國立中山大學 電機工程學系

碩士論文

一個以相似度為基礎的資料縮減方法

A Similarity-based Data Reduction Approach

研究生：歐陽正 撰

指導教授：李錫智 教授

中華民國 九十八 年 七 月

A Similarity-based Data Reduction Approach

Directed by: Professor Shie-Jue Lee

Graduate Student: Jeng OuYang

Department of Electrical Engineering

National Sun Yat-Sen University

Kaohsiung, Taiwan 804, R.O.C.

誌謝

在論文即將完成的此刻，兩年來的點滴歷歷在目，雖然在研究的路上總是艱辛的，有時也會感到失志、無力，所幸有恩師 李錫智老師一路的帶領，總在百忙之中抽空指導我研究的方向，也常為我們鼓勵、打氣，以自身的經歷教導我們，讓我不至於在研究的路上感到迷惘。老師豐富的學識及認真的處事態度更為我這兩年上了寶貴的一課。在這一路上，我最要感謝的人就是老師，謝謝老師您辛勤的教誨。另外，感謝吳志宏老師、賴智錦老師、歐陽振森老師、蔡賢亮老師抽空參加我的口試，給我寶貴的意見讓本篇論文可以更完整。

另一位要感謝的是博士班的吉原學長，認真負責的態度以及對研究的熱忱都是我該學習的榜樣，感謝學長總在我遇到困難時不厭其煩的指導我，提供想法與建議給我，給予我一個明確方向，讓我可以漸漸往前進，在沮喪時給我鼓勵，讓我有動力繼續做下去。我真的很幸運能認識這樣一位學長。感謝忠益、志峰、永申、益賢、佳諺學長們，在我有困難時能適時給我幫助；感謝文彬、世達、祺偉、士賢、仁嘉、杉榮學長們，謝謝你們帶我更快認識這裏的環境；感謝我的同窗好友文豪、連旺、憲奇，在生活上的幫忙與心靈上的鼓勵；感謝書嫻、旻宗、耀瓏、子典學弟妹們，許多事情多虧有你們幫忙才能順利完成。

最重要的我要感謝我的父母親，他們提供了我衣食無虞的生活，一直支持我，是我心靈上的支柱。最後感謝夙婷在這段時間的陪伴與體諒，給了我堅持下去的力量

摘要

由於資訊技術的快速成長，所需處理的資料數量也急遽增加，因此設計一個有效的資料縮減方法是相當重要的一件任務，這也是本篇論文的核心。在本論文裡，我們提出一個以相似度為基礎的自建構式模糊分群演算法來進行資料縮減。此自建構式模糊分群演算法根據資料在統計上的特性，將相似的資料歸為同一個群聚。當所有的資料被輸入至此演算法一遍即可完成分群，並得到每一個群聚的平均值和標準差，而這些平均值的集合就是資料萃取後的結果。最後使用這些少量的新代表點來取代原始的大量資料。此演算法有兩個最大的優點，第一是速度較快且對於記憶體的需求較低。第二是使用者不必事先決定要取出多少代表點。在實驗的部份，我們也以多組真實的資料去驗證此演算法在速度上比其他的方法快而且有更好的縮減率，並透過三種分類器來測試我們所提方法之資料萃取效果。

關鍵字: 大型資料，模糊相似度，資料縮減，資料取樣，資料萃取，資料過濾，資料分群

eThesys 論文

Abstract

Finding an efficient data reduction method for large-scale problems is an imperative task. In this paper, we propose a similarity-based self-constructing fuzzy clustering algorithm to do the sampling of instances for the classification task. Instances that are similar to each other are grouped into the same cluster. When all the instances have been fed in, a number of clusters are formed automatically. Then the statistical mean for each cluster will be regarded as representing all the instances covered in the cluster. This approach has two advantages. One is that it can be faster and uses less storage memory. The other is that the number of new representative instances need not be specified in advance by the user. Experiments on real-world datasets show that our method can run faster and obtain better reduction rate than other methods.

Keywords: Large-scale dataset, fuzzy similarity, data reduction, prototype reduction, instance-filtering, instance-abstraction.

目錄

圖目錄

表目錄

第一章　導論

由於資訊技術的快速發展，在文件探勘、網頁探勘以及影像探勘方面的資料量也急遽的增加。資料量的暴增，相對地也使得分類器對於記憶體的需求量越來越高，如此龐大的資料量可能導致分類器的執行速度越來越慢甚至是無法運作。因此，找出一個有效降低資料量且不影響到分類準確率的方法是一件迫切的工作。資料縮減技術又稱作資料簡化技術(prototype reduction techniques)，以這種技術對資料做前置處理，可以避免分類器使用過多訓練資料而導致記憶體負擔過大的問。資料縮減後，使用者在訓練分類器時只需使用縮減後較少的資料，而這種方式恰巧可以克服目前分類器所面臨訓練資料過多的問題。

資料縮減技術可分為兩大類，一種是資料過濾，它從原始的資料集裡挑選部分的資料來當代表點；另一種為資料萃取，它是產生一組包含原始資料特性的新資料，並使用這組新的資料集來代表原始資料。一般而言這兩種方法只會挑選其中一種使用。Lam et al. [16] 則整合了這兩種資料縮減方法的優點，並提出了資料生成與過濾的方法(prototype generation and filtering, PGF)，他指出了 PGF 在效果上會比只使用資料過濾或資料萃取做資料縮減的效果好。在文獻中，大部分的資料縮減技術的目的在為了改善分類器的效率，其中又以改善資料樣本學習法的分類器(instance-based classifier 例如:*k*-nearest neighbor [7])為最多，因為此類型的分類器在分類速度上較慢[16][34][10][28][9][35][5][29][20][22][11]。這種類型的分類器最大的缺點在於分類速度與訓練資料的數目有關，訓練資料越多速度越慢。為了克服這方面的問題，Kim and Oommen [11]提出了適應性遞迴分割演算法(adaptive recursive partitioning algorithm)，此演算法是把原始的訓練資料藉由 *k*-means 演算法[21]以遞迴的方式切割成較小的子資料集，再以一般常用的資料縮減方法去處理這些由原始訓練資料所構成的子資料集。

隨機抽樣是資料過濾中最簡單的方法，它從訓練資料集裡隨機挑選一小部分資料來代表原始的資料。當資料分布較平均時，通常此資料集的縮減率也較高[31][18][33]。我們將此類型的資料集做隨機抽樣並將抽樣後的結果由支援向量機[6][32]做分類測試，我們發現面對這類型的資料時，隨機抽樣出來的資料有不錯的分類效果。然而當訓練資料的資料分佈極不平均時，隨機抽樣的效果則較不理想。分層抽樣的方法是將隨機抽樣做延伸，他將訓練資料集依照類別各自隨機選取一小部分的資料，它的抽樣結果在做分類測試時比隨機抽樣的效果好[25]。Lee and Huang [17]也指出分層抽樣大大的改善了隨機抽樣的缺點。同時，他們也在研究中實現了平均的隨機子資料集抽樣方法(uniform random subset selection)，這是一種空間填充(space-filling)的抽樣方式，他們指出平均的隨機子資料集抽樣方法是最理想的抽樣方式。而 Wilson and Martinez [35]則提出了另一種資料過濾的方法，稱為遞減式資料縮減最佳化程序(decremental reduction optimization procedure, DROP)以解決資料樣本學習法(instance-based learning)速度上的缺點。在資料萃取的方法方面，包含了以取相似資料的平均當代表點的方式[9]，其中 *k*-means 演算法[21]是將同類別資料根據資料間的距離各自分群，並以每一群中資料的平均當作代表資料。Lozano et al.則實作了學習向量量化(Learning Vector Quantization, LVQ)的方法[19]，他將 LVQ 萃取後的資料以計算差異度方式做分類測試，發現將此種方法萃取出的資料用在訓練分類器上可使分類器擁有不錯的分類效果。Sanchez [29]則提出了三種資料萃取的方式，分別為 RSP-1、RSP-2 和 RSP-3，它們是以空間分割為基礎的資料縮減方法。而 Lam et al. [16]則採取將相似資料合併的方法，這種資料萃取方式是在每一個循環裡將距離最近的兩筆資料合併成一筆新的資料。

本篇論文所提出的是一種資料萃取的方法稱作自建構式模糊分群演算法(self-constructing fuzzy clustering, SCFC)，我們用 SCFC 的方法減少資料的數目以降低分類器的負擔。此演算法為一種遞增式學習法，每次只需讀入一筆訓練資料做運算。我們計算此訓練資料和群聚之間的相似度以判斷此資料與哪一群相似，而每一群資料的歸屬函數我們則應用統計上的平均值和標準差的概念來表示。相似的資料會被分成同一群，如果訓練資料和已存在的群相似則更新歸屬函數，如

果訓練資料和已存在的群皆不相似則成立新的一群。當所有的訓練資料都處理過一遍後，則我們即可得知這組資料集可分成幾群，接下來我們計算出每一群資料的平均當作我們的代表點。由於使用者不用事先決定要取多少資料來當代表點，因此這個方法可以避免為了找出最適合的資料點數目而做的錯誤嘗試。在實驗中我們以多組真實的資料集做測試，測試的結果顯示出我們的方法相較於其他的方法在速度上和資料縮減率上更有優勢。

本論文接下來的章節中，我們在第二章將簡短的介紹幾種不同的資料縮減方法以及我們用來測試縮減效果好壞的幾個分類器模型。第三章將介紹我們的研究方法：以相似度為基礎的資料縮減演算法，並以一個例子詳細說明這個方法是如何運作。在第四章我們以多組真實資料集做測試，比較我們的方法和其他的資料縮減方法的優缺點以及對實驗結果的分析。第五章則是對我們所提出的方法的優缺點下結論以及未來可繼續研究的方向。

第二章　文獻探討

為了解決大資料量對分類器所造成的負擔，目前在資料縮減方面已有許多相關的研究，也有許多方法被提出來，而我們在本章節裡將簡短的介紹幾種資料過濾和資料萃取的方法，並在之後的章節中會和我們所提的方法做個比較。為了能較客觀的比較幾種資料縮減間效果的好壞，我們依據分類器的分類效果來當我們評估的依據，因此在本章節中也會簡略的介紹幾種常用的分類器，並在往後的章節中依照這幾種分類器的分類效果來比較資料縮減的效果。

2.1 資料縮減方法

在做資料分類時，我們先給定 ℓ 筆已標記類別的訓練資料集 $S_{tr}=\{(\mathbf{x}_1,y_1),(\mathbf{x}_2,y_2),...,(\mathbf{x}_\ell,y_\ell)\}$，其中訓練資料 $\mathbf{x}_i=[x_{i,1},x_{i,2},...,x_{i,n}]$，$i=1,2,...,\ell$，$n$ 為資料的特徵數，資料類別以 y_i 表示，$y_i\in\{1,2,...,k\}$。在一般的分類問題中，我們會先拿訓練資料 S_{tr} 對分類器做訓練，使的分類器可以對未知的資料做較準確的預測。而資料縮減的工作則是從原始的訓練資料中產生一組數量較小的新的訓練資料集 $S_{tr}^{'}=\{(\mathbf{x}_1^{'},y_1),(\mathbf{x}_2^{'},y_2),...,(\mathbf{x}_J^{'},y_J)\}$，$J<<\ell$，並以 $S_{tr}^{'}$ 取代 S_{tr} 對分類器進行訓練，且期望 $S_{tr}^{'}$ 對未知的資料有和 S_{tr} 一樣好的預測效果。如果 J 遠小於 ℓ，則以 $S_{tr}^{'}$ 當訓練資料可大大減少運算時所需耗費的資源。目前的資料縮減方式大致可分為兩種類型: (1)資料過濾法 (2)資料萃取法 [16]。在接下來的內容中我們將簡短的介紹幾種和我們的方法比較的資料萃取技術。

