國 立 交 通 大 學

資訊科學與工程研究所

碩 士 論 文

基於動態調整權重之co-cluster演算法

Co-cluster with dynamic weighting

研究生：張智愷

指導教授：李嘉晃 教授

中 華 民 國 一百 年 六 月

基於動態調整權重之 co-cluster

Co-cluster with dynamic weighting

研 究 生：張智愷　　Student：Chih-Kai Chang

指導教授：李嘉晃　　Advisor：Chia-Hoang Lee

國 立 交 通 大 學

資 訊 科 學 與 工 程 研 究 所

碩 士 論 文

A Thesis

Submitted to Institute Computer Science and Engineering

College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer Science

Jun 2011

Hsinchu, Taiwan, Republic of China

中華民國一百年六月

基於動態調整權重之co-cluster

學生 ： 張智愷　　指導教授：李嘉晃 教授

國立交通大學資訊學院 資訊科學與工程研究所碩士班

摘要

由於科技的進步，網路的發展，造成資訊量迅速攀升，然而這樣的進步卻相對的造成使用者必須付出更多的時間去瀏覽所需的文件。有鑒於現今搜尋引擎的廣泛使用，人們希望以更高的效率與效能取得資訊，其中分群的技術應用，扮演著重要的角色。在搜尋的過程中，若能先將文件做好適當的分群，則可讓搜尋系統提供更結構性的結果給使用者。如此一來，不僅可以減少搜尋文件的時間，更可加快使用者找到自己想要的文件。

本研究利用 Co-Clustering 的分群方法為基底並做更進一步的改良，針對分群效能的改善以及 feature 權重的增減加以討論，並且以 Reuters、20newsgroup 及 classic3 資料集做分析，萃取出核心關鍵字，並給予適當的權重，進而過濾一些不必要的雜訊以及加強關鍵字的強度。利用座標的資訊，利用核心關鍵字在距離群中心的距離為基礎做關鍵字之調整權重。接著，利用 logistic function 的特性對關鍵字之權重調整到介於 0 與 1 之間，再將關鍵字賦予調整後權重之後，再做一次 Co-Clustering，重複以上的動作達到收斂後，進而得到較高的分群結果。

Co-clustering with Dynamic Weighting

Student ： Chih-Kai Chang Advisor：Prof. Chia-Hoang Lee

Institute Computer Science and Engineering
College of Computer Science
National Chiao Tung University

Abstract

This paper proposes a weighted co-clustering algorithm and applies it to document clustering problem. The weighted co-clustering is an extension of co-clustering, and it makes use of co-clustering properties to design a dynamic weighting algorithm for terms. Firstly, co-clustering presents both documents and words on the same coordinate system using spectral embedding technique. Secondly, co-clustering clusters documents and words simultaneously, so the documents that are within the same cluster should be clustered together with their corresponding words. Based on these two properties, the weighted co-clustering changes term weights iteratively. In addition, an outlier detection mechanism is proposed in this paper to eliminate outlier documents from clustering process. When the clustering process is completed, these outlier documents are assigned to appropriate clusters. We conduct experiments on three data sets and the experimental results show that the weighted co-clustering can effectively improve the performance.

誌謝

首先，感謝指導教授李嘉晃老師對我的悉心指導，才能有今日的成果。老師就像我的良師益友，時而嚴厲，時而慈祥，不論是研究討論或課堂授課時，所教導我的專業知識和處世道理，都著實讓我獲益良多。這些過程與經驗，都將成為我一生受用無窮的寶庫。

同時，我亦感謝這兩年來陪伴在我身邊的實驗室同學們、學長以及學弟。尤其是我的同學們，而益、士元、俊憲，總是不斷的鼓勵我，對我的幫助更是多不勝數。兩年的時間，雖然不是很長，但是曾經有過的歡笑淚水，這些回憶會一輩子永存在我的心中。

最後，我要感謝我的家人，感謝你們對我的愛護和包容。謝謝你們在背後默默的支持，使我能夠順利的完成碩士學位。

心中有太多的感謝不知道如何表達，在此僅以本篇論文表示我對你們最誠摯的感謝，並祝福你們身體健康、萬事如意，謝謝。

張智愷　謹誌

資訊科學與工程研究所

智慧型系統實驗室

中華民國一百年七月

目錄

圖目錄

表目錄

第一章、緒論

1.1 研究動機

現今的社會科技越來越進步，網路在這個社會中已經跟我們的生活密不可分，隨著網路的蓬勃發展，使得許多資訊的傳遞越來越簡便，從以前的寄信時代到現在的 e-mail，我們可以了解科技的進步讓我們的生活越來越便利，不論是新聞或是書本以及報章雜誌都進入了電子化的時代，進而演變出大量的文章該如何分類的問題，所以如何快速將這些大量的文章做最正確且最有效率的分類，是一個很重要的課題。

由於電腦的普及化，使得每個人都可簡單且輕鬆的擷取到網路上的文件。面對如此龐大的文件，進而造成了讀者閱讀上的困擾，面對每天都有成千上萬的文章在這個網路的世界上發表，這些發表的文章許多都會被拿來當作新聞。然而大量的新聞資訊所引發的問題往往讓人覺得很頭疼，該如何減低人力來讓系統有個自動化的方式整理這些龐大的新聞，讓使用者可以快速的找到其喜歡的新聞文章，這是本研究主要的目的。透過有效率的方式，系統快速且準確的對龐大的文章分群，如此一來便可以加快搜尋文件的速度，讀者可以快速且便利的找到他們想要的文章。

分群的方法可以視為一種藝術，透過群與群之間的關係性，除了可以定義群跟群之間的相似度以外，也可以進階的使用在許多地方，比方說像上敘所說的搜尋，我們也可以利用在導航、興趣分析，更廣泛的說法，其實不只是只能對文章做分群，透過分群的概念一樣可以讓許多系統可以達到更高的效能，比方說分散式系統、多處理器等，在此我們針對文章的分群做研究。我們將針對許多的新聞文章利用分群的方式來將相似的文章分成同一個群，實驗將用有別於一般傳統的分群方式，我們會利用 co-cluster[1]的方法除了可以達到不錯的分群效果以外，我們還可以找到字與文章之間的關係性，藉由此關係聽來做更深一步的探討以及

改進效能，已達到更好的精確度。

1.2 研究目的

由於搜尋引擎的廣泛使用，我們希望在搜尋的過程中，先將文件分群好以後，可以將相關性的文章一併列出，讓使用者除了能夠透過搜尋引擎的搜尋，找出自己所需要的文章，同時可以了解其他隸屬於同一個群的文章，藉此來達到節省時間以及迅速找到想要的文章之目的。

透過文章中字與字的關係性，本研究亦可改善許多雜訊的產生，在大量的文章之中存在許多我們不會去關注甚至於會影響到分群的結果的字，在本研究都會一一探討，並且透過實驗的方法可以有效的降低不重要字眼的權重，並且提升重要字眼的權重，進而提升分群的效果。

當我們要對文章做分群時，我們會建立出一個 vector space model[2]，在最基本的概念下，我們會建立一個 word 的 list，這個 list 中的每一個 word 都是獨立不重複的，然而我們會把每個 word 當成用來表示該篇文章的一種特徵，許多特徵可以表現出一個特徵的向量，也代表著這篇文章將由這條向量所表現的特徵空間來表示。

我們考慮到字的分群其實基本的架構在同一篇文章中有多少的文字與其一起發生，當所有的文章都沒有給定標籤的情況下，該如何利用有效的資訊達到更高的效果，這是本研究所要探討的目的，目前存在的文章分群法包含以下幾種常用的演算法，agglomerative cluster[3]、k-means 演算法[4]、Probabilistic latent semantic analysis(pLSA) [5]、self-organizing maps[6]、multidimensional scaling[7]，對於文章的分群，目標是要將一個集合的文章分到一個群，最近也吸引了越來越多人的興趣，分群可以用來自動檢索文件組成有意義的清單，這也是一個最廣泛使用的技術以用來進行數據的探勘，因為它可以捕捉到自然結構的數據，基本上分群是屬於非監督式的學習，所以它

