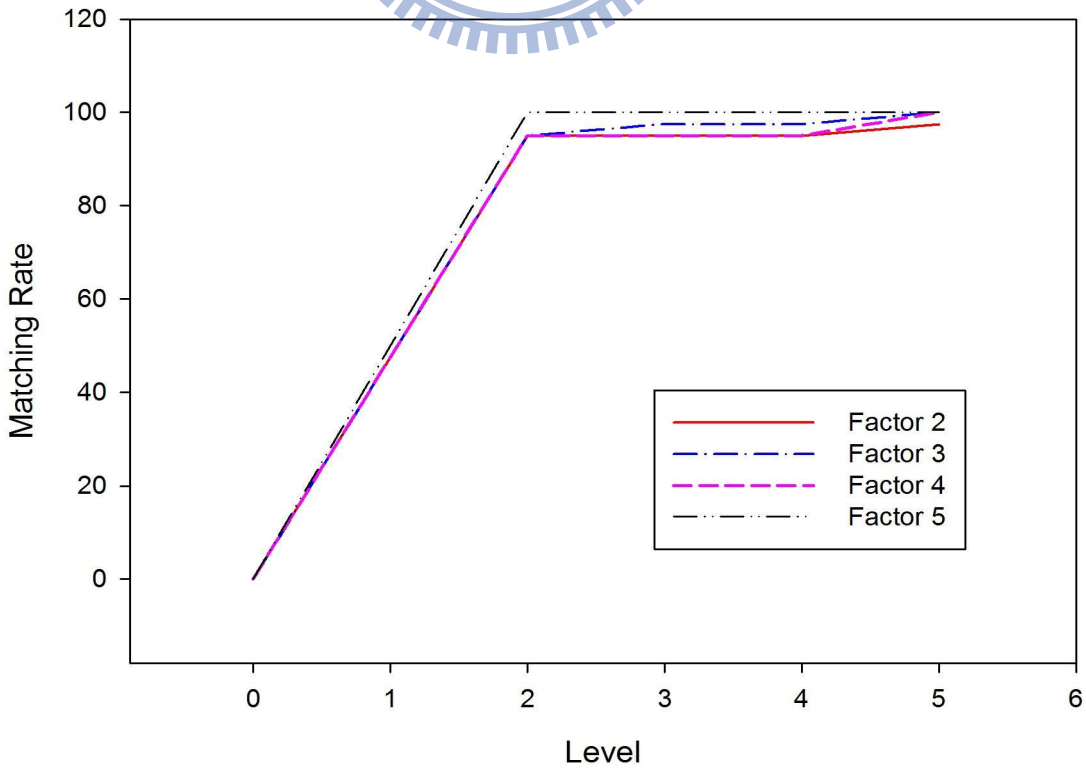


圖 4.8 實驗六之二 40 種影像不同分類數目與不同階層數目的比對正確率

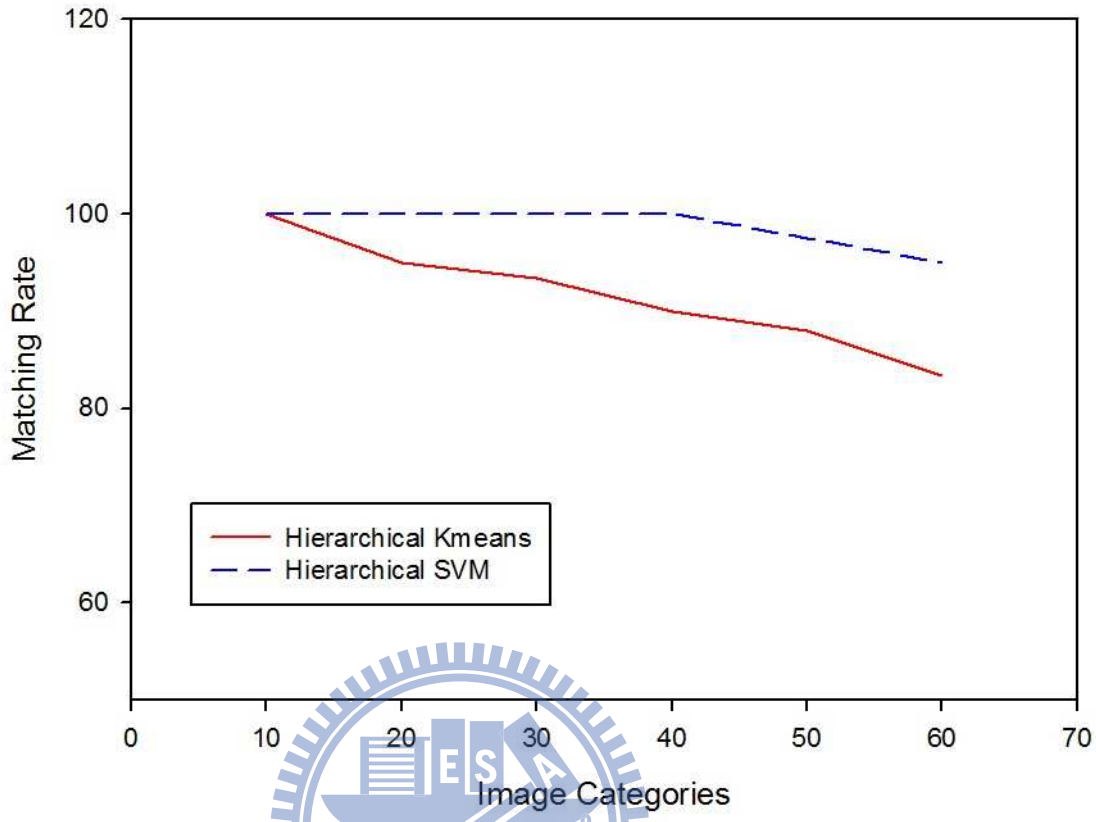


圖 4.9 實驗七比較[9]方法與 Hierarchical SVM 演算法的比對正確率

根據實驗六的結果可以發現，使用本論文所提出的 Hierarchical SVM 演算法，在一開始分類數目與階層數目很小的情況下，就可以得到很高的比對正確率，而且比對正確率隨著分類數目與階層數目的增加，比對正確率大致上也隨著增加，最後比對正確率都可以達到 100%比對正確率。

實驗七本論文將觀察使用[9]方法與 Hierarchical SVM 演算法所建立的 Vocabulary Tree，在不同的影像類別數比對正確率的比較。

實驗七的方式是從[24]的影像資料庫中選擇 10、20、30、40、50 與 60 個影像類別，並挑選影像類別中的第一張影像為搜尋影像，另外選取一張影像為 training 影像，分別使用[9]所提出的方法與 Hierarchical SVM 演算法建立 Vocabulary Tree，Vocabulary Tree 的分類數目 5 而且階層數目為 5，所得到的結果如圖 4.9 所示；其中，橫軸代表影像類別，縱軸代表比對正確率，紅色曲線代表[9]方法所計算出來的比對正