2.1.1 資料過濾法

資料過濾法的概念是從原本的訓練資料 S_{tr} 裡挑選部分資料組成

新的訓練資料集 $S_{tr}^{'}=\{(\mathbf{x}_1^{'},y_1),(\mathbf{x}_2^{'},y_2),...,(\mathbf{x}_J^{'},y_J)\}$。分層抽樣(stratified sampling)是一個統計上簡單且常見的資料過濾法，它從每一個類別 C_j 裡平均且隨機的抽取部分資料 h_j，$j=1,2,...,k$，$J=h_1+h_2+...+h_k$，此種方法的優點是不需做額外的運算，缺點則是此方法未考慮到資料的分布特性。

另一種資料過濾的方法，DROP3 [35]是我們比較的方法之一，DROP3 一開始必須先對所有的資料做距離的運算，最後的輸出結果為 $S_{tr}^{'}$ 且 $S_{tr}^{'}\subseteq S_{tr}$。DROP3 演算法可分為三個步驟，首先第一個步驟是先對所有資料做 k-NN，並拿掉預測錯誤的資料，在這個步驟中會將雜訊和邊緣部份的資料移除。接下來是計算每筆資料與不同類別資料的最短距離，並以此距離做資料排序的依據，此步驟的目的是希望在下一個步驟處理資料時先處理距離類別邊界較遠的資料。最後一個步驟是測試每一筆資料對分類結果的影響，測試的順序則依照前一個步驟排序後的結果。假若移除某筆資料 **x** 會導致整體準確率下降則將此筆資料保留。相反的，則移除此筆資料。DROP3 的優點是在做分群時有考量到資料的分布特性，且此種方法不用事先決定要取多少代表資料，缺點則是此方法是以 k-NN 為基礎，因此有運算量過大及較花時間的問題。很明顯的，DROP3 是一種遞減式的資料縮減法，除此之外此演算法是針對連續型態資料所設計的，如果要同時處理包含離散和連續型態特徵的資料，則必須將這個方法作些微的修正。修正的方式是將原本的訓練資料 S_{tr} 依照離散特徵的不同分成 T 組互斥的子集合 $S_{tr}=S_1\cup S_2\cup...\cup S_T$，每個子集合再各自做 DROP3。其中每一子資料集含有相同的離散型態特徵。在執行 DROP3 演算法時，會有兩種情況，第一種就是當子資料集只存在一筆資料時，我們直接將此筆資料當作我們的代表資料；另一種情形是當子資料集的資料不只一筆，則我們則將此子資料集裡的資料做 DROP3。最後所產生的代表資料 $S_{tr}^{'}=S_1^{'}\cup S_2^{'}\cup...\cup S_T^{'}$ 且 $S_{tr}^{'}\subseteq S_{tr}$。

2.1.2 資料萃取法

資料萃取法是一種經由整理出相似資料的特性而產生一組新的代表資料的方法，且新產生的資料集未必包含於原始的資料集中。一般而言，資料萃取法可視為一種資料分割的技術，它將資料分割成多群，並以同一群資料的平均當作新的代表資料，產生的新的訓練資料集 $S_{tr}' \propto S_{tr}$。在本節裡我們將介紹兩種資料分割的方法，*k*-means [21] 和 RSP3 [29]，並做為和我們的方法比較的目標。

k-means 是一種簡單的遞迴式分群法，它將訓練資料分成 *k* 群，*k* 必須由使用者先定義。傳統的 *k*-means 演算法僅能使用在連續型態的資料上，如果資料型態包含離散的特徵，則必須將離散的特徵轉換成連續型態才能做距離的運算。然而這種作法對於離散的特徵而言是不合理的，因而我們在此研究裡提供了一個計算資料點 $\mathbf{x}_i$ 與群 c_j 的差異度的方式:

$$dis(\mathbf{x}_i, \mathrm{m}_j) = dis^{(c)}\left(\mathbf{x}_i^{(c)}, \mathrm{m}_j^{(c)}\right) + dis^{(d)}\left(\mathbf{x}_i^{(d)}, \mathrm{m}_j^{(d)}\right) \tag{2.1}$$

$dis^{(c)}(\cdot,\cdot)$ 表示連續型態特徵的差異度，$dis^{(d)}(\cdot,\cdot)$ 表示離散型態特徵的差異度，資料點 $\mathbf{x}_i = \{\mathbf{x}_i^{(c)}, \mathbf{x}_i^{(d)}\}$，$\mathbf{x}_i^{(c)}$ 和 $\mathbf{x}_i^{(d)}$ 分別表示資料 $\mathbf{x}_i$ 的連續型態特徵和離散型態特徵，同樣的，$\mathrm{m}_j = \{\mathrm{m}_j^{(c)}, \mathrm{m}_j^{(d)}\}$，$\mathrm{m}_j^{(c)}$ 和 $\mathrm{m}_j^{(d)}$ 分別表示群 c_j 裡資料的連續型態的特徵平均和離散型態特徵的平均。在連續型態特徵的計算上，我們以歐幾里德距離的平方計算資料點 $\mathbf{x}_i$ 與群 c_j 的連續型態特徵差異度:

$$dis^{(c)}(\mathbf{x}_i^{(c)}, \mathrm{m}_j^{(c)}) = \sum_{k=1}^{n_1}\left(x_{i,k}^{(c)} - m_{j,k}^{(c)}\right)^2 \tag{2.2}$$

n_1 為連續型態特徵的數目。而資料點 $\mathbf{x}_i$ 與群 c_j 的離散型態特徵差異度計算方式如下:

$$dis^{(d)}(\mathbf{x}_i^{(d)}, \mathrm{m}_j^{(d)}) = \sum_{k=1}^{n_2}\left(x_{i,k}^{(d)}, m_{j,k}^{(d)}\right) \tag{2.3}$$

n_2 為離散型態特徵的數目，$n = n_1 + n_2$。

$$\theta(x_{i,k}^{(d)}, m_{j,k}^{(d)}) = \begin{cases} 0, & \text{if } x_{i,k}^{(d)} = m_{j,k}^{(d)} \\ 1, & \text{otherwise} \end{cases} \tag{2.4}$$

我們的做法是先將資料依照類別分開，每個類別各自做 k-means，這種資料萃取法的優點是有考量到資料的分布特性，缺點是 k-means 會受到初始值的影響，以及 k-means 需要做多個循環才會達到收斂，因此較花時間。

RSP3 演算法簡單而言是一種將子集合的資料不斷做分割，直到同一子集合裡的資料類別都相同的方法。換句話說，分完群後同一群資料的類別都相同[29]。且 RSP3 在完成資料分割後，每一個子集合會用一筆新資料來表示。假定有一個子集合 S 裡的資料不屬於同一類別，依照 RSP3 演算法，它先找出 S 裡距離最遠的兩筆資料 $\mathbf{x}_1$ 和 $\mathbf{x}_2$，再將 S 裡的資料依照與這兩點的距離遠近分成兩個子集合，依此類推直到同一子集合裡的資料類別都相同。此演算法的優點在於不必先定義資料分割的數量，缺點是必須計算兩兩資料間的距離，耗費許多運算資源。以傳統的 RSP3 演算法來說，它只能處理連續型態的資料，為了讓 RSP3 同時也能應用在混合型態的資料上，我們對這種法做了一點修正。我們先將原本的訓練資料 S_{tr} 依照離散型態特徵的不同分割成 T 組互斥的子集合，並各自做 RSP3。此時會有三種狀況，第一種情形是當子集合裡只存在一筆資料，此時我們直接將此子集合當作我們的代表資料；第二種情形是子集合裡的資料都屬於同一類別，我們則將此集合的資料取平均當這個集合的代表資料；最後一種情況是子集合裡的存在多筆資料且不屬於同一類別時，我們則繼續執行 RSP3 演算法進行資料分割。RSP3 最終的結果為 $S_{tr}^{'}$，$S_{tr}^{'} = S_1^{'} \cup S_2^{'} \cup ... \cup S_T^{'}$ 且 $S_{tr}^{'} \subset S_{tr}$。

2.2 分類器模型

假設我們有一組資料量較大的訓練資料集 S_{tr}，在經過資料縮減後得到一組數量較少的新的訓練資料集 S_{tr}'。我們用 S_{tr}' 來訓練分類器並以分類器的分類結果做為比較我們所提出的演算法和其他的資料縮減方法效果好壞的依據。在此我們以支援向量機(SVM [6][32][8])，C4.5 決策樹[26][27]和 *k*-nearest neighbor [7]當作我們測試用的分類器，會選擇這幾種分類器不僅是因為在先前的研究中這幾種分類器在解決分類問題上的效果較好，同時這幾種分類方法也被選為資料探勘領域中前 10 名的演算法[15]。

2.2.1 支援向量機

SVM [6][32][8]是一種核心學習的方法，它的目的是希望能在特徵空間裡找到一個能使邊界(margin)最大的超平面(hyperplane)，使得輸入資料可達到線性分割。為了達到最佳的分類效果，在決定超平面時，我們必須容忍部份資料被預測錯誤，但這些分類錯誤的資料我們必須給予一個懲罰值做為修正。因此，我們用鬆弛變數(slack variable) ξ_i 來記錄分類錯誤的資料，而這個分類器模型可用以下的式子表示:

$$\min_{\mathbf{w},b} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{J}\xi_i \qquad (2.5)$$

$$y_i\left(\left\langle \mathbf{w}, \phi\left(\mathbf{x}_i'\right)\right\rangle + b\right) \geq 1-\xi_i \text{ ， } \xi_i \geq 0 \text{ ，} i = 1, 2, \ldots, J.$$

此處 J 表示資料縮減後新的訓練資料集的資料數量，C 則是在最大邊界和資料分類錯誤量之間的一個平衡參數，當 C 設大則表示當資料落在邊界外時會有較大的懲罰值，即重視分類準確度，因而邊界較小。若 C 設小，則資料分類錯誤容忍度高，相對的邊界會較大。而函式 $\phi: X \to F$ 表示將資料從輸入空間映射到特徵空間使資料更容易做分割，$\mathbf{w}$ 為特徵空間的權重向量，b 為超平面的偏移量，而 y_i 為目標類別。超平面的表示方式為 $\left\langle \mathbf{w}, \phi\left(\mathbf{x}_i'\right)\right\rangle + b = 0$，它將資料分成 $y_i = +1$ 和 $y_i = -1$ 兩

個類別。

前面所介紹的 SVM 只能將資料分成 y_i =+1和 y_i =-1兩類，但為了使 SVM 能用在解決多類別的問題上，必須對 SVM 做些改善。改善的方式是將原本多類別的問題分成多個兩類別的子問題個別處理，然後再用鑑別函數(discriminant function)達到多類別的判斷。鑑別函數可分為: 一對多(One-Against-All, OVA [3])，一對一(One-Against-One, OVO [14])和不循環有向圖(Directed Acyclic Graph, DAG [24])。我們所採用的是一對一的方式來建立以 SVM 為基礎的多類別分類器。假定資料有 k 個類別，就需要有 $k(k\text{-}1)/2$ 個 SVM。每一個 SVM 分別處理任兩個類別 C_u 和 C_v，$1 \le u$，$v \le k$ 且 $u \ne v$，其中一個類別標記成 y_i =+1另一個則為 y_i =-1。訓練資料在每一個 SVM 裡必定被預測為兩類別中的一類。在做測試資料預測時則是由所有的 SVM 的預測結果來決定，預測結果數量最多的類別即為此筆資料的預測結果。