不需要事先給定群的類別。

分群的目的是為了將目標對象分配成組，使目標對象在同一個群組中更相似，並且與不同組不相似。在分群的演算法分成判別性(discriminative)以及生成性(generative)，而判別性的演算法採用兩兩之間的相似性為基礎，以確定一個目標函數以及優化這個函數來達到目標結果，k-means 是一種典型的判別性演算法，其目的在於最小化目標對象與群中心之間的平方和，另一方面生成演算法是假定數據是由一個基本的參數模型，目標是從觀測數據中估計參數，然而群中心可以進一步的獲得模型與參數，高斯混合模型(The Gaussian mixture mode)是個典型的生成演算法，其使用一個混合多個高斯分佈的模型，解決多種混合分解已經被提出，其中許多集中在最大似然法(maximum likelihood methods)，如最大化期望值(EM)[8]，或最大後驗估計(MAP)。

目前大多數分群演算法集中在一維度的分群[9]，隨著網路速度的發達，以及電腦的計算速度的增長，許多數據探勘(data mining)的應用已經從對簡單的資料型態分群延伸到對多類別資料型態作 co-clustering，但是通常都討論高異質性數據[10]。實際上許多現實例子對象與其相對應的特徵是有互相關聯性的，在一個語料庫中文章與字就是個典型的例子，因為文章是由字組成的；而將字搜集起來就成了文章，直觀上一群被分割的文件應該會驅使著一群字被同時的分割，同樣的當一群字被劃分的同時也意味著一群文章已經被分割了，這也表明了在許多應用中 co-clustering 比分群更有效率的沿用在單一維度中，因此 co-clustering 已經在許多的應用領域中被廣為研究，例如文件的分析(text mining)[1][11]，生物訊息(Bioinformatics)[12][13]、以及資料探勘(data mining)[14]。

1.3 論文架構

第一章：前言，敘述本研究之動機與目的。

第二章：相關研究，敘述本研究之相關研究與背景。

第三章：系統設計，將本研究之系統整體架構與概念方法做一個完整介紹。

第四章：實驗結果與討論，將本研究之系統產生的結果根據不同類型的資料做全面性的分析。

第五章：結論與未來展望，將本研究之系統成果做一總結，並提出結論與探討未來研究之方向。

第二章、相關研究

2.1 Co-clustering 概念

Co-clustering 的概念在先前就有人提到，其組成與 Hofmann[15]等人提出的非監督氏學習(unsupervised learning)框架中使用的成對數據(dyadic data)很相似；一成對的數據會對映至一個由兩種成對關係建立的可見集合，並且將每個元素對映到其中一個集合之中。成對的概念是常用的數據表示方式，常常用在許多領域例如：文件分析(text analysis)、電腦視覺、以及計算式的語言學。比方說在一個語料庫中，文章與字是一個典型的例子，因為文章是由字來組成的，同時字的組成就成了文章。直觀上，一些被區分好的文章，也應該驅使著字的區分，同時區分好的字集合應該意味著一些文章已經被區分好了，這樣的原理類似於 Zhu[16]所提的相互加強原理(mutual reinforcement principle)，並且已經成功的被運用在摘要的系統之中[16][17]。

Co-clustering 或者 bi-clustering 的概念吸引著許多的研究人員開發新的演算法來執行多類型的文件分群的任務，Dhillon[1]利用了 bi-partite 的圖形來分別表示文章和字的關係，然後對文章以及字同時進行分群，其理論是架構於 Spectral clustering[18][19][20]之上，實驗結果顯示，co-clustering 方法可以有效的運用在文件分析的領域。同樣的 Zha[21]等人運用了雙分圖(bi-partite graph)以及 spectral clustering 來執行分群的目的，利用對象以及特徵的對偶關係有效的同時針對對象以及特徵分群，Li[22]提出了一個二元的生成模組，允許模組下的特徵結構(feature structure)與每個群產生連繫，並且採用優化方法(optimization procedure)優化群的結構以及更新群的資訊。

本質上來說，物件與其特徵可以用矩陣來表示，所以 co-clustering 可以建構一個模組用來同時將矩陣的行與列進行分群；因此 co-clustering 可以藉由矩陣的分解[23][24][25]來解決，co-clustering 的模型可以做為優化問題以及涉及三重(triple)的矩陣分解，Long[26]等人使用了其它的矩陣分解技術，稱做

Block Value Decomposition 來分解成對資料的矩陣至一個列系數矩陣、Block Value 矩陣以及行系數矩陣。

在文件的分析，語料庫常被表示成文章與字組的矩陣，其中列代表了文章，行代表了在字典裡面的字，許多的文件探勘(text mining)利用 co-clustering 的技術在近期中被開發出來，Mandhani[27]等人提出一個階層式的 co-clustering 演算法，並且運用在文章與字的矩陣；每個 co-cluster 都是一個子矩陣並且包含了文章所成的群以及與該群有關連的字。Park[28]等人檢視了 spectral clustering 背後的文章以及字，並與 Latent Semantic Analysis(LSA)[29]做比較，發現 spectral co-clustering 以及 LSA 依照同樣的執行程序，並且運用不同的正規劃技術以及度量方式。Bisson 以及 Hussain[30]定義一個相似度的計算稱做**χ**-Sim 迭代的完成物件與特徵的相似度，本篇文章將 Dhillon[1]的 co-clustering 做延伸，提出一個有權重的 co-clustering 演算法，將不利於分群的字給予較低的權重，同時將那些能夠促進分群的字給予較高的權重。

2.2 Spectral Graph Bipartitioning

Spectral clustering[18][19][20]是一個分群演算法，它根據點與點之間的相似度將空間上的結果分群，使得同一個群內的點與點之間的相似度越高越好，不同群的點與點之間的相似度則越低越好。這類演算法討論到了 graph 的 bipartitioning，是一個有效率的 heuristic，早在 1970 年時就被介紹了[31][32][33]，更進一步在 1990 年被推廣[34]。這一節將簡單說明 spectral graph bipartitioning 的基本原理[18]以及著名的幾個找 graph minimum cut 的方法。

2.2.1 符號定義

假設有一個無向圖(undirected graph)$\mathbf{G} = (\mathbf{V}, \mathbf{E})$，圖中每條邊(edge)上都

有權重(weighted)，$\mathbf{V} = \{\mathbf{v_1}, \ldots, \mathbf{v_n}\}$是點(vertex)所形成的集合，$\mathbf{w_{ij}}$為$\mathbf{v_i}$與$\mathbf{v_j}$相連的邊上的權重。其中，$\mathbf{w_{ij}} \geq \mathbf{0}$且$\mathbf{w_{ij}} = \mathbf{w_{ji}}$，若$\mathbf{w_{ij}} = \mathbf{0}$代表$\mathbf{v_i}$與$\mathbf{v_j}$之間無邊相連。

定義 weighted adjacency matrix $\mathbf{W}$:

$$\mathbf{W} = (\mathbf{W_{ij}})_{\mathbf{ij}=\mathbf{1},\ldots,\mathbf{n}}$$

G 中的每個點$\mathbf{v_i} \in \mathbf{V}$的 degree 定義為：

$$\mathbf{d_i} = \sum_{\mathbf{j}=\mathbf{1}}^{\mathbf{n}} \mathbf{w_{ij}}$$

定義 degree matrix D:

$$\mathbf{D} = \begin{bmatrix} \mathbf{d_1} & \cdots & \mathbf{0} \\ \vdots & \ddots & \vdots \\ \mathbf{0} & \cdots & \mathbf{d_n} \end{bmatrix}$$

D 為對角矩陣，且對角線上的值為$\mathbf{d_1}, \ldots, \mathbf{d_n}$。

假設有一V的子集合$\mathbf{A} \subset \mathbf{V}$，則A的補集合(complement)標記為$\bar{\mathbf{A}}$，