確率，而藍色曲線代表 Hierarchical SVM 演算法所計算出來的比對正確率。

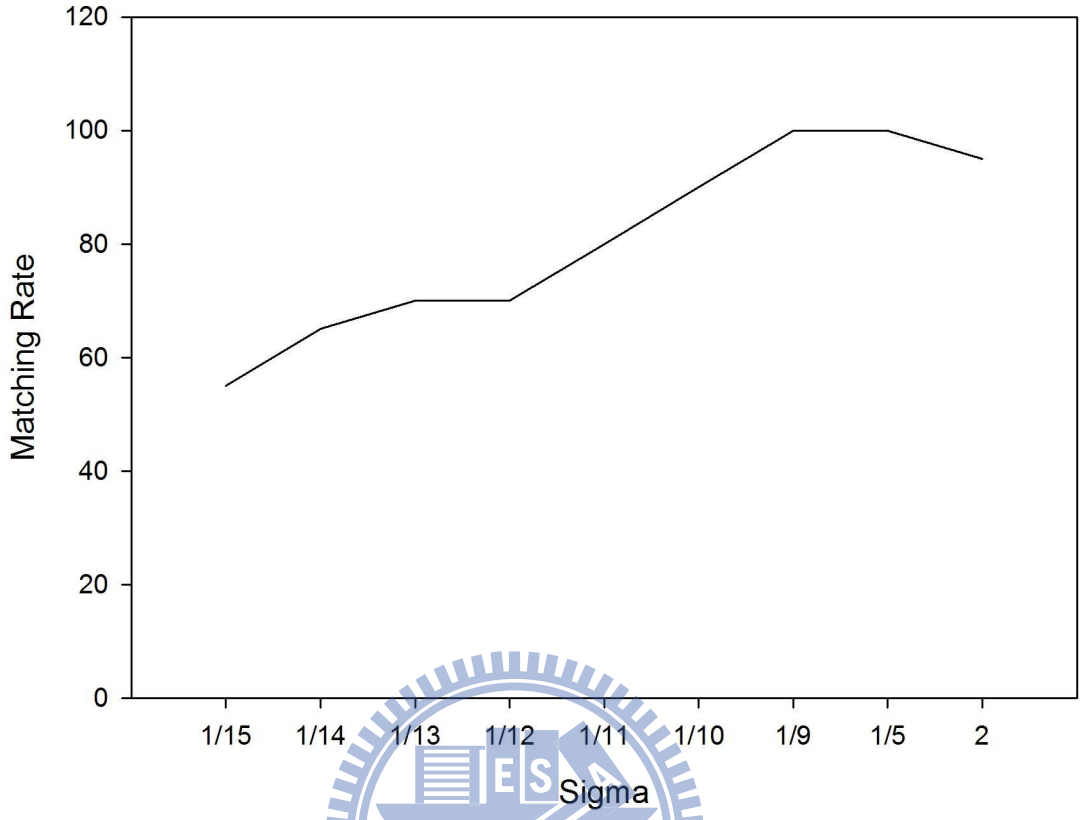


圖 4.10 實驗八 RBF 函數不同 Sigma 值的比對正確率

由實驗七的結果可以發現，本論文的平均比對正確率可以達到 97.6%，而[9]方法的平均比對正確率為 90.5%，可見本論文的方法可以提高比對正確率；另外在影像類別越來越多的時候，本論文的方法也可以維持很好的比對正確率。

最後實驗八本論文將觀察 RBF 函數的 Sigma 值，對於比對正確率的影響。由於本論文的 SVM 演算法使用非線性 RBF 函數，根據之前 SVM 理論的分析，非線性 RBF 函數的 Sigma 值會影響分類區域的大小，Sigma 值越小可以分類的越準確，但是比對錯誤率會增加。

實驗八的方式為從[24]的影像資料庫中選擇 20 個影像類別，並挑選影像類別中的第一張影像為搜尋影像，另外選取一張影像為 training 影像，根據 Hierarchical SVM 演算法建立 Vocabulary Tree，Vocabulary Tree 的分類數目 3 而且階層數目為 2，改變 Sigma 的平方值從 1/15 到 2，所得到的結果如圖 4.10 所示；其中，橫軸代表所採用的

Sigma 平方值，縱軸代表比對正確率。

根據實驗八的結果可以發現，RBF 函數的 Sigma 值確實會影響到比對正確率，當 Sigma 值太小時，比對正確率會下降許多，雖然 Sigma 值減小可以得到較好的分割，但是比對上就難以比對，隨著增加 Sigma 值，比對正確率也隨之提高，但是 Sigma 如果太大，一開始分割的錯誤就增加，最後也會影響比對率，根據多次以及不同影像數目實驗，本論文 RBF 函數 Sigma 平方值為 1/6。

4.4. 葉節點比對複雜度的實驗結果與探討

實驗九的方式是從[24]的影像資料庫中選擇 60 個影像類別，並挑選影像類別中的第一張影像為搜尋影像，另外選取一張影像為 training 影像，根據[9]與本論文所提出的方法建立 Vocabulary Tree，Vocabulary Tree 每層的分類數目 4、5、6、7 而且階層數目為 2、3、4、5，所得到的結果分別為圖 4.11、4.12 與 4.13 所示；其中圖 4.11 與 4.12，橫軸代表階層數目，縱軸代表葉節點上比對的複雜度，每一條曲線是每層不同分類數目在不同階層數目的比對複雜度；圖 4.13，橫軸代表階層數目，縱軸代表比對正確率，每一條曲線是每層不同分類數目使用不同演算法在不同階層數目的比對正確率。

由圖 4.13 可知，隨著分類數目與階層數目的增加，不論是 Hierarchical SVM 或是 Hierarchical K-means 演算法，其比對正確率也隨著上升，但是隨著階層數目的不斷增加，使用 Hierarchical K-means 演算法不但無法提高比對正確率，反而比對正確率下降，其原因在於，當階層數目過多的時後，所有的葉解點數目已經超過了影像特徵點的數目，所有的特徵點有可能會散落在每個葉解點而造成比對困難；反之 Hierarchical SVM 演算法使用影像數目決定是否向下分類的條件，因此即使設定很大的階層數目也因為達到結止條件不會再向下分類，也不會有比對正確率下降的問題。