2.2.2 C4.5 決策樹

決策樹是一種簡單且廣泛應用的分類技術，決策樹的分支節點表示資料的屬性，葉節點表示決策結果。通常我們藉由資訊獲利(information gain)來幫助我們選擇分支節點的屬性。目前較常見的決策樹分類器有 ID3，C4.5 [26][27]和 CART [4]。我們使用由 Quinlan [26][27]所提出的 C4.5 決策樹做為我們測試用的分類器。首先我們將訓練資料以 2:1 的比例分成兩部份，前面部分的資料用來建立(growing)這棵樹的，剩下部分的資料則是用來修剪(pruning)這棵樹。C4.5 以分割與克服(divide-and-conquer)的方式建立決策樹，分支節點則是以計算每一個屬性的獲得量比例(gain ratio)來決定，採用這種計算方式可以避免使用屬性值種類過多的屬性當分支節點，其計算方式如下:

$$\text{Gain ratio} = \frac{E(parent) - \sum_{j=1}^{t} \frac{N(v_j)}{N} E(v_j)}{-\sum_{j=1}^{t} p(v_j)\log_2 p(v_j)} \tag{2.6}$$

t 為此屬性的屬性值數目，N 為父節點的資料數目，$N(v_j)$ 為通過子節點屬性值 v_j 的資料數目，$E(\cdot)$ 則表示此點的亂度(entrop)，表示如下：

$$E(node) = -\sum_{i=1}^{k-1} p(i|node)\log_2 p(i|node) \tag{2.7}$$

k 是類別的數目，$p(i|node)$ 表示在已知 $node$ 的資料量下類別為 i 所佔的比例。當決策樹建立完成後，我們將剩餘部份的資料拿來做樹的修剪，以防止過適的現象(overfitting)。

2.2.3 *k*-Nearest Neighbor

k-NN 演算法[7]是資料樣本學習分類法中最常使用到的分類方式，它將所有的訓練資料都記錄下來以供測試一一比對，因而他也稱作惰式學習法(lazy-learning)。在做測試時，我們以計算歐幾里德距離(Euclidean distance)的平方來計算測試資料 **x** 與所有訓練資料的距離，並選出最接近 **x** 的前 k 筆資料 $S_{tmp} = \{(\mathbf{x}_1', y_1), (\mathbf{x}_2', y_2), ..., (\mathbf{x}_k', y_k)\}$，最後依照這些資料來預測 **x** 的類別。假設找出的鄰近資料的類別不屬於同一類，則以預測類別數量最多的當此測試資料的預測結果。其表示式如下:

$$\hat{y} = \arg\max_{v} \sum_{(\mathbf{x}_j', y_j) \in S_{tmp}} I(v = y_j) \tag{2.8}$$

v 為所有資料類別，y_j 為 k 筆鄰近的資料的類別，$I(\cdot)$ 為一指示函數:

$$I(v = y_j) = \begin{cases} 1, & \text{if } v = y_j \\ 0, & \text{otherwise} \end{cases} \tag{2.9}$$

雖然 *k*-NN 不需要花費時間在訓練上，但在分類測試上卻比先行學習(eager-learning)的分類方式(例如: SVM 和 C4.5)需要耗費更多的時間。

第三章　研究方法

大部分現存的資料縮減技術幾乎都存在運算量過大的問題，這些資料縮減方法大都需要計算訓練資料集裡兩兩資料之間的距離。為了解決資料縮減技術面臨大量運算的問題以及有效降低資料量減輕分類器的運算負擔，我們提出了以相似度為基礎的自建構式模糊分群演算法，這是一種遞增式的資料縮減方法。這個方法有兩個優勢，第一點是在速度上以及記憶體的花費上負擔較輕，第二點是使用者不用決定要取多少資料量來代表原始的訓練資料。除此之外，我們在計算訓練資料和群之間的相似度時也以標準差的方式去考量同一群資料裡分布的狀態。

3.1 自建構式模糊分群演算法

自建構式模糊分群演算法(SCFC)的基本概念是一次只需讀入一筆資料做處理，資料處理的方式是將讀入的資料和現在存在的群做相似度的運算，所得到的相似度則用來判斷此筆資料與哪一群較相似或是是否新增一群。如果此筆資料被分配到某一群，則更新那一群的歸屬函數；如果是新增一群，則將這新增的一群的歸屬函數給一個初始化的結果。我們所提出的 SCFC 演算法是一個五階層的網路架構，其架構參照圖 3.1 所示。此方法最大的優點在於分群時只需將所有訓練資料都檢查過一遍，則可自動決定這組資料集需要被分成多少群。

假設我們有 ℓ 筆已經標記類別的訓練資料集 S_{tr}，$S_{tr}=\{(\mathbf{x}_1,y_1),(\mathbf{x}_2,y_2),...,(\mathbf{x}_\ell,y_\ell)\}$，則第 p 筆訓練資料 $\mathbf{x}_p$ 表示為 $\mathbf{x}_p=[x_{p,1},x_{p,2},...,x_{p,n_1},x_{p,n_1+1},...,x_{p,n}]$，$p=1,2,...,\ell$，$n$ 為全部特徵的總數，n_1 是連續型態特徵的數目，ℓ 是訓練資料集的數目，$y_p\in\{1,2,...,k\}$，k 為類別數。特別注意的是 $[x_{p,1},x_{p,2},...,x_{p,n_1}]$ 表示 $\mathbf{x}_p$ 連續型態的特徵，$[x_{p,n_1+1},x_{p,n_1+2},...,x_{p,n}]$ 表示的是 $\mathbf{x}_p$ 離散型態的特徵。每

一群我們則用 $c_1, c_2, \ldots, c_J$ 來表示，而群 c_j 的中心(平均值) m_j 表示為 $m_j = [m_{j,1}, m_{j,2}, \ldots, m_{j,n}, m_{j,n+1}, \ldots, m_{j,n}]$，標準差 $\sigma_j = [\sigma_{j,1}, \sigma_{j,2} \ldots, \sigma_{j,n_1}]$，$s_j$ 表示 c_j 群所包含的資料數量。在初始化時，我們定 $J=0$ 表示一開始不存在任何群，接著讀入第一筆訓練資料，並將它定為第一群 c_1 且 $m_1 = \mathbf{x}_1$，$y_{c_1} = y_1$，$\sigma_1 = \sigma_0$，$s_1 = 1$，在這裡 σ_0 是由使用者定義的一個常數向量 $\sigma_0 = [\sigma_{0,1}, \sigma_{0,2} \ldots, \sigma_{0,n_1}]$。此時第一群產生，$J=1$。

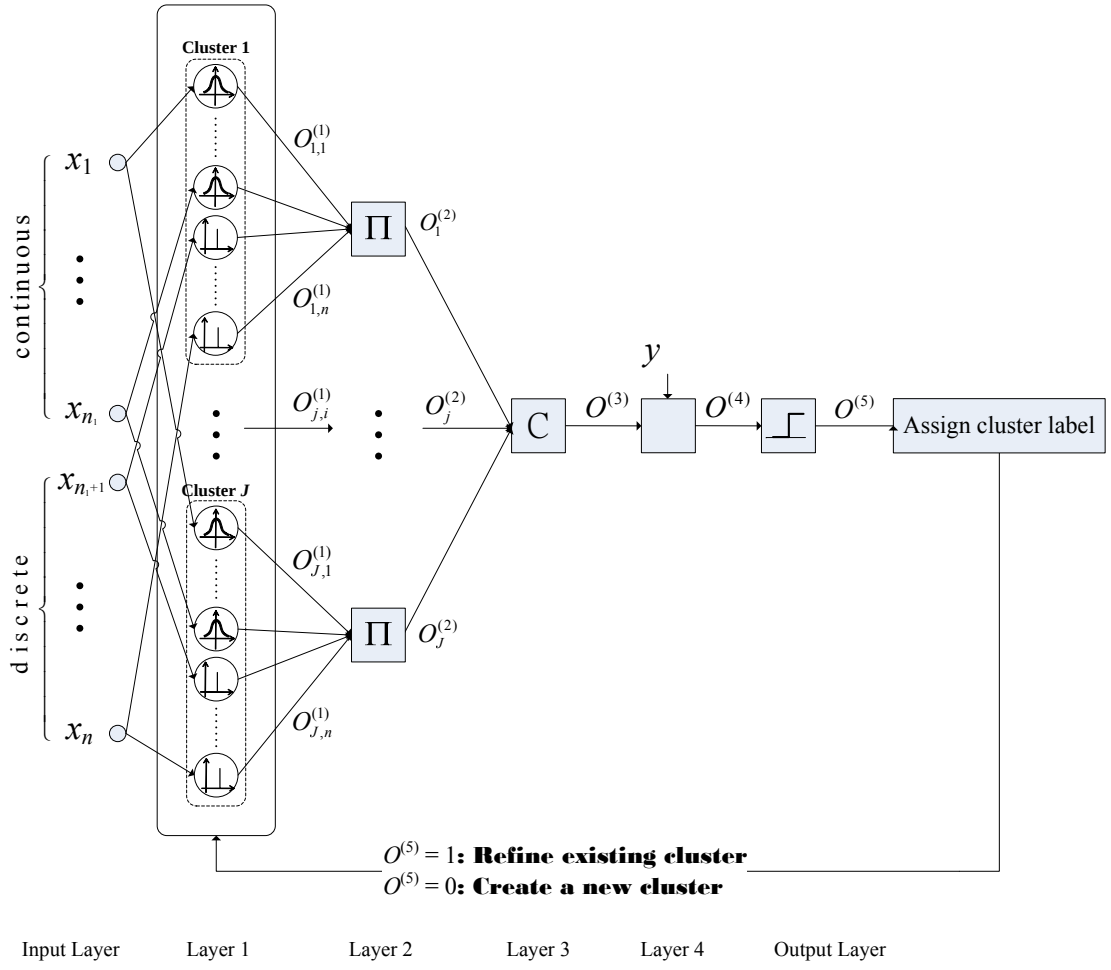


圖 3.1: 自建構式模糊分群網路

第一層：此層執行模糊化運算，每一個節點表示一群，節點的數量是自動決定的。要注意的是，如果訓練資料集 S_{tr} 的資料型態包含離散型態和連續型態兩種特徵，則歸屬函數必須分開來運算。在計算連續型態的特徵上，我們使用高斯函數來運算會比用其他函數的效果較好

[12]。在離散型態特徵的運算上，我們以模糊單值(fuzzy singleton)的方式來表示。此層連續型態特徵的歸屬值如下：

$$O_{j,i}^{(1)} = \exp\left[-\left(\frac{x_{p,i} - m_{j,i}}{\sigma_{j,i}} \right)^2 \right] \tag{3.1}$$

$i = 1, 2, \ldots, n_1$，$j = 1, 2, \ldots, J$，$x_{p,i}$ 表示訓練資料 $\mathbf{x}_p$ 連續型態的特徵，$m_{j,i}$ 和 $\sigma_{j,i}$ 分別表示 c_j 群的第 i 個維度的平均值和標準差。而此層離散型態特徵的歸屬值如下：

$$O_{j,i}^{(1)} = \begin{cases} 1, & \text{if } x_{p,i} = m_{j,i} \\ 0, & \text{otherwise} \end{cases} \tag{3.2}$$

$i = n_{1+1}, n_{1+2}, \ldots, n$，$j = 1, 2, \ldots, J$，$x_{p,i}$ 表示訓練資料 $\mathbf{x}_p$ 離散型態的特徵，$m_{j,i}$ 表示 c_j 群的離散型態的特徵。