定義indicator vector $\mathbf{f_A} = (\mathbf{f_1}, \ldots, \mathbf{f_n})^{\mathbf{T}} \in \mathbb{R}^{\mathbf{n}}$為：

$$\begin{cases} \mathbf{f_i} = \mathbf{1}, & \text{if } \mathbf{v_i} \in \mathbf{A} \\ \mathbf{f_i} = \mathbf{0}, & \text{if } \mathbf{v_i} \in \bar{\mathbf{A}} \end{cases}$$

假設$\mathbf{A}$、$\mathbf{B} \subset \mathbf{V}$且$\mathbf{A} \cap \mathbf{B} = \emptyset$則定義A、B之間的weight為：

$$\mathbf{W}(\mathbf{A}, \mathbf{B}) = \sum_{\mathbf{i} \in \mathbf{A}, \mathbf{j} \in \mathbf{B}} \mathbf{w_{ij}}$$

2.2.2 Laplacian matrix

Laplacian matrix 可以分為 unnormalized 與 normalize。首先說明 unnormalized Laplacian matrix，其定義為：

$$\mathbf{L} = \mathbf{D} - \mathbf{W}$$

矩陣**L**具有下列性質：

(1). 對於所有的向量$\mathbf{f} \in \mathbb{R}^{\mathbf{n}}$，下列式子成立：

$$\mathbf{f^T L f} = \frac{\mathbf{1}}{\mathbf{2}} \sum_{\mathbf{i,j=1}}^{\mathbf{n}} \mathbf{w_{ij}} (\mathbf{f_i} - \mathbf{f_j})^2$$

(2). L 是 symmetric 且 positive semi-definite。

因為**D**與**W**皆為 symmetric，所以 **L** 也是 symmetric。且由(1)可知 L 為 semi-definite。

(3). L 最小的 eigenvalue 為 0，且相對於 0 之 eigenvector 為 1。

由此可知，0 必為 L 之 eigenvalue。且因 L 為 semi-definite，所以所有的 eigenvalue 皆$\geq \mathbf{0}$，由此可知，0 為最小的 eigenvalue。

(4). L 的所有 eigenvalue(n 個)皆為實數，且$\mathbf{0} = \boldsymbol{\lambda}_{\mathbf{1}} \leq \boldsymbol{\lambda}_{\mathbf{2}} \leq \cdots \boldsymbol{\lambda}_{\mathbf{n}}$。

接下來說明 normalized graph Laplacian。normalized graph Laplacian 分為兩種，其定義分別為：

$$\mathbf{L_{sym}} = \mathbf{D}^{-\frac{\mathbf{1}}{\mathbf{2}}} \mathbf{L} \mathbf{D}^{\frac{\mathbf{1}}{\mathbf{2}}}$$

與

$$\mathbf{L_{rw}} = \mathbf{D}^{-\mathbf{1}} \mathbf{L}$$

矩陣$\mathbf{L_{sym}}$**與**$\mathbf{L_{rw}}$**具有下列性質：**

(1). 對於所有的向量$\mathbf{f} \in \mathbb{R}^{\mathbf{n}}$，下式子成立：

$$\mathbf{f^T L_{sym} f} = \frac{\mathbf{1}}{\mathbf{2}} \sum_{\mathbf{i,j=1}}^{\mathbf{n}} \mathbf{w_{ij}} \left(\frac{\mathbf{f_i}}{\sqrt{\mathbf{d_i}}} - \frac{\mathbf{f_j}}{\sqrt{\mathbf{d_j}}} \right)^{\mathbf{2}}$$

(2). $\mathbf{L_{rw} u = \lambda u \Leftrightarrow L_{sym} w = \lambda w}$，其中$\mathbf{w = D^{1/2} u}$。

(3). $\mathbf{L_{rw} u = \lambda u \Leftrightarrow Lu = \lambda Du}$。

(4). $\mathbf{L_{sym}}$與$\mathbf{L_{rw}}$是 positive semi-definite。

由(1)可知L_{sym}為 semi-definite，接著根據(2)可知，若 λ 是L_{sym}的 eigenvalue，則 λ 也是L_{rw}的 eigenvalue，所以L_{rw}的所有 eigenvalue 皆≥ 0，由此可知L_{rw}為 semi-dininite。

(5). $\mathbf{L_{rw}}$最小 eigenvalue 為 0，且相對於 0 之 eigenvector 為 1。$\mathbf{L_{sym}}$最小的 eigenvalue 為 0，且相對於 0 之 eigenvector 為$\mathbf{D^{1/2} 1}$。

因為$\mathbf{L_{rw} 1 = D^{-1} L 1 = 0 = 0 \times 1}$且$\mathbf{L_{rw}}$為 semi-definite，所以 0 為$\mathbf{L_{rw}}$之最小的 eigenvalue。由(2)與(4)可知，0 也是$\mathbf{L_{sym}}$最小的 eigenvalue 且$\mathbf{L_{sym}}$相對於 0 之 eigenvector 為$\mathbf{D^{1/2} 1}$。

(6). $\mathbf{L_{sym}}$與$\mathbf{L_{rw}}$的所有 eigenvalue(n 個)皆為實數，且$\mathbf{0 \le \lambda_1 \le \lambda_2 \le \cdots \lambda_n}$。

2.2.3 RatioCut 與 Ncut

給定一個 similarity graph $\mathbf{G = (V, E)}$，為了將所有的點$\mathbf{V}$分成任意$\mathbf{k}$群，首先定義 cut 如下：

$$\mathbf{cut(A_1, \ldots, A_k)} = \frac{\mathbf{1}}{\mathbf{2}} \sum_{\mathbf{p=1}}^{\mathbf{k}} \mathbf{W(A_p, \overline{A_p})}$$

其中，$\mathbf{A_1 \cup \cdots \cup A_k = V}$，且$\mathbf{A_1 \cap \cdots \cap A_k = \emptyset}$。

Spectral clustering 的目標就是找出一組分割(partition) $\mathbf{A_1, \ldots, A_k}$，使得$\mathbf{cut(A_1, \ldots, A_k)}$為最小。但是在分群時，有時會希望分群完成之後每一個群的大小都差不多，所以必須將群的數量也考慮進去，於是就有 RatoiCut 與 Ncut。分別定義如下：

$$\mathbf{RatioCut(A_1, \ldots, A_k)} = \sum_{\mathbf{p=1}}^{\mathbf{k}} \frac{\mathbf{cut(A_p, \overline{A_p})}}{\mathbf{|A_p|}}$$

$$\mathbf{Ncut(A_1, \ldots, A_k)} = \sum_{\mathbf{p=1}}^{\mathbf{k}} \frac{\mathbf{cut(A_p, \overline{A_p})}}{\mathbf{vol(A_p)}}$$

其中，$\mathbf{|A|}$代表$\mathbf{A}$中點的數量，$\mathbf{vol(A)} = \sum_{\mathbf{i \in A}} \mathbf{d_i}$。

接下來就可以將重點放在如何找到 RatioCut 與 Ncut 的最小值上面。在這一小節中，只說明 k = 2 之原理，因為 k 為任意值時，其原理與 k = 2 時是一樣的。先說明 RatioCut：

根據之前的敘述，目標是要找出：

$$\min_{\mathbf{A \subset V}} \mathbf{RatioCut(A, \bar{A})}$$

首先定義向量$\mathbf{f} = (\mathbf{f_1}, \ldots, \mathbf{f_n}) \in \mathbb{R}^{\mathbf{n}}$為：

$$\mathbf{f_i} = \begin{cases} \sqrt{\mathbf{|\bar{A}|/|A|}}, & \mathbf{if\ v_i \in A} \\ -\sqrt{\mathbf{|\bar{A}|/|A|}}, & \mathbf{if\ v_i \in \bar{A}} \end{cases} \qquad (\mathbf{2.1})$$