由圖 4.12 可知當分類數目為 4 與階層數目為 2 時，使用 Hierarchical K-means 演算法的複雜度為 59540，近似於 57600 為葉節點數目 16 乘以葉節點影像數目 60 與影像數目 60，也就是說，之前的估計在分類數目為與階層數目不大的情況下是十分正確的；但是隨著分類數目為與階層數目的增加，葉節點的數目也因此增加，葉節點數目與

葉節點影像數目隨著減少，複雜度的增加也因此減少，當分類數目為與階層數目的增加到一定程度之後，複雜度反而因此而減少。另外在圖 4.11 上，當分類數目為 4 與階層數目為 2 時，使用 Hierarchical SVM 演算法的複雜度為 124564，與估計的 135000 為特徵點平均 600 數目乘以葉節點影像平均數目 3.75 與影像數目 60 還要小，其原因應該是受到在葉節點上的影像並非平均的分佈有關係，有些影像不需要比對的影響。

隨著分類數目與階層數目的增加，Hierarchical SVM 演算法的比對複雜度隨著下降，但這是指葉節點上的比對複雜度，如果考慮從 Vocabulary Tree 根節點到葉節點的複雜度，事實上整體地比對複雜度應該是不變的，在葉節點上減少的比對複雜度其實是轉嫁到 Vocabulary Tree 上，相反地使用 Hierarchical K-means 演算法葉節點上的比對複雜度無法完全地轉嫁到 Vocabulary Tree 上，使得整體地比對複雜度隨著分類數目與階層數目的增加而增加，為了提高比對的正確率，Hierarchical K-means 演算法不得不選取較高的分類數目與階層數目，最後整體來說，使用 Hierarchical SVM 演算法的複雜度不會高於 Hierarchical K-means 演算法。