第二層：本層的節點為固定的，標記為 Π，輸出與輸入的關係表示如下：

$$O_j^{(2)} = \prod_{i=1}^{n} O_{j,i}^{(1)} \tag{3.3}$$

$j = 1, 2, \ldots, J$，每一個節點的輸出表示訓練資料 $\mathbf{x}_p$ 和群 c_j 的相似度。我們明顯的發現，同一群的資料必擁有相同的離散型態特徵。

第三層：此層為一單一固定的節點，標示為 C，此為一競爭式運算，運算後的結果僅輸出勝出的。本層的輸出結果即為輸入的最大值：

$$O^{(3)} = \max_{1 \le j \le J} O_j^{(2)} \tag{3.4}$$

假定勝出的群為 c_a 則:

$$a = \arg \max_{1 \le j \le J} O_j^{(2)} \tag{3.5}$$

第四層：此層為一單一固定的節點，它的輸出結果表示訓練資料 $\mathbf{x}_p$ 與最接近的群 c_a 在輸入特徵與輸出類別的相似度關係：

$$O^{(4)} = O^{(3)} \times \delta(y_p, y_{\mathrm{c}_a}) \tag{3.6}$$

$\delta(\cdot,\cdot)$ 的定義如下:

$$\delta(y_p . y_{c_a}) = \begin{cases} 1, & \text{if } y_p = y_{c_a} \\ 0, & \text{otherwise} \end{cases} \tag{3.7}$$

在此層我們不僅考慮輸入特徵相似度，同時也計算了輸出類別的相似度。很明顯的，我們發現同一群的資料必定屬於同一類別。

第五層：此層為一單一固定的節點，為一硬限制函數(hard limit function)，其表示如下:

$$O^{(5)} = \text{hardlim}(O^{(4)}) = \begin{cases} 1, & \text{if } O^{(4)} \geq \rho \\ 0, & \text{otherwise} \end{cases} \tag{3.8}$$

ρ 為使用者定義的門檻值，$0 \leq \rho \leq 1$

如果 $O^{(5)} = 1$ 表示 $\mathbf{x}_p$ 通過了和 c_a 群的相似度門檻值，此時我們認定 $\mathbf{x}_p$ 屬於 c_a 這一群，當 c_a 加入了 $\mathbf{x}_p$ 這筆資料，我們必須修正 c_a 的 m_a 和 σ_a，修正後的 c_a 定義如下:

$$m_{a,i} = \frac{s_a \times m_{a,i} + x_{p,i}}{s_a + 1} \tag{3.9}$$

$i = 1, 2, \ldots, n_1$，

$$\sigma_{a,i} = \sqrt{\frac{(s_a^2 - 1)(\sigma_{a,i} - \sigma_{0,i})^2 + s_a (x_{p,i} - m_{a,i})^2}{s_a (s_a + 1)}} + \sigma_{0,i} \tag{3.10}$$

$\sigma_{0,i}$ 為使用者定義的初始標準差，$i = 1, 2, \ldots, n_1$，且

$$s_a = s_a + 1 \tag{3.11}$$

在 $O^{(5)} = 1$ 的情況下 m_a 的離散型態特徵和 J 是不會改變的。

如果 $O^{(5)} = 0$ 表示 $\mathbf{x}_p$ 和所有的群皆未通過相似度的門檻值。在這種情形下我們認定 $\mathbf{x}_p$ 和現有的群皆不相似，因此我們新增新的一群 c_{J+1} 且

$$m_{J+1} = \mathbf{x}_p \, , \sigma_{J+1} = \sigma_0 \tag{3.12}$$

此時 $\mathbf{x}_p$ 被判定屬於 c_{J+1} 群，並對 c_{J+1} 的資料量大小 s_{J+1} 做初始化

$$s_{J+1} = 1 \tag{3.13}$$

而總群數也增加了 1

$$J = J + 1 \tag{3.14}$$

前面的過程顯示出了 SCFC 處理資料的方式，一次處理一筆資料直到所有的資料都處理完畢，最終我們會得到資料總群數 J 以及，以及新的一組訓練資料集 $S_{tr}^{'} = \{(\mathbf{x}_1^{'}, y_1), (\mathbf{x}_2^{'}, y_2), ..., (\mathbf{x}_J^{'}, y_J)\}$，$\mathbf{x}_j^{'}$ 為 c_j 中心點（平均值），$j = 1, 2, \ldots, J$，整個分群的演算法整理如演算法 1。當資料完成分群，屬於同一群資料間的相似度會比其他群的資料高。此外，不論是要更新已存在的群或是新增一群，都不需要拿同一群裡所有的資料來計算。

演算法 1: 以相似度為基礎的自建構式模糊分群演算法

Input: (1) $\mathbf{x}_p = [x_{p,1}, x_{p,2}, \ldots, x_{p,n}]^{\mathrm{T}} \in S_{tr}$, where p = 1, 2, ..., ℓ. (2) Threshold: ρ.
(3) Initial standard deviation: σ_0.

Output: Clusters $\mathrm{c}_1, \mathrm{c}_2, \ldots, \mathrm{c}_J$

Initialization: $J = 0$.

1: c_1 is created by Eqs.(3.12)-(3.14).
2: **for** i = 2 to ℓ **do**
3: Calculate the membership degree $O_{j,i}^{(1)}$ between $\mathbf{x}_p$ and c_j by Eqs.(3.1), (3.2), where $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, J$.
4: Calculate the similarity $O_j^{(2)}$ between $\mathbf{x}_p$ and c_j by Eq.(3.3), where $j = 1, 2, \ldots, J$.
5: Find the winner cluster by Eqs.(3.4), (3.5). Let c_a be the winner cluster.
6: Calculate the input-output similarity by Eqs.(3.6), (3.7).
7: Calculate $O^{(5)}$ by Eq.(3.8).
8: **if** $O^{(5)} = 0$ **then**
9: A new cluster c_{J+1} is created by Eqs.(3.12)-(3.14). Assign $\mathbf{x}_p$ to the new cluster.
10: **else**
11: Combine $\mathbf{x}_p$ into c_a by Eqs.(3.9)-(3.11). Assign $\mathbf{x}_p$ to the winner cluster.
12: **end if**
13: **end for**

3.2 範例

本節我們提供一個範例去描述我們的方法實際上運作的情況。假定我們有一組簡單的訓練資料集 S_{tr} 此資料集包含了 10 筆資料 $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_{10}, y_{10})\}$ 並分為兩個類別 C_1 和 C_2，每一筆資料有 5 個連續型態的特徵和 3 個離散型態的特徵，此組資料表示如下:

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \\ \mathbf{x}_4 \\ \mathbf{x}_5 \\ \mathbf{x}_6 \\ \mathbf{x}_7 \\ \mathbf{x}_8 \\ \mathbf{x}_9 \\ \mathbf{x}_{10} \end{bmatrix} = \begin{bmatrix} 0.82 & 0.69 & 0.73 & 0.92 & 0.58 & A & D & H \\ 0.78 & 0.82 & 0.75 & 0.88 & 0.63 & A & D & H \\ 0.83 & 0.70 & 0.71 & 0.93 & 0.65 & B & E & G \\ 0.48 & 0.39 & 0.31 & 0.51 & 0.25 & B & E & G \\ 0.81 & 0.68 & 0.76 & 0.87 & 0.62 & C & E & G \\ 0.25 & 0.31 & 0.41 & 0.19 & 0.28 & C & E & G \\ 0.50 & 0.45 & 0.22 & 0.35 & 0.62 & A & F & K \\ 0.52 & 0.47 & 0.25 & 0.33 & 0.65 & A & F & K \\ 0.53 & 0.49 & 0.18 & 0.37 & 0.63 & A & F & K \\ 0.27 & 0.30 & 0.40 & 0.20 & 0.30 & C & E & G \end{bmatrix}, \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \\ y_9 \\ y_{10} \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 2 \\ 2 \\ 2 \\ 2 \\ 2 \end{bmatrix}$$

我們在執行自建構式模糊分群演算法時，必須先定義一些參數，以這個例子而言，我們訂 $\rho = 0.8^5 = 0.328$，$\sigma_0 = [0.1, 0.1, 0.1, 0.1, 0.1]$。首先，我們處理第一筆資料 $\mathbf{x}_1$ 並得到第一群 c_1 且

$$\mathrm{m}_1 = [\,0.82\; 0.69\; 0.73\; 0.92\; 0.58\; \mathrm{A}\; \mathrm{D}\; \mathrm{H}\,]$$

$$\sigma_1 = [\,0.1\; 0.1\; 0.1\; 0.1\; 0.1\,]$$

$$y_{\mathrm{c}_1} = 1$$

$$s_1 = 1$$

$$J = 1$$

接著, 我們讀入第二筆資料 $\mathbf{x}_2$ 其處理流程如下:

第一層： 每一維度和第一群計算得到的歸屬值

$$O_{1,i}^{(1)} = [\,0.852\; 0.914\; 0.961\; 0.779\; 0.779\; 1\; 1\; 1\,]$$

第二層： 計算輸入相似度

$$O_1^{(2)} = 0.852 \times 0.914 \times 0.961 \times 0.779 \times 0.779 \times 1 \times 1 \times 1 = 0.454$$

第三層：得到勝出的一群為 c_1

$$O^{(3)} = 0.454$$

第四層：計算輸入與輸出的相似度

$$O^{(4)} = 0.454 \times 1 = 0.454$$

第五層：因為 $O^{(4)} > \rho$，所以得到 $O^{(5)} = 1$，因此 $\mathbf{x}_2$ 被歸類屬於 c_1，此時必須更新 c_1 的 m_1 和 σ_1：

$$m_{1,1} = \frac{s_1 \times m_{1,1} + x_{2,1}}{s_1 + 1} = \frac{1 \times 0.82 + 0.78}{1+1} = 0.8000$$

$$m_{1,2} = \frac{s_1 \times m_{1,2} + x_{2,2}}{s_1 + 1} = \frac{1 \times 0.69 + 0.82}{1+1} = 0.705$$

$$m_{1,3} = \frac{s_1 \times m_{1,3} + x_{2,3}}{s_1 + 1} = \frac{1 \times 0.73 + 0.75}{1+1} = 0.740$$

$$m_{1,4} = \frac{s_1 \times m_{1,4} + x_{2,4}}{s_1 + 1} = \frac{1 \times 0.92 + 0.88}{1+1} = 0.900$$

$$m_{1,5} = \frac{s_1 \times m_{1,5} + x_{2,5}}{s_1 + 1} = \frac{1 \times 0.58 + 0.63}{1+1} = 0.605$$