則$\mathbf{f}$具有以下性質：

(1). $\mathbf{f \perp 1}$

(2). $\mathbf{\|f\| = \sqrt{n}}$

(3). $\mathbf{f^T L f = |V| \cdot RatioCut(A, \bar{A})}$

因為$\mathbf{|V|}$為定值，所以找出$\mathbf{RatioCut(A, \bar{A})}$的最小值相當於找出$\mathbf{f}$使得$\mathbf{f^T L f}$的值為最小，因此可以將問題重新表現如下：

$$\min_{A \subset V} \mathbf{f^T L f} \quad \textbf{subject to } \mathbf{f \perp 1} \text{ 且 } \mathbf{f_i} \text{定義於} (2.1)，\|\mathbf{f}\| = \sqrt{\mathbf{n}}$$

但是這是一個 NP hard 的問題，沒有辦法有效率的被解決，所以放寬一些限制，改成$\mathbf{f} \in \mathbb{R}^{\mathbf{n}}$皆可，如此可以將問題簡化成：

$$\min_{\mathbf{f} \in \mathbb{R}^{\mathbf{n}}} \mathbf{f^T L f} \quad \textbf{subject to } \mathbf{f \perp 1} \text{ ，} \|\mathbf{f}\| = \sqrt{\mathbf{n}}$$

由 Rayleigh-Ritz theorem[35]可知：

$$\boldsymbol{\lambda_1} \leq \frac{\mathbf{f^T L f}}{\mathbf{f^T f}} \leq \boldsymbol{\lambda_n}$$

且 f 若取 L 之 eigenvector 即可，但是前面的敘述可知，L 相對於$\boldsymbol{\lambda_1}$之 eigenvector 為 1，但是因為 f 被限制要垂直於 1，所以改取相對於$\boldsymbol{\lambda_2}$之 eigenvector。求得 f 後將 f 當成 indicator vector，就可以藉由 f 得到 A，也就是將 V 分成兩群：

$$\begin{cases} \mathbf{v_i \in A}, & \textbf{if } \mathbf{f_i \geq 0} \\ \mathbf{v_i \in \bar{A}}, & \textbf{if } \mathbf{f_i < 0} \end{cases} \qquad (2.2)$$

接下來說明如何求 Ncut 之最小值，其過程與求 RatioCut 之最小值非常類似。

首先定義向量$\mathbf{f} = (\mathbf{f_1}, \ldots, \mathbf{f_n}) \in \mathbb{R}^{\mathbf{n}}$為：

$$\mathbf{f_i} = \begin{cases} \sqrt{\dfrac{\mathbf{vol}(\bar{\mathbf{A}})}{\mathbf{vol}(\mathbf{A})}}, & \textbf{if } \mathbf{v_i \in A} \\ -\sqrt{\dfrac{\mathbf{vol}(\bar{\mathbf{A}})}{\mathbf{vol}(\mathbf{A})}}, & \textbf{if } \mathbf{v_i \in \bar{A}} \end{cases} \qquad (2.3)$$

則可求得：

(1). $\mathbf{(Df)^T 1 = 0}$

(2). $\mathbf{f^T D f = vol(V)}$

(3). $\mathbf{f^T L f = vol(V) Ncut(A, \bar{A})}$

所以可以將問題表示如下：

$$\min_{A \subset V} \mathbf{f^T L f} \quad \textbf{subject to } \mathbf{Df \perp 1} \text{ 且 } \mathbf{f_i} \text{定義於} (2.3)，\mathbf{f^T D f = vol(V)}$$

同樣可以放寬限制變成：

$$\min_{f \in \mathbb{R}^n} \mathbf{f^T L f} \quad \textbf{subject to } \mathbf{Df \perp 1} \text{ ，} \mathbf{f^T D f = vol(V)}$$

最後將$\mathbf{f = D^{-1/2} g}$代入：

$$\min_{g \in \mathbb{R}^n} \mathbf{g^T D^{-1/2} L D^{-1/2} g} \quad \textbf{subject to } \mathbf{g \perp D^{-1/2} 1} \text{ ，} \mathbf{\|g\|^2 = vol(V)}$$

其中，$\mathbf{D^{-1/2} L D^{-1/2} = L_{sym}}$，且如前所述$\mathbf{L_{sym}}$相對於$\boldsymbol{\lambda_1}$之eigenvector為$\mathbf{D^{-1/2} 1}$，所以根據Rayleigh-Ritz theorem，取g為$\mathbf{L_{sym}}$相對於$\boldsymbol{\lambda_2}$之eigenvector，則$\mathbf{f = D^{-1/2} g}$。最後如同(2.2)，可以利用f將V分成$\mathbf{A}$與$\bar{\mathbf{A}}$。

2.3 Co-cluster Algorithm

基於spectral clustering的架構，Dhillon[1]提供了一個spectral co-clustering的演算法，此演算法取左邊以及右邊第二個奇異值向量(singular vectors)適度的縮放文章及字的矩陣而使其產生更好的二元分割(bipartitionings)。co-clustering演算法將使用一個二分的無向圖模組，一個點的集合包含了兩個獨立的集合，其中一者代表文章，另外則代表字，將文章以及字進行co-clustering必須依照以下準則：一群已經分割好的文章應該促使著一群字的分群，同樣的當一群字分割好也應意味著一群文章的分群。

傳統上，spectral clustering 轉移了分群的問題到解圖的最小分割(minimum cut)問題，在圖型的分割，如果分群的演算法沒有考慮到群的大小，不平均的問題就有可能會發生，有一種方法去避免此問題就是限定群的大小，因此不同的 spectral clustering 演算法的發展也會根據它不同的限制，如上面 2.2.2 小節所述，RatioCut[36]是根據點的數量進行計算，同樣的 Ncut[37]則會考慮邊的權重，適當的放寬限制後 spectral clustering 演算法可變成解特徵方程式(eigenvalue equation)的問題，例如 Ncut 可以形成一個廣益的特徵值問題 $\mathbf{Lz} = \boldsymbol{\lambda}\mathbf{Dz}$, 它提供了一種真正的放寬至離散優化問題去尋找最小且正規劃的切集。

Dhillon 建議將文章以及字一起進行處理，所以 Laplician matrix 以及對角矩陣則採用方塊矩陣來一併考慮文章以及字，式子(2.4)將呈現它們的不同處。

$$\mathbf{L} = \begin{bmatrix} \mathbf{D_1} & -\mathbf{A} \\ -\mathbf{A^T} & \mathbf{D_2} \end{bmatrix}, \mathbf{D} = \begin{bmatrix} \mathbf{D_1} & \mathbf{0} \\ \mathbf{0} & \mathbf{D_2} \end{bmatrix} \tag{2.4}$$

$\mathbf{Lz} = \boldsymbol{\lambda}\mathbf{Dz}$可以被改寫為式子(2.5)

$$\begin{bmatrix} \mathbf{D_1} & -\mathbf{A} \\ -\mathbf{A^T} & \mathbf{D_2} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} = \boldsymbol{\lambda} \begin{bmatrix} \mathbf{D_1} & \mathbf{0} \\ \mathbf{0} & \mathbf{D_2} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \tag{2.5}$$

以上兩個式子可以被進一步的改寫成兩個子式子並且正規劃後的矩陣 $\mathbf{A_n} = \mathbf{D_1}^{-1/2}\mathbf{A}\mathbf{D_2}^{-1/2}$正是奇異值分解(singular value decomposition)的定義，對於分兩群而言，$\mathbf{A_n}$的左邊第二以及右邊第二奇異值向量將會產生良好的分割，對於多群的問題，k-way 分割可以被應用到矩陣$\mathbf{A_n}$；首先先完成$\mathbf{A_n}$左邊第$\boldsymbol{\ell}$個以及右邊第$\boldsymbol{\ell}$個奇異值，左邊的奇異值向量為$\mathbf{u_1}, \ldots, \mathbf{u_{\ell+1}}$，右邊的奇異值向量為$\mathbf{v_1}, \ldots, \mathbf{v_{\ell+1}}$，而$\mathbf{u_i}$以及$\mathbf{v_i}$代表矩陣$\mathbf{A_n}$第左以及第右第 i 大的奇異值所對應的奇異值向量，第二步將$\mathbf{u_1}, \ldots, \mathbf{u_{\ell+1}}$組成矩陣 U，以及$\mathbf{v_1}, \ldots, \mathbf{v_{\ell+1}}$**組成**$\mathbf{V}$，第三部將資料及以$\boldsymbol{\ell}$維度下呈現式子(2.6)，最後對$\boldsymbol{\ell}$維度下資料的 Z 執行 k-means 演算法並獲