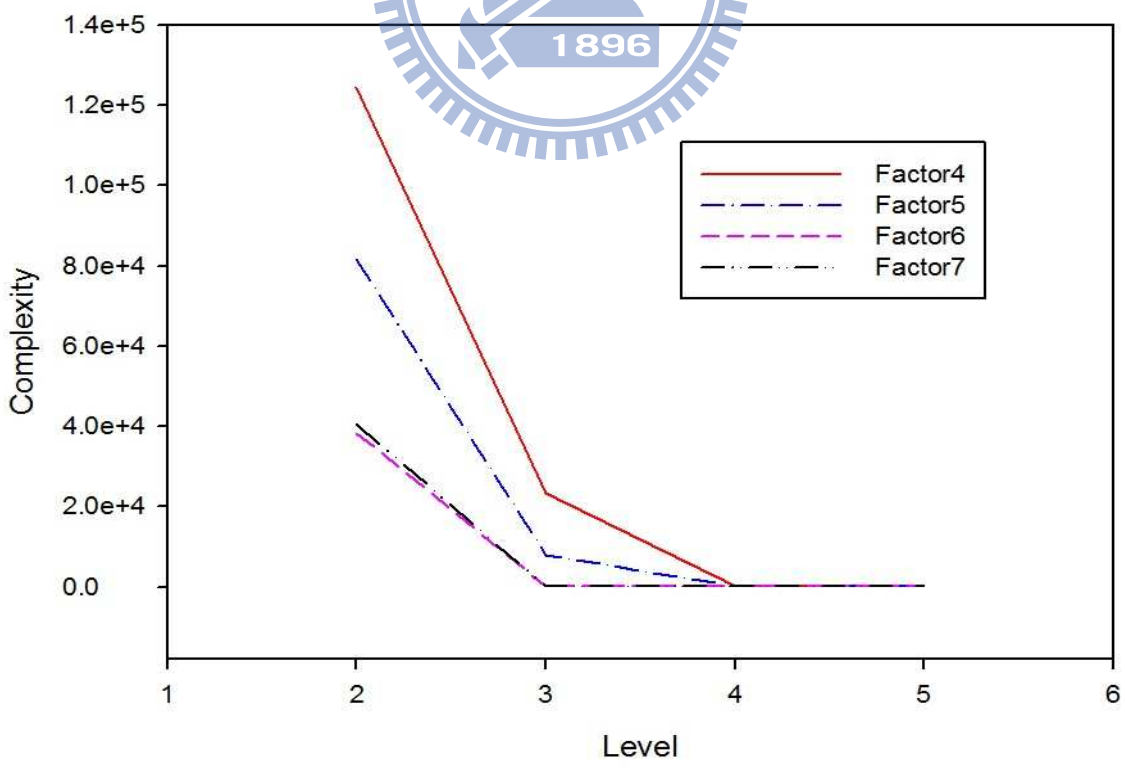


圖 4.11 Hierarchical SVM 不同分類與階層數目的 Complexity

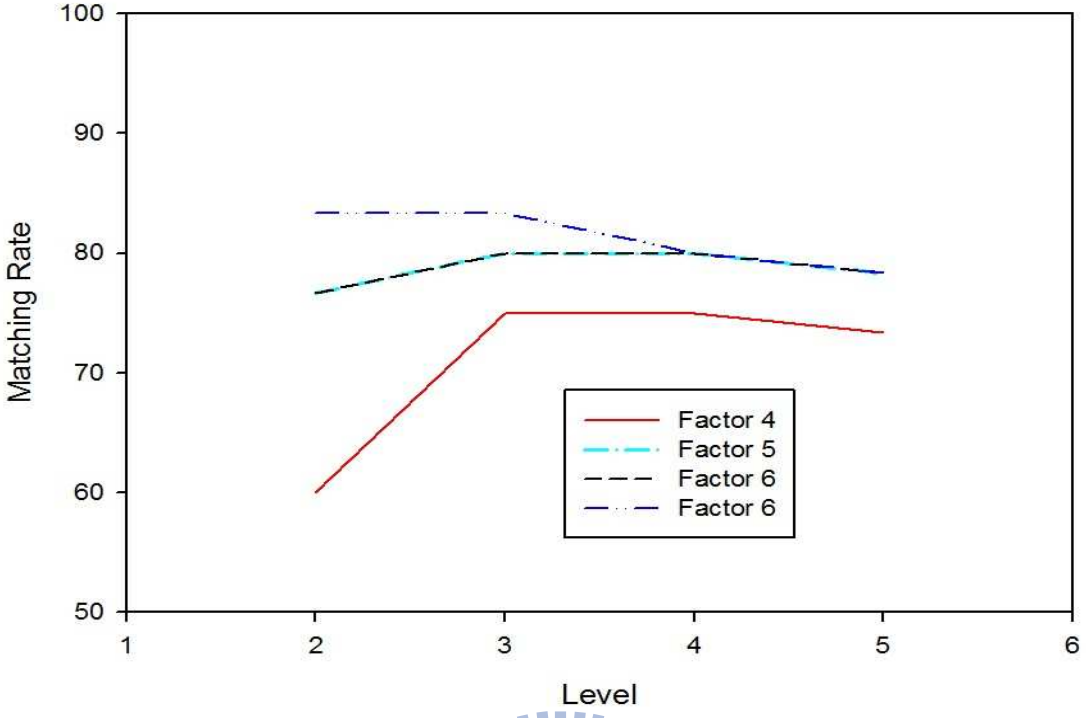


圖 4.12 Hierarchical K-means 不同分類與階層數目的 Complexity

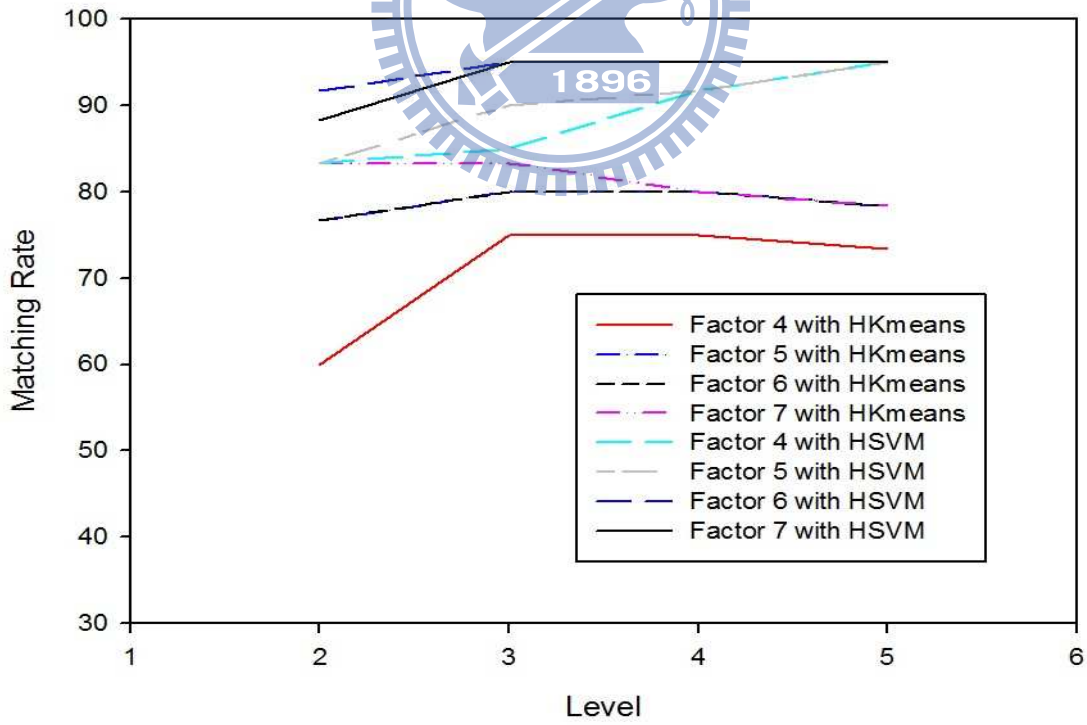


圖 4.13 60 種影像不同分類數目與不同階層數目的比對正確率

5. 結論與未來研究方向

使用本論文所提出的 Hierarchical SVM 演算法建立 Vocabulary Tree，不論在理論上或是實驗結果，在影像的辨識率上，都比原本使用 Hierarchical K-means 演算法提高許多，也解決了 K-means 演算法造成實驗結果隨著實驗變動的問題；不但如此，使用 Hierarchical SVM 演算法所建立的 Vocabulary Tree，在影像比對上，也比 Hierarchical K-means 演算法的 Vocabulary Tree 正確與簡單許多，基本上 Vocabulary Tree 上就已經包含了比對的功能，當分類數目與階層數目超過影像數目時，經過 Vocabulary Tree 的搜尋之後就可以找到正確地相似影像，不需要在葉節點上經過其他運算才能分辨相似影像，也因此使得整體地比對複雜度不會隨著分類數目與階層數目的增加而增加。

未來，針對 Hierarchical SVM 演算法，還有許多可以值得研究的地方如下：

1. 在本論文中，影像初步分類是使用 K-means++演算法，雖然使用 K-means++演算法比 K-means 演算法更接近最佳化 K-means 問題的結果，但是真正所需要的分類方式也可能不一定是 K-means 最佳化的結果，初步分類結果的好壞，將會影響到之後細部分類的難易度以及辨識的容易度，相信是可以值得再研究的地方。