$m_{1,6} = \mathrm{A}$，$m_{1,7} = \mathrm{D}$，$m_{1,8} = \mathrm{H}$

$$\sigma_{1,1} = \sqrt{\frac{(s_1^2 - 1)(\sigma_{1,1} - \sigma_{0,1})^2 + s_1(x_{2,1} - m_{1,1})^2}{s_1(s_1+1)}} + \sigma_{0,1}$$

$$= \sqrt{\frac{(1^2 - 1)(0.1 - 0.1)^2 + 1(0.78 - 0.82)^2}{1(1+1)}} + 0.1 = 0.128$$

$$\sigma_{1,2} = \sqrt{\frac{(s_1^2 - 1)(\sigma_{1,2} - \sigma_{0,2})^2 + s_1(x_{2,2} - m_{1,2})^2}{s_1(s_1+1)}} + \sigma_{0,2}$$

$$= \sqrt{\frac{(1^2 - 1)(0.1 - 0.1)^2 + 1(0.82 - 0.69)^2}{1(1+1)}} + 0.1 = 0.121$$

$$\sigma_{1,3} = \sqrt{\frac{(s_1^2 - 1)(\sigma_{1,3} - \sigma_{0,3})^2 + s_1(x_{2,3} - m_{1,3})^2}{s_1(s_1+1)}} + \sigma_{0,3}$$

$$= \sqrt{\frac{(1^2 - 1)(0.1 - 0.1)^2 + 1(0.75 - 0.73)^2}{1(1+1)}} + 0.1 = 0.114$$

$$\sigma_{1,4}=\sqrt{\frac{(s_1^2\text{-}1)(\sigma_{1,4}\text{-}\sigma_{0,4})^2+s_1(x_{2,4}\text{-}m_{1,4})^2}{s_1(s_1+1)}}+\sigma_{0,4}$$

$$=\sqrt{\frac{(1^2\text{-}1)(0.1\text{-}0.1)^2+1(0.88\text{-}0.92)^2}{1(1+1)}}+0.1=0.128$$

$$\sigma_{1,5}=\sqrt{\frac{(s_1^2\text{-}1)(\sigma_{1,5}\text{-}\sigma_{0,5})^2+s_1(x_{2,5}\text{-}m_{1,5})^2}{s_1(s_1+1)}}+\sigma_{0,5}$$

$$=\sqrt{\frac{(1^2\text{-}1)(0.1\text{-}0.1)^2+1(0.63\text{-}0.58)^2}{1(1+1)}}+0.1=0.135$$

$$\mathrm{m}_1=[\,0.800\;0.705\;0.740\;0.900\;0.605\;\mathrm{A}\;\mathrm{D}\;\mathrm{H}\,]$$

$$\sigma_1=[\,0.128\;0.121\;0.114\;0.128\;0.135\,]$$

$$y_{\mathrm{c}_1}=1$$

$$s_1=1+1=2$$

$$J=1$$

當我們讀入第三筆資料 $\mathbf{x}_3$ 時其處理流程如下:

第一層： 每一維度和第一群計算得到的歸屬值

$$O_{1,i}^{(1)}=[\,0.947\;0.998\;0.933\;0.947\;0.895\;0\;0\;0\,]$$

第二層： 計算輸入相似度

$$O_1^{(2)}=0.947\times0.998\times0.933\times0.947\times0.895\times0\times0\times0=0$$

第三層： 得到勝出的一群為 c_1

$$O^{(3)}=0$$

第四層： 計算輸入與輸出的相似度

$$O^{(4)}=0\times1=0$$

第五層： 因為 $O^{(4)}<\rho$，所以得到 $O^{(5)}=0$，因此 $\mathbf{x}_3$ 與現有的群皆不相似因此必需新增一群 c_2 且

$$\mathrm{m}_2=[\,0.83\;0.70\;0.71\;0.93\;0.65\;B\;E\;G\,]$$

$$\sigma_2=[\,0.1\;0.1\;0.1\;0.1\;0.1\,]$$

$$y_{\mathrm{c}_2}=1$$

$$s_2 = 1$$

$$J = 1 + 1 = 2$$

當所有的訓練資料集 S_{tr} 裡所有的資料都處理過後，我們得到 6 群，$c_1, c_2, c_3, c_4, c_5, c_6$ 表示如表 3.1

表 3.1: 訓練資料集 S_{tr} 被分成六群

	size s	mean	standard deviation
c_1	2	[0.800 0.705 0.740 0.900 0.605 *A D H*]	[0.128 0.121 0.114 0.128 0.135]
c_2	1	[0.830 0.700 0.710 0.930 0.650 *B E G*]	[0.100 0.100 0.100 0.100 0.100]
c_3	1	[0.480 0.390 0.310 0.510 0.250 *B E G*]	[0.100 0.100 0.100 0.100 0.100]
c_4	1	[0.810 0.680 0.760 0.870 0.620 *C E G*]	[0.100 0.100 0.100 0.100 0.100]
c_5	2	[0.260 0.305 0.405 0.195 0.290 *C E G*]	[0.114 0.107 0.107 0.107 0.114]
c_6	3	[0.517 0.470 0.217 0.350 0.633 *A F K*]	[0.115 0.120 0.135 0.120 0.115]

產生的新的訓練資料集 S'_{tr} 如下:

$$\mathrm{X}' = \begin{bmatrix} \mathbf{x}_1' \\ \mathbf{x}_2' \\ \mathbf{x}_3' \\ \mathbf{x}_4' \\ \mathbf{x}_5' \\ \mathbf{x}_6' \end{bmatrix} = \begin{bmatrix} 0.800 & 0.705 & 0.740 & 0.900 & 0.605 & A & D & H \\ 0.830 & 0.700 & 0.710 & 0.930 & 0.650 & B & E & G \\ 0.480 & 0.390 & 0.310 & 0.510 & 0.250 & B & E & G \\ 0.810 & 0.680 & 0.760 & 0.870 & 0.620 & C & E & G \\ 0.260 & 0.305 & 0.405 & 0.195 & 0.290 & C & E & G \\ 0.517 & 0.470 & 0.217 & 0.350 & 0.633 & A & F & K \end{bmatrix}, \quad \mathrm{y}' = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 2 \\ 2 \end{bmatrix}$$

之後我們就用新產生的訓練資料集 S'_{tr} 建立分類器模型。

第四章　實驗與結果

我們在本章將以實驗的方式來呈現我們所提出的自建構式模糊分群法與兩種資料過濾法:分層抽樣[25][17]，DROP3 [35]以及兩種資料萃取法:*k*-means [1][36][13]，RSP3 [29]的效果差異。我們所使用的實驗資料集分別來自於 UCI 提供的機器學習資料庫[2]，Statlog 資料庫中的資料集[23]以及由 MIT Lincoln Labs 提供的 1998 DARPA 入侵偵測資料集[30]。在實驗裡為了方便表示，我們分別將自建構式模糊分群演算法和分層抽樣以縮寫 SCFC 和 SS 表示。最後，我們再分別以支援向量機(SVM) [6][32][8]，C4.5 決策樹[26][27]和 *k*-nearest neighbor(*k*-NN) [7]當分類器，測試每種縮減法縮減後的新資料集的分類效果。我們實驗所使用的電腦設備: 主機板 P5BV-C TS100-E5。處理器 QuadCore Intel Xeon X3330@2.66GHz。記憶體 2.00GB DDR2-800。硬碟 WDC WD10EADS-00L5B1(931GB)。晶片組北橋 Intel Bigby 3200，南橋 Intel 82801GR ICH7R。作業系統 Microsoft Windows XP Professional Version 2002 Service Pack 3。使用軟體 Matlab Version 7.7.0.741 (R2008b)，WEKA Version 3.6.0。

4.1 資料集的描述

我們所使用的實驗資料集分別為由 UCI [2]所提供的: pendigit 和 covertype，由 Statlog 資料庫[23]提供的: satimage、letter 和 shuttle 以及由 MIT Lincoln Labs 提供的 1998 DARPA 入侵偵測資料集[30]: 1999 KDD Cup。

pendigit 為一組 0 到 9 的數字資料集，此資料集包含了 10,992 筆資料，其中 7,494 筆為訓練資料 3,498 筆為測試資料，每一筆資料由 16 個連續屬性表示。covertype 資料集包含了 581,012 筆資料且分成 7 個類別，我們將 387,369 筆當訓練資料，193,643 筆當測試資料，每

一筆資料包含了 10 個連續屬性和 44 個二元屬性。covertype 為一分布不均勻的資料集，類別一與類別二的資料約佔了全部的 85%。satimage 資料集包含了 6,435 筆資料且資料分為 6 個類別，其中 4,435 筆為訓練資料，2000 筆為測試資料，並以 36 個連續屬性表示每筆資料。letter 為一 20,000 筆 A 到 Z 的字母影像資料集，此資料集的組成方式是由 20 種不同的字型以及將每張字母影像在水平和垂直方向做些微扭曲所組成的，其中 15,000 筆為訓練資料 5,000 筆為測試資料，每張影像用 16 個連續屬性表示。shuttle 資料集包含了 58,000 筆資料，訓練資料佔 43,500 筆，測試資料佔 14,500 筆，此資料集分為 7 個類別，每一筆資料用 9 個連續屬性表示。shuttle 為一分布不均勻的資料集，有將近 80%的資料屬於類別一。1999 KDD 的資料集裡包含了 4,898,431 筆訓練資料和 311,029 筆測試資料。不論是訓練或測試資料集裡都包含了正常型態的網路傳輸資料(Normal)以及四種主要的攻擊型態資料: Denial-of-Service(DoS)，Probing(Prob)，User-to-Root (U2R)以及 Remote-to-Local (R2L)。每一筆資料以 34 個連續屬性和 7 個離散屬性表示。1999 KDD 資料集的分佈極不平均，約有將近 20%資料屬於 normal 類別，80%資料屬於 DoS 類別。所有的資料集在做資料縮減之前我們先將連續屬性的資料正規化到-1 至 1 之間。

4.2 實驗一

遞增式的資料分群演算法最大的缺點在於分群結果會受資料輸入順序影響，換句話說，不同的資料順序會得到不同的結果。然而，我們所提出的 SCFC 對於資料順序對縮減結果的影響並不敏感。SCFC 在不同的資料順序下所產生的資料縮減結果在做分類時對準確率的影響不大。為了測試資料順序對 SCFC 的影響，我們的實驗以 10 種不同資料順序及三種分類器做測試。結果如表 4.1 所示，每一欄位所表示的為平均準確率以及標準差。而我們每一組資料集所用的 ρ 則如表 4.2 所示。

表 4.1: 對六組測試資料集以不同資料順序測試 SCFC 的結果

Datasets	Pendigit	Covertype	Satimage
No. of sampled data	636.10±28.02	645.90±17.72	1219.20±49.20
Reduction rate (%)	91.51±0.37	99.84±0.01	72.51±1.11
SVM (%)	97.23±0.29	62.02±1.73	88.05±0.82
C4.5 (%)	80.76±1.43	58.45±3.41	79.33±1.61
1-NN (%)	95.86±0.37	50.80±1.47	84.47±0.68
Datasets	Letter	Shuttle	1999 KDD Cup
No. of sampled data	4409.40±44.77	281.10±4.23	1481.8±30.14
Reduction rate (%)	70.60±0.30	99.35±0.01	99.70±0.01
SVM (%)	96.25±0.18	99.66±0.15	92.01±0.25
C4.5 (%)	76.87±0.60	98.77±0.40	91.79±0.53
1-NN (%)	93.63±0.20	94.64±0.60	92.30±0.13

表 4.2: 對六組資料集所使用的參數 ρ

Datasets	Pendigit	Covertype	Satimage
ρ	0.502^{16}	0.619^{10}	0.5275^{36}
Datasets	Letter	Shuttle	1999 KDD Cup
ρ	0.44^{16}	0.757	10^{-12}

從表 4.1 我們可以看出 SCFC 對於六組資料集的平均縮減率，pendigit 約 91.51%，covertype 約 99.84%，satimage 約 72.51%，letter 約 70.60%，shuttle 約 99.35%，1999 KDD Cup 約 99.70%。然而在我們所使用的資料集裡 Covertype、Shuttle 以及 1999 KDD Cup 這三組資料集裡每類別的資料量差異較大，Covertype 資料集的類別一和類別二的資料量各佔了全部的 41.47%和 44.13%。Shuttle 資料集裡類別一的資料則佔了 78.41%；而 1999 KDD Cup 資料集裡約有將近 20%資料屬於 Normal 類別，80%資料屬於 DoS 類別。在這種資料分布懸殊的資料集裡，若我們每一類別都用相同的 ρ 可能會導致分類器對部