得 k-way 的多群結果，演算法 1 清楚的呈現 co-clustering[1]的演算法。

$$\mathbf{Z} = \begin{bmatrix} \mathbf{D_1}^{-1/2}\mathbf{U} \\ \mathbf{D_2}^{-1/2}\mathbf{V} \end{bmatrix} \qquad (2.6)$$

======== Algorithm 1. Co − clustering Algorithm ==========

Intput：Given matrix A and the number of cluster K

Output：Clustering index I and ℓ − dimension data Z

1. begin

2. $\mathbf{A_n} \leftarrow \mathbf{D_1}^{-\frac{1}{2}}\mathbf{A}\mathbf{D_2}^{-\frac{1}{2}}$

3. $\ell \leftarrow [\log_2 \mathbf{K}]$

4. Compute ℓ left and right singular vectors of $\mathbf{A_n}$. Left singular vectors are $\mathbf{u_1}, \ldots, \mathbf{u_{\ell+1}}$ and right singular vectors are $\mathbf{v_1}, \ldots, \mathbf{v_{\ell+1}}$, where $\mathbf{u_i}$ and $\mathbf{v_i}$ represents the left and right singular vectors corresponding to ith largest singular values of $\mathbf{A_n}$.

5. $\mathbf{U} \leftarrow [\mathbf{u_1}, \ldots, \mathbf{u_{\ell+1}}]$

6. $\mathbf{V} \leftarrow [\mathbf{v_1}, \ldots, \mathbf{v_{\ell+1}}]$

7. Form the matrix Z as shown in Equation(2.6)

8. $\mathbf{I} \leftarrow \mathbf{k-means}(\mathbf{Z}, \mathbf{K})$

9. return I, Z

10. end

第三章、系統設計

3.1 概念

傳統的分群的方法例如 K-means、pLSA[38]往往只考慮到單一維度的關係來分群，例如在文件分群問題上，只是利用文章裡面的字計算群與群之間的關係性，但是卻沒有考慮到字跟所在的文章之間具有什麼樣的關係，本論文採用 co-clustering 的方法來做文章分群，同時考慮文章與所含之文字以及文字與所在的文件關係同時分群;co-clustering 背後是透過 spectral clustering 分群，在多群分群上，spectral clustering 可以使用 spectral embeddeding 技術做類似降維之動作；假設當文章被降維到三度空間的情況下，我們可以從三維座標圖中了解到一篇文章在這個三維的空間所在的位置以及群中心的位置;同樣的，基於 co-cluster 演算法的架構，我們也可以知道每個字在這三度空間的位置，這點提供了我們很大的想法來對 co-cluster 的演算法做更進一步的改善。

我們知道群中心往往代表一群裡面最能表現一個群，換句話說當我們知道一個點如果靠近群中心，我們可以間接的了解到這個點應該具備足夠的特徵可以讓它表現這個群，同理，如果一個點距離群中心很遠，這個資訊告訴我們它不具足夠的特徵可以表現這個群，更進一步的推廣，以兩群為例，當我們知道如果一個點靠近某個群中心，相對的遠離另外一個群中心的話，我們可以知道這個點對於靠近的那個群而言是相對的重要，反之如果一個點與某兩個群都很靠近，也就是說他可能介於兩群的中間，我們可以推得這個點並不是一個具備足夠特徵可以將兩群分開的點，所以具備這樣條件的點，我們可以把它想成是一種雜訊，比方說當我們要做新聞分群的時候，”新聞”這樣的一個關鍵詞，出現次數很多，但是他往往不會給我們帶來很大的幫助，因為它可能出現在每一篇文章裡面。

有鑒於上述的種種想法，本論文希望透過一個幾何的距離概念，給予字適當的權重，經由一個加權的方法讓 co-cluster 這個演算法的效能有更進一步的提升。

3.2 Outlier detection

3.2.1 目的

本節我們討論使用 outlier detection 的目的，首先我們先探討分群的意義，一個分群的演算法將一些相似的文章分成一群，我們可以將文章所代表的座標點，例如根據 k-means 的演算法分群到所指定的群裡，首先我們根據 co-cluster[1] 的演算法先將維度降到三維的空間，將所有的點標上去如圖 3-1，這樣的方法可以將每篇文章與字利用視覺化的技術將每篇文章與字顯示在三維的空間。最後利用 k-means 做分群得到的結果如圖 3-2，最後我們實際觀察正確答案將原本文章的標籤標記上去，利用顏色的不同區隔出兩群之間的差異得到圖 3-3，可以了解到利用 k-means 的演算法得到的結果並不理想，紅色群幾乎一半的點都被分到綠色的群。