2. SVM 參數決定了影像分類的好壞，以本論文使用的 RBF Kernel 函數來說，Sigma 值小可以很正確地進行分類，但辨識時可能無法達到很好的效果；SVM 參數或是不同 Kernel 函數對於 Vocabulary Tree 的影響，是一值得探討的問題。
3. 本論文對於 SVM 演算法的運算複雜度並無探討，另外對於其他 supervised machine learning 的演算法是否適合取代 SVM 演算法也無研究，也許其他 supervised machine learning 的演算法可以更容易分類，更正確的分辨影像，或是更低運算的複雜度都是可以再深入研究的問題。
4. 在 Tom Yeh[26]中，Adaptive Vocabulary Tree 的方法可以動態加入新的影像，相信本論文的 Hierarchical SVM 演算法未來也可以應用在動態影像的增加。

參考文獻

[1] http://scien.stanford.edu/pages/conferences/mvs/presentations/WorkshopMVSDec2009.pdf

[2] V. Chandrasekhar, G. Takacs, D. Chen, S. Tsai, and B. Girod, "Transform Coding of Feature Descriptors," in VCIP, 2009.

[3] David Chen, Sam Tsai, Vijay Chandrasekhar, Gabriel Takacs, Jatinder Singh, and Bernd Girod, "Tree histogram coding for mobile image matching", IEEE Data Compression Conference (DCC), Snowbird, Utah, March 2009.