分類別的資料學習不足，因此我們對這幾組資料集的ρ依照類別分別作了些微的變動，如表 4.3 所示。而我們以新的ρ去做 SCFC 後再以三種分類器做測試，其結果表示如表 4.4。我們發現面對類別數量分布差異大的資料集時，依照每一類別各自取ρ的效果會比用相同的ρ效果略好。

表 4.3: 針對 Covertype, Shuttle, 1999 KDD Cup 三組資料集每一類別各自設定參數ρ

Datasets	Covertype	Shuttle	1999 KDD Cup
Class	ρ for each class	ρ for each class	ρ for each class
C_1	0.55^{10}	0.75	10^{-13}
C_2	0.7^{10}	0.55	10^{-10}
C_3	0.5^{10}	0.05	10^{-10}
C_4	0.4^{10}	0.92	10^{-7}
C_5	0.4^{10}	0.5	10^{-7}
C_6	0.4^{10}	0.65	-
C_7	0.45^{10}	0.9	-

表 4.4: Covertype, Shuttle, 1999 KDD Cup 三組資料集每一類別以不同ρ測試 SCFC 的結果

Datasets	Covertype	Shuttle	1999 KDD Cup
No. of sampled data	640.40±28.83	284.40±2.01	1470.80±27.92
Reduction rate (%)	99.83±0.01	99.35±0.01	99.70±0.01
SVM (%)	67.73±0.86	99.80±0.02	91.97±0.26
C4.5 (%)	66.26±1.98	99.90±0.02	92.05±0.52
1-NN (%)	61.64±1.25	99.50±0.13	92.28±0.17

從以上的實驗結果，我們發現 SCFC 演算法對於分佈密集的資料有較高的縮減率，而對於分佈鬆散的資料其縮減率則較低。例如:大

部分的惡意攻擊軟體會以傳送大量封包的方式來癱瘓網路頻寬或是目標系統的資源。換句話說，1999 KDD Cup 資料集是一組分佈密集的資料集，因此它在 SCFC 可以達到很高的縮減率。然而在 letter 資料集裡，此資料集的產生方式會將字母影像資料在水平和垂直方向做些微的扭曲，所以此資料集的資料分布相較於 1999 KDD Cup 是較鬆散的，所得到的縮減率相對也比 1999 KDD Cup 來的低。更重要的是我們從縮減率的平均值和標準差來看，資料順序對於 SCFC 的縮減效果影響並不大。

從表 4.5 的結果我們可以得知，SCFC 可以處理不同類別間資料量差異大的問題。假定資料集裡某一類別的資料分佈密集，則 SCFC 在此類別會得到較少的代表資料。相反的，如果某一類別的資料較鬆散，則 SCFC 會以較多的代表資料來表示此類別。以 1999 KDD Cup

表 4.5: SCFC 對於 covertype, shuttle, 1999 KDD Cup 三組不同資料集做資料縮減的詳細結果

Datasets	Covertype		Shuttle		1999 KDD Cup (10%)	
Class	Original	Reduced	Original	Reduced	Original	Reduced
C_1	160631	93	34108	134	97278	805
C_2	170945	410	37	9	391458	351
C_3	25000	22	132	9	4107	213
C_4	709	2	6748	76	52	34
C_5	4150	10	2458	36	1126	62
C_6	12254	5	6	5	-	-
C_7	13680	24	11	11	-	-
Total	387369	566	43500	280	494021	1465

資料集為例，類別一的資料分佈較鬆散，此類資料的縮減率為 99.17%，而類別二的資料分佈較密集，此類資料的縮減率則較高為

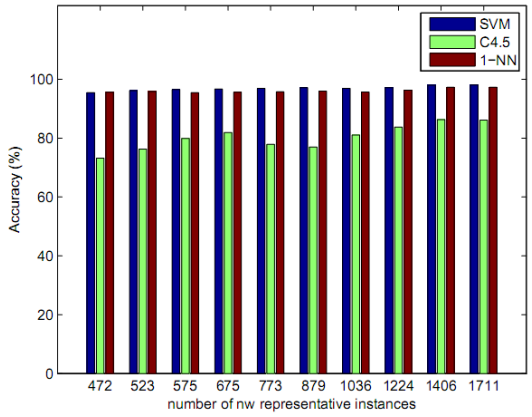
99.91%。1999 KDD Cup 資料集有個有趣的現象，雖然在原始資料裡類別二的資料比類別一多，但縮減後的結果，類別二的代表資料卻比類別一的代表資料少，且這種現象並不影響分類的準確度。

4.3 實驗二

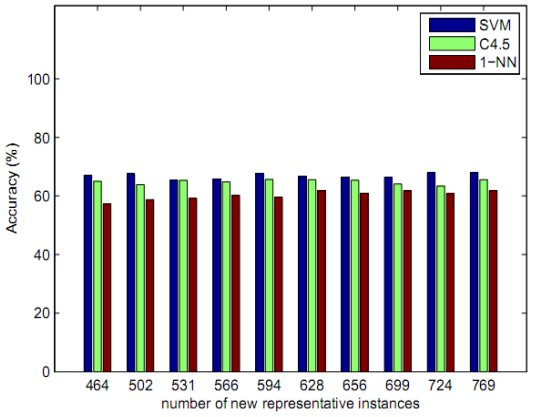
ρ 為 SCFC 的門檻值，如果 ρ 增加，取得的代表資料量也會跟著增加，若 ρ 減少，則代表資料量也隨之減少。在本實驗裡，我們由大到小定義不同的 ρ 並得到不同數量的代表資料。我們將這些代表資料以不同的分類器做測試，並觀察不同的 ρ 對於 SCFC 的縮減率以及分類準確率的影響。以 1999 KDD Cup 資料集為例，我們將此資料集的 ρ 定在 10^{-3} 到 10^{-12} 之間。圖 4.1 為將每組資料集各自以 10 個不同的 ρ 做 SCFC，得到的代表資料分別以三個不同的分類器測試分類準確率(%)。縱軸表示分類正確的百分比，橫軸表示代表資料的數量。我們將這個實驗的結果整理在表 4.6 中，表中每一個欄位表示用 10 個不同 ρ 值測試所得到的平均準確率及標準差。同樣的，我們從平均值和標準差中可以發現 ρ 對 SCFC 僅影響其取出代表資料的數量多寡，對於取出的代表點的分類效果影響並不明顯。

表 4.6: SCFC 用不同參數 ρ 對於六組測試資料集的結果

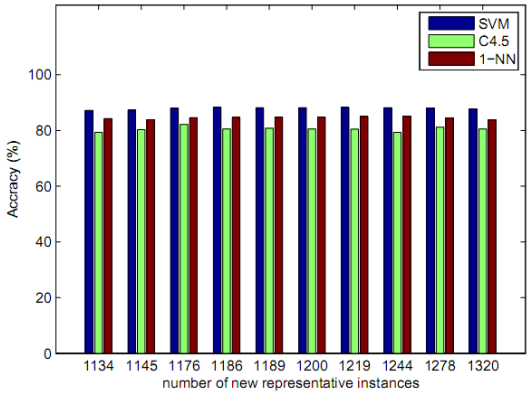
Datasets	Pendigit	Covertype	Satimage
No. of sampled data	927.40±412.02	613.30±99.97	1209.10±57.95
Reduction rate (%)	87.63±5.50	99.84±0.03	72.74±1.31
SVM (%)	96.90±0.79	66.92±0.88	87.90±0.36
C4.5 (%)	80.35±4.32	64.86±0.82	80.50±0.83
1-NN (%)	96.08±0.67	60.21±1.49	84.51±0.43
Datasets	Letter	Shuttle	1999 KDD Cup
No. of sampled data	1278.40±488.54	331.20±77.01	1786.20±703.64
Reduction rate (%)	71.48±3.26	99.24±0.18	99.64±0.15
SVM (%)	96.19±0.34	99.81±0.02	92.25±0.14
C4.5 (%)	74.70±1.82	99.91±0.02	91.58±0.77
1-NN (%)	93.09±0.51	99.36±0.14	90.71±1.27



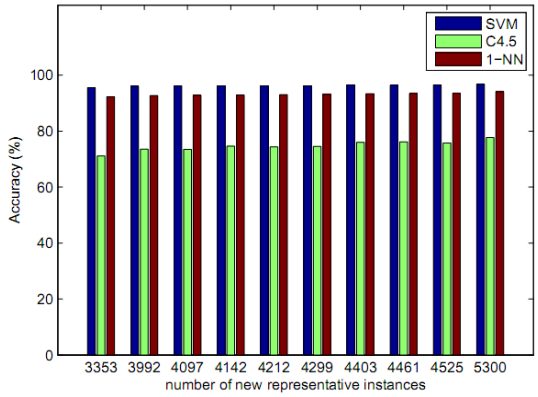
(a) Pendigit dataset



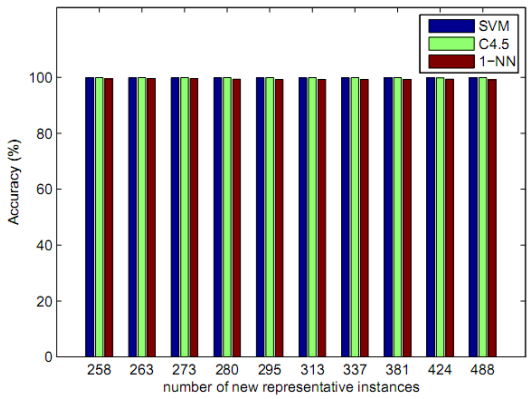
(b) Covertype dataset



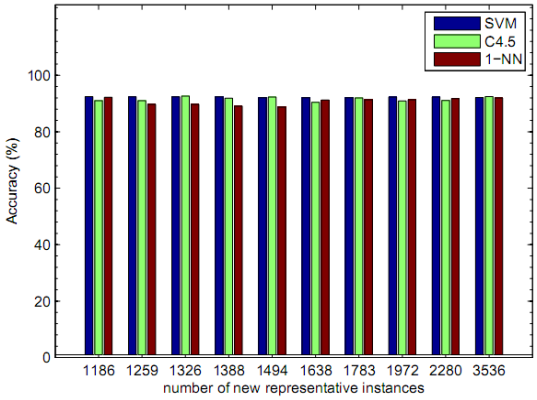
(c) Setimage dataset



(d) Letter dataset



(e) Shuttle dataset



(f) 1999 KDD Cup dataset

圖 4.1: SCFC 用不同的參數 ρ 對於六組測試資料的準確率

4.4 實驗三

在本實驗裡我們將比較我們所提出的方法和其他的資料縮減方

法的效果。由於分層抽樣(SS)和 *k*-means 必須事先決定所要取的代表資料數量，因此我們以 SCFC 的數量為標準，SS 和 *k*-means 每一類別的代表資料數量皆依照 SCFC 的數量來取。而 RSP3 和 DROP3 這兩種方法則不必事先決定代表資料的數量。實驗的結果整理在表 4.7 到表 4.12，其中列出了縮減率、執行時間以及準確度。由於 SS 的結果會受到抽樣出來的代表資料影響，而 *k*-means 的分群結果則與每一群一開始隨機給的初始中心有關，因此我們將這兩種方法做 10 次，並以一個平均結果來表示這兩種方法的效果。表中除了 Original，RSP3 和 DROP3 這幾欄外，其他的欄位都以平均值和標準差的方式表示。顯然的，由於 SS 不需做額外的運算，因此速度最快。然而，如果在抽樣時挑選到不適合的資料則可能導致分類效果不佳。以 1999 KDD Cup 的資料集為例，SS 的資料對於三種分類器的準確率分別為 SVM:76.05±25.46%，C4.5:74.58±24.38%，1-NN:76.09±44.75%。