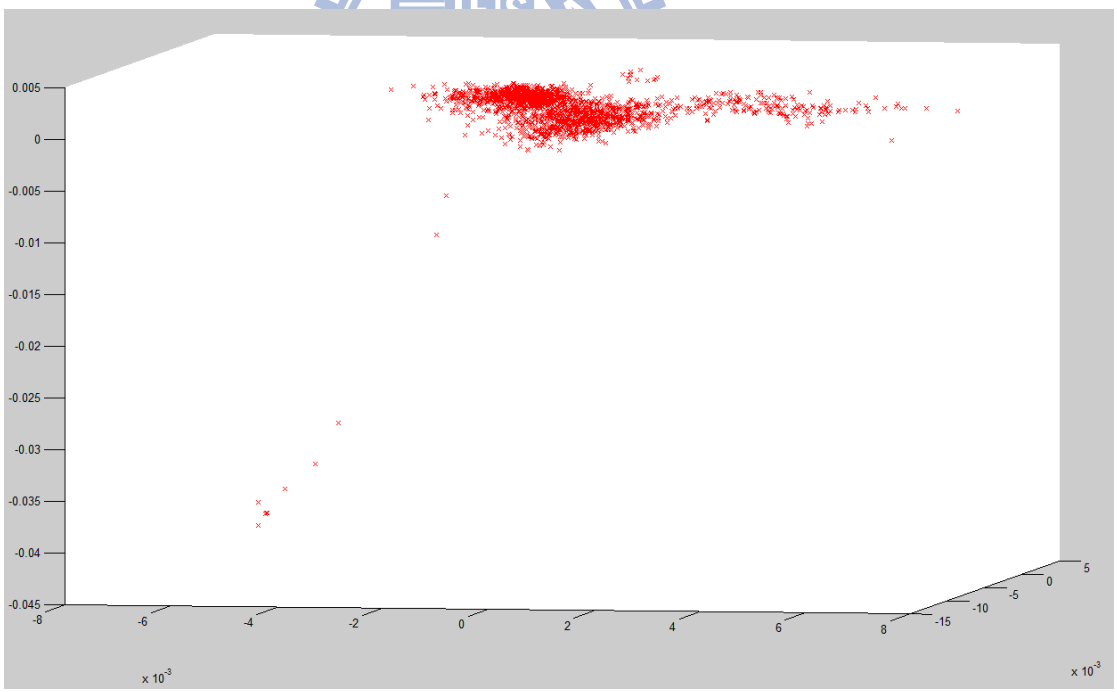


圖3-1 用co-cluster降維後點圖

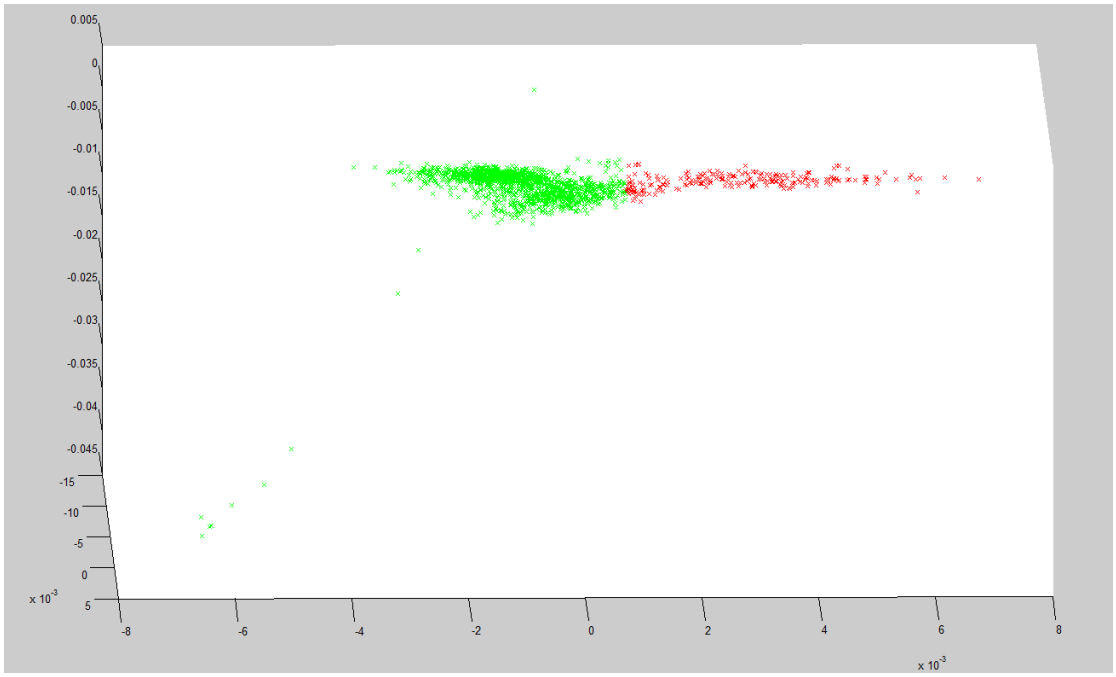


圖3-2 用co-cluster降維後點圖利用k-means分群的結果

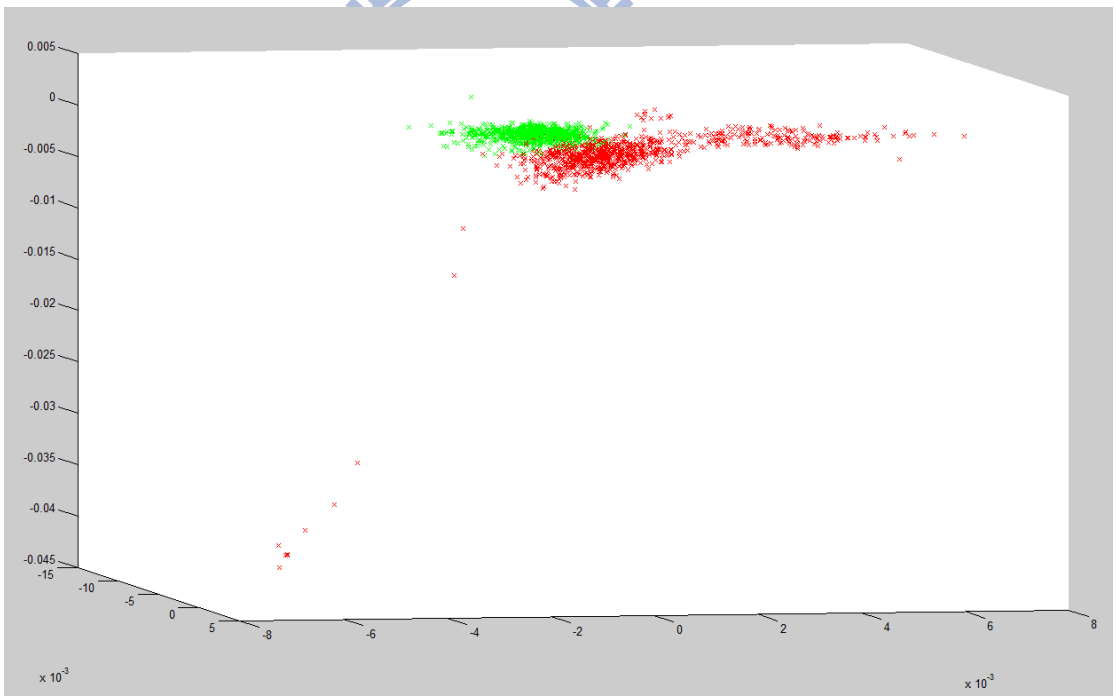


圖3-3 用co-cluster降維後點圖實際的label

基於以上的因素，我們試著將 outlier detection 的概念帶入本系統，首先我們先利用 outlier detection 找出屬於 outlier 的點並標記如圖 3-4，綠色點標定為 outlier，我們觀察紅色點可以發現大部分的文章都著落在不是屬於

outlier 的區塊，也就是說這區塊的文章的相似度相較於 outlier 的文章是比較接近的。有鑒於此，去除 outlier 除了可以取出大部份的文章以外，點與點之間的分佈也相較於沒有使用 outlier 還要均勻，進而分析紅色區塊的點我們將這些點對應回原本的文章篇數，可以找到哪些文章是不屬於 outlier 我們將此定為 inner document。

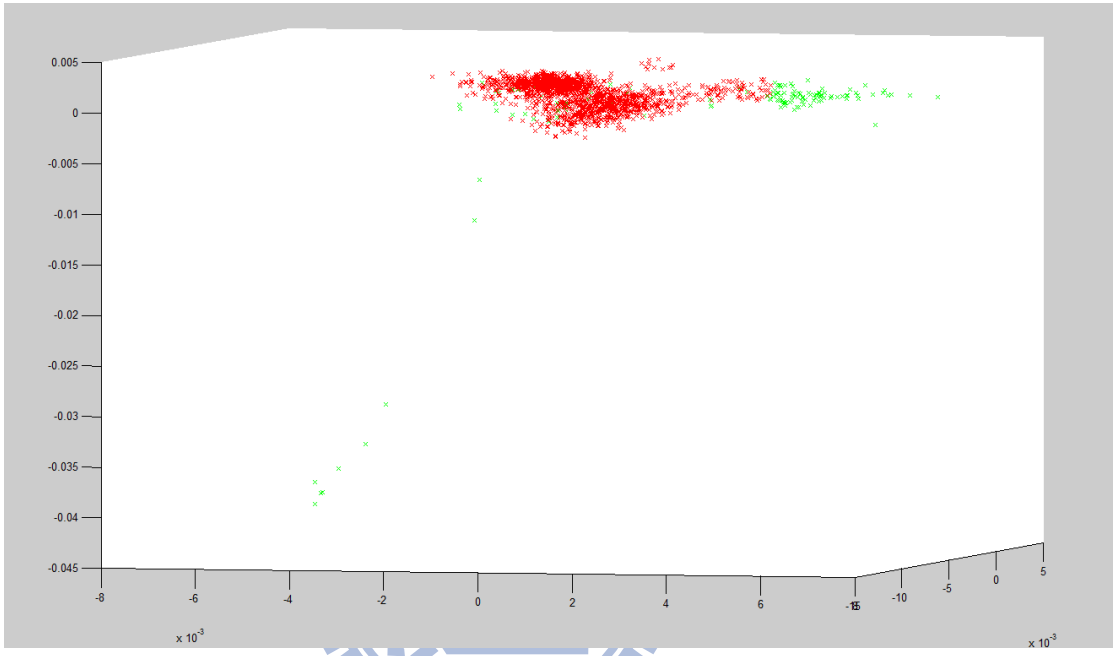


圖3-4 做outlier detection後的點圖，綠色為outlier

3.2.2 節我們將說明 outlier detection function 的符號定義，藉由計算 covariance matrix 以及利用馬氏距離(Mahalanbios distance)的公式將所有的點算出其對應的馬氏距離後，對其分佈我們取平均值加減 t 個標準差的範圍當作 inner document，反之在外的即為 outlier document。