[4] V. Chandrasekhar, G. Takacs, D. M. Chen, S. S. Tsai, R. Grzeszczuk, and B. Girod. CHoG: Compressed Histogram of Gradients - A low bit rate feature descriptor. In Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Miami, Florida, June 2009.

[5] K. Grauman and T. Darrell. The Pyramid Match Kernel: Efficient Learning with Sets of Features. Journal of Machine Learning Research (JMLR), 8 (Apr): 725--760, 2007.

[6] K. Grauman. Matching Sets of Features for Efficient Retrieval and Recognition, Ph.D. Thesis, MIT, 2006.

[7] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," International Journal of Computer Vision, vol. 60, no. 2, pp. 91 - 110, 2004.

[8] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded Up Robust Features," in ECCV (1), 2006, pp. 404 - 417.

[9] D. Nist´er and H. Stew´enius, "Scalable Recognition with a Vocabulary Tree," in IEEE Conference on Computer Vision and Pattern Recognition (CVPR),

vol. 2, 2006, pp. 2161－2168.

[10]Lloyd, S. P. (1957). "Least square quantization in PCM". Bell Telephone Laboratories Paper. Published in journal much later: Lloyd., S. P. (1982). "Least squares quantization in PCM". IEEE Transactions on Information Theory 28 (2): 129－137.

[11]Arthur, D. and Vassilvitskii, S. (2007). "k-means++: the advantages of careful seeding". Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms. pp. 1027--1035.

[12]Aloise, D.; Deshpande, A.; Hansen, P.; Popat, P. (2009). "NP-hardness of Euclidean sum-of-squares clustering". Machine Learning 75: 245－249.

[13]Dasgupta, S. and Freund, Y. (July 2009). "Random Projection Trees for Vector Quantization". Information Theory, IEEE Transactions on 55: 3229－3242.

[14] W. Fernandez de la Vega, Marek Karpinski, Claire Kenyon, and Yuval Rabani. Approximation schemes for clustering problems. In STOC ’03: Proceedings of the thirty-fifth annual ACM symposium on Theory of computing, pages 50－58, New York, NY, USA, 2003. ACM Press.

[15] Sariel Har-Peled and Soham Mazumdar. On coresets for k-means and k-median clustering. In STOC ’04: Proceedings of the thirty-sixth annual ACM symposium on Theory of computing, pages 291－300, New York, NY, USA, 2004. ACM Press.

[16] Nello Cristianini and John Shawe-Taylor. An Introduction to Support Vector Machines and other kernel-based learning methods. Cambridge University Press, 2000. ISBN 0-521-78019-5.

[17] A Tutorial on Support Vector Machines for Pattern Recognition by Christopher J. C. Burges. Data Mining and Knowledge Discovery 2:121－167, 1998.

[18] Dibike, Y. B., S. Velickov, D. Solomatine, and M. B. Abbott (2001). “Model

Induction with Support Vector Machines: Introduction and Applications." Journal of Computing in Civil Engineering, 15(3), 208-216.

[19] Boser, B.E., Guyon, I., & Vapnik, V.N. (1992). A training algorithm for optimal margin classifiers. InProceedings of the Fifth Annual Workshop of Computational Learning Theory, 5, 144－152. Pittsburgh, ACM.

[20] http://www.csie.ntu.edu.tw/~cjlin/libsvm/

[21] http://www.chrisevansdev.com/computer-vision-opensurf.html

[22] http://opencv.willowgarage.com/wiki/

[23] http://www.stanford.edu/~darthur/kmpp.zip

[24] http://www.vis.uky.edu/~stewe/ukbench/

[25] http://tahiti.mis.informatik.tu-darmstadt.de/oldmis/Research/Projects/

[26] T. Yeh, J. Lee, and T. Darell. "Adaptive vocabluary forests for dynamic indexing and category learning. " In International Conference on Computer Vision, 2007.

自傳

蕭仁惠，民國六十一年生於台北縣，民國八十三年於國立成功大學畢業，曾經服務於茂德科技、聯華電子等知名企業；目前服務於居易科技擔任專案經理一職，並於民國九十四年進入國立交通大學資訊學院碩士在職專班在職進修碩士學位；專長為嵌入式系統、網路應用、專案管理、多媒體系統以及視訊壓縮等等。