表 4.7: 比較不同的資料縮減方法對於 pendigit 資料集的效果

Data reduction method	Original	Filtering-based	
		SS	DROP3
No. of sampled data	7494	653	2476
Reduction rate (%)	0.00	91.26	66.96
Time (sec)	0.00	0.001±0.001	824.17
SVM (%)	98.54	96.51±0.49	98.00
C4.5 (%)	92.05	77.89±1.33	88.42
1-NN (%)	97.74	94.63±0.44	97.14
Data reduction method	Abstraction-based		
	SCFC	*k*-means	RSP3
No. of sampled data	636.10±28.02	653	857
Reduction rate (%)	91.51±0.37	91.29	88.56
Time (sec)	0.79±0.03	3.22±0.19	46.56
SVM (%)	97.23±0.29	97.70±0.19	98.14
C4.5 (%)	80.76±1.43	74.55±2.31	82.16
1-NN (%)	95.86±0.37	97.00±0.25	97.42

表 4.8: 比較不同的資料縮減方法對於 covertype 資料集的效果

Data reduction method	Original	Filtering-based	
		SS	DROP3
No. of sampled data	387369	641	133230
Reduction rate (%)	0.00	99.83	65.61
Time (sec)	0.00	0.06±0.004	68356.31
SVM (%)	64.96	66.86±1.33	66.43
C4.5 (%)	67.32	65.15±1.99	63.89
1-NN (%)	60.63	60.22 ±1.37	60.57
Data reduction method	Abstraction-based		
	SCFC	*k*-means	RSP3
No. of sampled data	640.40±28.83	641	89966
Reduction rate (%)	99.83±0.01	99.83	76.78.
Time (sec)	153.62±11.35	5748.57±1184.95	28851.20
SVM (%)	67.73±0.86	67.79±1.24	68.59
C4.5 (%)	66.26±1.98	66.57±1.71	63.52
1-NN (%)	61.64±1.25.	63.82±0.76	59.52

表 4.9: 比較不同的資料縮減方法對於 satimage 資料集的效果

Data reduction method	Original	Filtering-based	
		SS	DROP3
No. of sampled data	4436	1231	1350
Reduction rate (%)	0.00	72.24	69.56
Time (sec)	0.00	0.001±0.001	458.04
SVM (%)	91.65	85.65±0.42	87.80
C4.5 (%)	85.30	80.88±1.02	83.35
1-NN (%)	89.05	86.93±0.68	88.00
Data reduction method	Abstraction-based		
	SCFC	*k*-means	RSP3
No. of sampled data	1219.20±49.20	1231	1207
Reduction rate (%)	72.51±1.11	72.24	72.78
Time (sec)	1.69±0.13	14.47±0.43	37.25
SVM (%)	88.05±0.82	88.16±0.33	86.05
C4.5 (%)	79.33±1.61	79.84±1.63	80.88
1-NN (%)	84.47±0.68	89.60±0.59	86.39

表 4.10: 比較不同的資料縮減方法對於 letter 資料集的效果

Data reduction method	Original	Filtering-based	
		SS	DROP3
No. of sampled data	15000	4403	5025
Reduction rate (%)	0.00	70.65	66.50
Time (sec)	0.00	0.003±0.001	1564.41
SVM (%)	97.98	94.41±0.19	95.10
C4.5 (%)	87.66	77.48±0.93	79.76
1-NN (%)	95.66	90.56±0.35	92.30
Data reduction method	Abstraction-based		
	SCFC	*k*-means	RSP3
No. of sampled data	4409.49±44.77	4403	5821
Reduction rate (%)	70.60±0.30	70.65	61.19
Time (sec)	3.62±0.06	12.83±0.23	205.87
SVM (%)	96.25±0.18	96.81±0.11	97.66
C4.5 (%)	76.87±0.60	76.46±0.76	78.56
1-NN (%)	93.63±0.20	95.32±0.15	95.48

表 4.11: 比較不同的資料縮減方法對於 shuttle 資料集的效果

Data reduction method	Original	Filtering-based	
		SS	DROP3
No. of sampled data	43500	280	15614
Reduction rate (%)	0.00	99.36	64.10
Time (sec)	0.00	0.006±0.001	5389.00
SVM (%)	99.79	97.66±1.13	99.74
C4.5 (%)	99.95	98.98±1.24	99.92
1-NN (%)	99.94	96.86±1.20	99.84
Data reduction method	Abstraction-based		
	SCFC	*k*-means	RSP3
No. of sampled data	284.40±2.01	280	279
Reduction rate (%)	99.35±0.01	99.36	99.36
Time (sec)	7.40±0.10	51.42±29.05	2070.48
SVM (%)	99.80±0.02	97.90±0.70	98.54
C4.5 (%)	99.90±0.02	96.58±2.90	98.69
1-NN (%)	99.50±0.13	98.48±0.64	99.81

表 4.12: 比較不同的資料縮減方法對於 1999 KDD Cup 資料集的效果

Data reduction method	Original	Filtering-based	
		SS	DROP3
No. of sampled data	494021	1465	49403
Reduction rate (%)	0.00	99.70	90.00
Time (sec)	0.00	0.02±0.001	157795.70
SVM (%)	92.51	76.05±25.46	92.41
C4.5 (%)	92.83	74.58±24.38	92.51
1-NN (%)	92.11	76.09±44.75	92.50
Data reduction method	Abstraction-based		
	SCFC	*k*-means	RSP3
No. of sampled data	1470.80±27.92	1465	1555
Reduction rate (%)	99.70±0.01	99.70	99.69
Time (sec)	447.71±6.50	12618.35±2148.59	218208.58
SVM (%)	91.97±0.26	87.56±5.21	92.20
C4.5 (%)	92.05±0.52	69.80±26.76	83.08
1-NN (%)	92.28±0.17	90.42±4.38	92.06

此標準差高達 44.75%的原因在於有將近 80%的資料為 DoS 類別，可是在做測試資料分類時，許多資料被判斷為 Normal 類別。我們可以明顯的從表 4.7 到表 4.12 中看出 SCFC 在執行資料縮減的速度上遠比 *k*-means，RSP3，DROP3 快的多，僅次於 SS。以表 4.8 為例，SCFC 需要 153.62 秒，且產生了 640 筆代表資料，*k*-means 則需 5748.57 秒。而 RSP3 的運算時間為 28851.20 秒，產生的代表資料為 89966 筆。DROP3 則要 68356.31 秒，代表資料的數量為 133230 筆。雖然在速度上 RSP3 和 DROP3 明顯較慢，但在縮減率上 RSP3(76.78%)比 DROP3(65.61%)較高。同時我們發現了一個有趣的現象，當雜訊資料被去除時，會讓整體準確率上升，例如以 SCFC 的結果做 SVM 所得到分類準確率(67.73%)比用全資料做 SVM(64.96%)來的高。以表 4.11 為例，SCFC 需花費 7.40 秒，並產生 280 筆代表資料，*k*-means 則要 51.42 秒。而 RSP3 和 DROP3 分別需要 2070.48 秒和 5389 秒，且 DROP3 產生 15614 筆代表資料。雖然在速度上 RSP3 和 DROP3 依然比其他

方法慢，但在縮減率上 RSP3(99.36%)比 DROP3(41.12%)高出許多。以 SCFC 當資料縮減的方法比用 *k*-means，SS 以及 RSP3 在分類準確度上有較好的效果，雖然 DROP3 的分類準確率比 SCFC 略高一些，但在縮減率上 DROP3(41.12%)卻比 SCFC(99.35%)要低許多。從表 4.12 中我們得知，SCFC 需要 447.71 秒做資料縮減，並產生 1470 筆新的代表資料，而 *k*-means 則需要 12618.35 秒。RSP3 和 DROP3 則分別需要 218208.58 秒和 157795.70 秒，且 DROP3 產生的新的代表資料數量為 49403 筆。雖然速度上 RSP3 和 DROP3 比其他方法慢，但在縮減率上 RSP3(99.70%)比 DROP3(90.00%)較高。SCFC 所得到的資料在分類準確度上較 *k*-means，SS 以及 RSP3 的方法來的高。DROP3 (92.41%, 92.51%, 92.50%)雖然分類準確度比 SCFC (91.97%, 92.05%, 92.28%)高一些，但 DROP3(90.00%)的縮減率卻比 SCFC(99.70%)低。SS 的資料在分類效果上表現較差，以 1999 KDD Cup 資料集為例，將 SS 的結果以 SVM 做測試，得到的平均準確率僅有 76.05%。我們所提出的 SCFC 可以處理大型的資料，我們試著對原始的 1999 KDD Cup 資料集做縮減，此資料集包含了 4,898,431 筆資料。在做 SCFC 時需要 2048.66 秒並得到了 3152 筆新的代表資料，結果如表 4.13 所示。

表 4.13: SCFC 對於 1999 KDD Cup 資料集做資料縮減的詳細結果

Datasets	1999 KDD Cup – whole data				
Class	Original data	Reduced data	SVM(%)	C4.5(%)	1-NN(%)
C_1	972781	1729	98.15	97.48	98.18
C_2	3883370	451	96.96	96.77	97.12
C_3	41102	879	82.50	74.58	81.30
C_4	52	33	7.46	3.95	14.47
C_5	1126	60	9.23	3.32	7.96
Total	4898431	3152	92.38	91.68	92.42

將 SCFC 後的結果做 SVM 可得到 92.38%的分類準確率。雖然以 *k*-means 做資料縮減在某些資料集(satimage 和 covertype)可以得到較

好的分類準確度，但它受初始中心點的影響很大。以 1999 KDD Cup 資料集為例，將 *k*-means 後的結果以 C4.5 測試分類準確率，我們得到很高的準確率標準差(26.76%)。此外，*k*-means 還必須事先決定代表資料的數量。

4.5 實驗四

我們在前面提到了兩種不需要事先決定代表點數量多寡的分群方法 RSP3 與 DROP3。在本實驗中，將比較我們的方法與 RSP3 以及 DROP3 在速度上與分類效果上的好壞。由於 RSP3 和 DROP3 皆沒有參數可供調整，因此我們的做法是用調整 SCFC 的參數 ρ 的方式使得取出來的資料量與 RSP3 或 DROP3 取出的代表點數量相當，我們在這種基準點上比較這三種資料縮減的速度以及代表資料在分類效果上的好壞。其結果分別如表 4.14 及表 4.15 所示。

表 4.14: 調整六組資料集的 ρ 當 SCFC 的數量與 RSP3 相近時的效果

Datasets	Pendigit	Covertype	Satimage
No. of sampled data	854.9±29.32	91212±2078.6	1205±20.46
Reduction rate (%)	88.59±0.39	76.45±0.54	72.84±0.46
Time (sec)	0.84±0.03	13923±844.98	1.30±0.10
SVM (%)	97.41±0.34	66.63±0.22	88.59±0.84
C4.5 (%)	81.38±1.38	64.74±1.16	78.19±1.60
1-NN (%)	96.37±0.25	59.61±0.08	84.70±0.37
Datasets	Letter	Shuttle	1999 KDD Cup
No. of sampled data	5843±53.37	277.30±4.14	1570.9±24.77
Reduction rate (%)	61.05±0.36	99.36±0.01	99.68±0.01
Time (sec)	4.29±0.04	5.23±0.05	388.76±4.76
SVM (%)	97.18±0.12	99.76±0.06	92.05±0.19
C4.5 (%)	80.67±0.78	98.76±0.39	92.14±0.21
1-NN (%)	94.80±0.11	94.82±0.48	92.34±0.10