3.2.2 符號定義

此小節我們針對 Outlier detection 內的符號做一個完善的定義，給定一個矩陣$\mathbf{A_{m\times n}}$其中 m 代表總共有 m 篇文章，n 代表總共有 n 個字，故一個數$\mathbf{A_{ij}}$代表第 i 篇文章的第 j 個字，其中$\mathbf{i}\in\{\mathbf{1}\ldots\ldots\mathbf{m}\}$，$\mathbf{j}\in\{\mathbf{1}\ldots\ldots\mathbf{n}\}$，經過 co-cluster[1]的方法以後，我們可以得到一矩陣 Z 代表著 m 篇文章字在 d 個維度下的座標點，我們令一矩陣$\mathbf{dp(i,j)}=\left\{\mathbf{Z(i,j)}\middle|\mathbf{i}\in\{\mathbf{1}\ldots\ldots\mathbf{m}\},\mathbf{j}\in\{\mathbf{1}\ldots\ldots\mathbf{L}\}\right\}$，故$\mathbf{dp_i}$代表第 i 篇文章在維度 d 下的點座標，又$\overline{\mathbf{dp}}_\mathbf{m}$代表 m 篇文章的期望值(平均數)，$\mathbf{V_m}$代表矩陣 dp 的 covariance matrix，然而 covariance matrix 的算式可以寫成如下：

$$\mathbf{V_m}=\frac{\mathbf{1}}{\mathbf{m-1}}\sum_{\mathbf{i=1}}^{\mathbf{m}}\left(\mathbf{dp_i}-\overline{\mathbf{dp}}_\mathbf{m}\right)\left(\mathbf{dp_i}-\overline{\mathbf{dp}}_\mathbf{m}\right)^{\mathbf{T}} \qquad (\mathbf{3.1})$$

接著利用 covariance matrix 我們可以計算馬氏距離(Mahalanbios distance)$\mathbf{M_i}$，其中$\mathbf{i}\in\{\mathbf{1}\ldots\ldots\mathbf{m}\}$，馬氏距離的計算公式如下：

$$\mathbf{M_i}=\left(\left(\mathbf{dp_i}-\overline{\mathbf{dp}}_\mathbf{m}\right)^{\mathbf{T}}\mathbf{V_m}^{-\mathbf{1}}\left(\mathbf{dp_i}-\overline{\mathbf{dp}}_\mathbf{m}\right)\right)^{\mathbf{1/2}} \qquad (\mathbf{3.2})$$

我們取得每個點的馬氏距離以後，計算標準差$\boldsymbol{\sigma}=\sqrt{\frac{\mathbf{1}}{\mathbf{m}}\sum_{\mathbf{i=1}}^{\mathbf{m}}(\mathbf{M_i}-\bar{\mathbf{M}})^{\mathbf{2}}}$，取$\mathbf{M_i}$在與平均值相差 t 個標準的距離外為 outlier 定其為 outlierdoc 反之在與平均值相差 t 個標準差內的點定其為 innerdoc，其中 t 是一個參數讓用來調整須要幾個標準差。

如果我們將所有在 Z 上的點表示在三維的空間中，可以發現點跟點的分布在三維的空間比較可以顯示有 outlier 的點與不是 outlier 的點，為了將所有的點表現在三度空間下，在此我們定$\mathbf{d}=\mathbf{3}$來取 outlier 的點。

3.2.2 演算法

根據上一小節的符號定義，本節將寫出完整的實驗演算法。

======== Algorithm 2. function Outlier detection ==========

input Give document matrix $D = \{dp_1, \dots dp_m\}$, where $dp_i \in \mathbb{R}^d$ and the number of standard deviation t

output innerdoc，outlierdoc

Given $D = \{dp_1, \dots, dp_m\}$

1. $\overline{dp_m}$ = mean of D

2. $V_m = \frac{1}{m-1}\sum_{i=1}^{m}(dp_i - \overline{dp_m})(dp_i - \overline{dp_m})^T$

3. for i = 1 to m

4. $M_i = \left((dp_i - \overline{dp_m})^T {V_m}^{-1}(dp_i - \overline{dp_m})\right)^{\frac{1}{2}}$

5. end

6. $\bar{M}$ = mean of M ($M = \{M_1, \dots M_m\}$)

7. for i = 1 to m

8. if $\left((\bar{M} - t\sigma) \le M_i \le (\bar{M} + t\sigma)\right)$

9. innerdoc $\leftarrow M_i$

10. end

11. if $\left((\bar{M} + t\sigma) < M_i || (\bar{M} - t\sigma) > M_i\right)$

12. outlierdoc $\leftarrow M_i$

13. end

14. end

在 outlier detection 的演算法中，參數 t 是用以讓我們調整須要多少個標

準差，我們實驗不同的標準差來評估最後實驗的效果的好壞，我們選擇了 talk.politics.guns 以及 talk.politics.mideast 這兩筆資料分群，圖 3-5 顯示出結果。當 t=1 時紅色的點代表著 inner document 的部份，約略有 7%的資料被我們分到 outlier，當 t=2 時紅色與綠色的點代表 inner document 的部份，約略有 1.7%得資料被我們分到 outlier，當 t=3 時紅色、綠色以及藍色的點代表 inner document 的部份，約略有 0.9%的點被分成 outlier，當我們選定的資料非常相似，例如都在 politics 分類下的文章，大部分的點會相當的集中，如果選定的 t 值太過於大，被定義到 outlier 的點將會非常少，我們也將 t 在不同值的結果列出如表 3-1。