表 4.15: 調整六組資料集的ρ當 SCFC 的數量與 DROP3 相近時的效果

Datasets	Pendigit	Covertype	Satimage
No. of sampled data	2468±30.97	129743±3094.2	1357.10±46.72
Reduction rate (%)	67.07±0.41	66.51±0.80	69.41±1.05
Time (sec)	1.30±0.07	22229±2533.4	1.17±0.10
SVM (%)	98.24±0.10	66.27±0.20	88.64±0.70
C4.5 (%)	89.57±0.57	65.62±0.98	79.42±1.13
1-NN (%)	97.51±0.20	60.09±0.14	84.69±0.74
Datasets	Letter	Shuttle	1999 KDD Cup
No. of sampled data	5278±59.11	15560±27.52	47653±63.51
Reduction rate (%)	64.81±0.39	64.23±0.06	90.35±0.01
Time (sec)	3.89±0.04	215.94±1.74	4367.1±287.46
SVM (%)	97.01±0.19	99.92±0.01	92.02±0.01
C4.5 (%)	78.89±0.65	99.94±0.01	92.38±0.03
1-NN (%)	94.29±0.24	99.94±0.01	92.28±0.07

我們觀察表 4.7 到表 4.12 的 RSP3 結果及表 4.14，我們發現在取出資料量相近的情況下，SCFC 做資料縮減的時間遠比 RSP3 所需的時間要少。Pendigit 做 SCFC 需要 0.84 秒，做 RSP3 則需 46.56 秒。Covertype 做 SCFC 需要 13923 秒，做 RSP3 則需 28851.20 秒。Satimage 做 SCFC 需要 1.30 秒，做 RSP3 則需 37.25 秒。Letter 做 SCFC 需要 4.29 秒，做 RSP3 則需 61.19 秒。Shuttle 做 SCFC 需要 5.23 秒，做 RSP3 則需 2070.48 秒。1999 KDD Cup 做 SCFC 需要 388.76 秒，做 RSP3 則需 218208.58 秒。而在分類準確方面除了 Shuttle 在 1-NN 上 RSP3(99.81%)比 SCFC(94.82%)高，以及 1999 KDD Cup 在 C4.5 上 SCFC(92.14%)比 RSP3(83.08%)高，其餘的分類準確率測試的結果，這兩種方法的結果都很接近。

同樣的，我們觀察表 4.7 到表 4.12 的 DROP3 結果及表 4.15，我們也發現在取出資料量相近的情況下，SCFC 做資料縮減的時間也比 DROP3 所需的時間少很多。Pendigit 做 SCFC 需要 1.30 秒，做 DROP3

則需 824.17 秒。Covertype 做 SCFC 需要 22229 秒，做 DROP3 則需 68356.31 秒。Satimage 做 SCFC 需要 1.17 秒，做 DROP3 則需 458.04 秒。Letter 做 SCFC 需要 3.89 秒，做 DROP3 則需 1564.41 秒。Shuttle 做 SCFC 需要 215.94 秒，做 DROP3 則需 5389.00 秒。1999 KDD Cup 做 SCFC 需要 4367.1 秒，做 DROP3 則需 157795.70 秒。而在分類準確方面除了 Satimage 在 C4.5 上 DROP3(83.35%)比 SCFC(79.42%)略高以及 1-NN 上 DROP3(88.00%)比 SCFC(84.69%)略高，其餘的分類準確率測試的結果，這兩種方法的結果都很接近。

綜合本章的實驗結果，我們發現 SS 為速度最快的方法，原因在於它不需要額外的運算，然而他所選出的資料卻未必適合分類。SCFC 不僅速度上比 *k*-means，RSP3 及 DROP3 快，它所萃取出來的代表資料也較適合分類器使用。RSP3 雖然縮減率和 SCFC 差不多，但它花了太多時間在計算兩筆資料間的距離。DROP3 對於分類的效果雖然不錯，可是它在資料縮減時耗費過多時間，且對於大型的資料其資料的縮減效果不如 SCFC 或 RSP3 來的好。而 *k*-means 的結果則受到一開始隨機給的初始中心影響很大。

第五章　結論

我們實現了自建構式模糊分群演算法(SCFC)，我們以此演算法做資料縮減，並將此方法和兩個資料縮減法:分層抽樣和 DROP3 以及兩個資料萃取法:*k*-means 和 RSP3 做比較。我們以六組資料集做實驗測試，實驗的結果顯示 SCFC 在速度及縮減率上都比其他的方法要好，更重要的是它可以有效的處理大型的資料集。同時我們也發現資料縮減對於雜訊較高的資料集有抑制雜訊的效果。

參考文獻

[1] M. B. de Almeida, A. de Padua Braga, and J. P. Braga, "SVM-KM: speeding SVMs learning with a priori cluster selection and k-means," in *Proceedings of the 6th Brazilian Symposium on Neural Networks*, pp. 162–167, November 2000.

[2] A. Asuncion and D. Newman, "UCI machine learning repository," 2007. [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html

[3] L. Bottou, C. Cortes, J. S. Denker, H. Drucker, I. Guyon, L. D. Jackel, Y. LeCun, U. A. Müller, E. Säckinger, P. Simard, and V. Vapnik, "Comparison of classifier methods: a case study in handwriting digit recognition," in *International Conference on Pattern Recognition ICPR94*. Jerusalem, Israel: IEEE Computer. Society Press, vol. 2, pp. 77–87, September 1994.

[4] L. Breiman, J. H. Friedman, C. J. Stone, and R. A. Olshen, *Classification and Regression Trees*, 1st ed. Boca Raton, FL, USA: Chapman & Hall/CRC, January 1984.

[5] H. Brighton and C. Mellish, "Reduction techniques for instance-based learning algorithms," *Data Mining and Knowledge Discovery*, vol. 6, no. 2, pp. 153–172, April 2002.

[6] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, September 1995.

[7] T. M. Cover and P. E. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, January 1967

[8] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machinesand other kernel-based learning methods*. New York, NY, USA: Cambridge University Press, March 2000.

[9] P. Datta and D. Kibler, "Symbolic nearest mean classifiers," in *Proceedings of the 14th International Conference on Machine Learning*, pp. 82–87, July 1997.

[10] G. W. Gates, "The reduced nearest neighbor rule," *IEEE Transactions on Information Theory*, vol. 18, no. 3, pp. 431–433, May 1972.

[11] S.-W. Kim and B. J. Oommen, "Enhancing prototype reduction schemes with recursion: A method applicable for "large" data sets," *IEEE Transactions on*

Systems, Man, and Cybernetics, part B: Cybernetics, vol. 34, no. 3, pp. 1384–1397, June 2004.

[12] G. J. Klir and B. Yuan, *Fuzzy sets and fuzzy logic: theory and applications*, 1st ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, May 1995.

[13] R. Koggalage and S. K. Halgamuge, "Reducing the number of training samples for fast support vector machine classification," *Neural Information Processing - Letters and Reviews*, vol. 2, no. 3, pp. 57–65, March 2004.

[14] U. H.-G. Kreßel, "Pairwise classification and support vector machines," in *Advances in Kernel Methods: Support Vector Learning*. Cambridge, MA, USA: MIT Press, pp. 255–268, 1999.

[15] X. Wu, V. Kumar, J. R. Quinlan, J. Ghosh, Q. Yang, H. Motoda, G. J. McLachlan, A. F. M. Ng, B. Liu, P. S. Yu, Z.-H. Zhou, M. Steinbach, D. J. Hand, and D. Steinberg, "Top 10 algorithms in data mining," *Knowledge and Information Systems*, vol. 14, no. 1, pp. 1–37, January 2008.

[16] W. Lam, C.-K. Keung, and C. X. Ling, "Learning goodprototypes for classification using filtering and abstraction of instances," *Pattern Recognition*, vol. 35, no. 7, pp. 1491–1506, July 2002.

[17] Y.-J. Lee and S.-Y. Huang, "Reduced support vector machines: A statistical theory," *IEEE Transactions on Neural Networks*, vol. 18, no. 1, pp. 1–13, January 2007.

[18] K.-M. Lin and C.-J. Lin, "A study on reduced support vector machines," *IEEE Transactions on Neural Networks*, vol. 14, no. 6, pp. 1449–1559, November 2003.

[19] Y. Linde, A. Buzo, and R. Gray, "An algorithm for vector quantizer design," *IEEE Transaction on Communications*, vol. 28, no. 1, pp. 84–95, January 1980.

[20] M. Lozano, J. M. Sotoca, J. S. Sánchez, F. Pla, E. Pękalska, and R. P. W. Duin, "Experimental study on prototype optimisation algorithms for prototype-based classification in vector spaces," *Pattern Recognition*, vol. 39, no. 10, pp. 1827–1838, October 2006.

[21] J. Macqueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281–297, 1967.

[22] E. Marchiori, "Hit miss networks with applications to instance selection,"

Journal of Machine Learning Research, vol. 9, pp. 997–1017, June 2008.

[23] D. Michie, D. J. Spiegelhalter, and C. C. Taylor, *Machine Learning, Neural and Statistical Classification*. Englewood Cliffs, N.J.: Prentice Hall, 1994. [Online]. Available: Data available at http://www.ncc.up.pt/liacc/ML/statlog/datasets.html

[24] J. C. Platt, N. Cristianini, and J. Shawe-Taylor, "Largemargin DAGs for multiclass classification," in *Advances in Neural Information Processing Systems*. Cambridge, MA, USA: MIT Press, vol. 12, pp. 547–553, 2000.

[25] E. Pękalska, R. P. W. Duin, and P. Paclík, "Prototype selection for dissimilarity-based classifiers," *Pattern Recognition*, vol. 39, no. 2, pp. 189–208, February 2006.

[26] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, no. 1, pp. 81–106, March 1986.

[27] J. R. Quinlan, C4.5: *Programs for Machine Learning*, 1st ed. San Mateo, CA, USA: Morgan Kaufmann, January 1993.

[28] G. L. Ritter, H. B. Woodruff, S. R. Lowry, and T. L. Isenhour, "An algorithm for a selective nearest neighbor decision rule," *IEEE Transactions on Information Theory*, vol. 21, no. 6, pp. 665–669, November 1975.

[29] J. Sánchez, "High training set size reduction by space partitioning and prototype abstraction," *Pattern Recognition*, vol. 37, no. 7, pp. 1561–1564, July 2004.

[30] S. J. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. K. Chan, "Cost-based modeling for fraud and intrusion detection: Results from the jam project," in *Proceedings of the 2000 DARPA Information Survivability Conference and Exposition*, pp. 130–144, January 2000.

[31] N. A. Syed, H. Liu, and K. K. Sung, "A study of support vectors onmodel independent example selection," in *Proceedings of the 5th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 272–276, August 1999.

[32] V. Vapnik, *The nature of statistical learning theory*, 2nd ed. New York, NY, USA: Springer, November 1999.

[33] J. G. Wang, P. Neskovic, and L. N. Cooper, "Training data selection for support vector machines," in *Proceedings of the 1st International Conference on Advances in Natural Computation*, pp. 554–564, August 2005.

[34] D. L. Wilson, "Asymptotic properties of nearest neighbor rules using edited

data," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 2, no. 3, pp. 408–421, July 1972.

[35] D. R. Wilson and T. R. Martinez, "Reduction techniques for instance-based learning algorithms," *Machine Learning*, vol. 38, no. 3, pp. 257–286, March 2000.

[36] S. Zheng, X. Lu, N. Zheng, and W. Xu, "Unsupervised clustering based reduced support vector machines," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, pp. 821–824, April 2003.