	F1-value
t=1	93.21%
t=2	63.21%
t=3	63%

表 3-1 在 outlier detection 中不同參數 t 所分群的實驗結果。

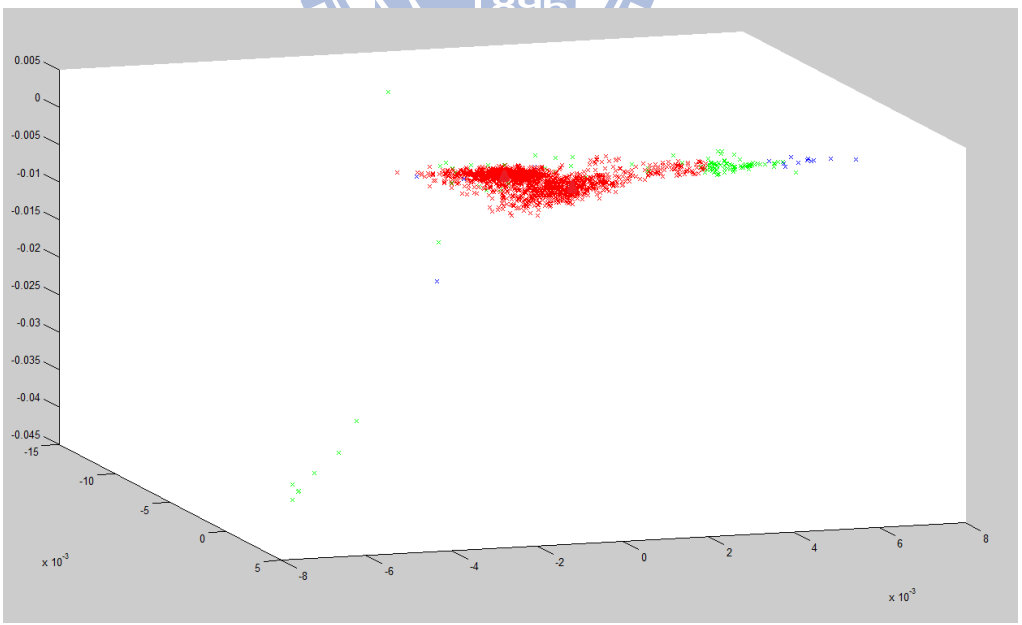


圖3-5 talk.politics.guns以及talk.politics.mideast執行outlier detection的結果

3.2.2 實驗效果分析

首先我們選取 20newsgroups 的 corpus 之中的 talk.politics.guns 以及 talk.politics.mideast 這兩個資料做分析，經過去除 outlier 後我們針對 inner document 我們將其在進行一次 co-cluster 最後利用 k-means 做分群，得到的結果如圖 3-6，最後比較實際上帶有標籤的點圖 3-7，得到的結果可讓正確率大大的提升，因為我們將原本分佈比較散亂的圖，經過 outlier detection 以後得到的點較原本均勻，此時做 k-means 較不會因為原本點圖的分散而找到不恰當的群中心，當找到恰當的群中心以後，分群的效果自然而然會有顯著的提升，但是針對傳統的 data mining 方法，我們會將 outlier 的點做刪除的動作，由於本實驗找出的 outlier 並不是真的雜訊，故我們利用 3.4 節的 merge function 方法將屬於 outlier 的文章找到其最適合的群，並將他們分到距離最近的群，最後的結果作為評估最終的效能。

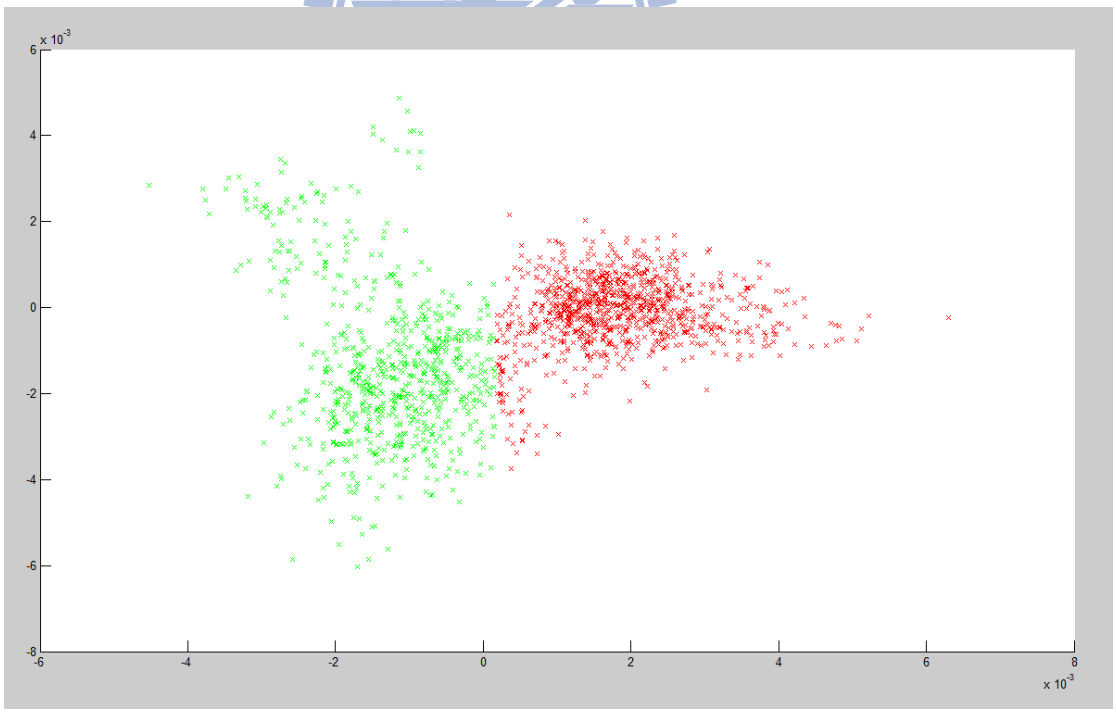


圖3-6 內點利用co-cluster分群的結果

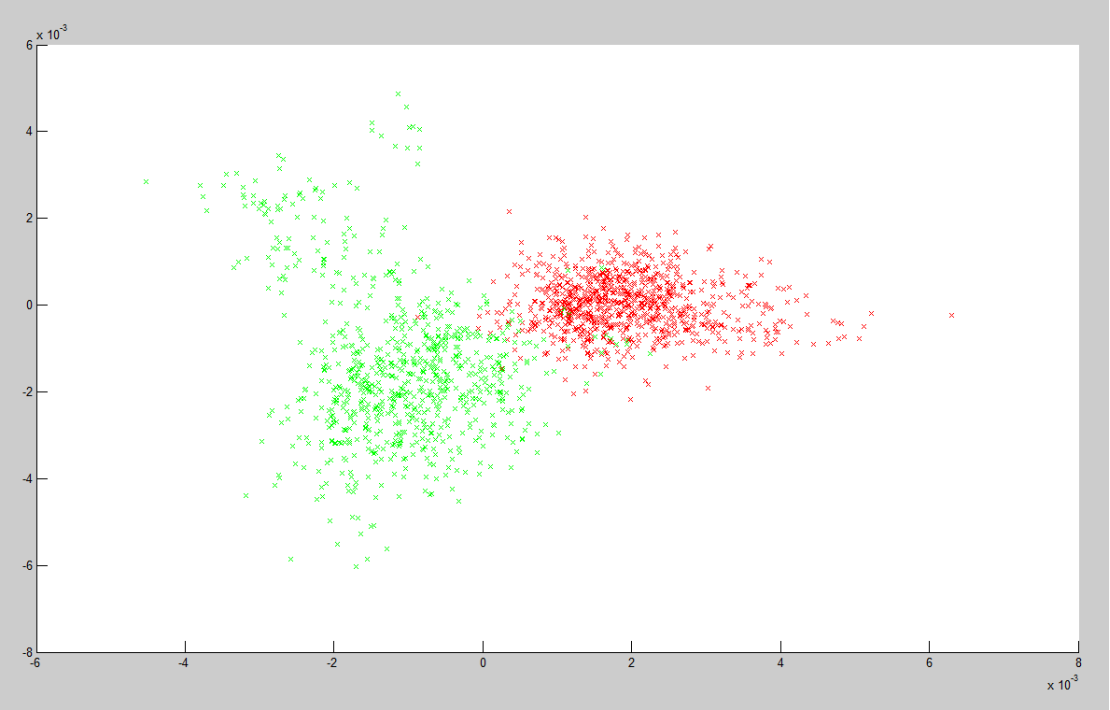


圖3-7 內點實際的label

3.3 Weighted Term Matrix Construction Algorithm

3.3.1 目的與概念

分群的概念架構在群與群之間的相似度，很值觀的我們可以認為假如群跟群之間用字的差異性較大，這樣的分群效果應當是好的，比方說以兩群為例子如果文章 1、3、5 裡面的字都是 w1、w3、w5 然而文章 2、4、6 裡面的字都是 w2 、w4、w6 那這樣子可以很清楚的將文章 1、3、5 分成一群，反之文章 2、4、6 可以分成另一群，但現實生活中文章內容的字往往夾帶著許多的常用字(stop word)，除此之外也有許多不是屬於 stop word 但依舊沒有鑑別度的字，比方說我們探討新聞類的文章，”新聞”這個字眼往往會出現在許多文章中，然而這個字眼並沒有辦法告訴我們是哪一類的新聞，這樣沒有鑑別度的字其實在每一篇文章中占很大的篇幅。

為了改善以上的問題，首先我們先探討矩陣的類型，給定一個矩陣$\mathbf{A}_{m\times n}$其

中 m 代表總共有 m 篇文章，n 代表總共有 n 個字，故每個數$\mathbf{A_{ij}}$代表第 i 篇文章的第 j 個字，其中裡面的值只考慮出現不出現也就是說指只能填 0 代表不出現 1 代表出現，我們通稱為 0/1 矩陣，以上的方法在語料庫給定以後這樣子的矩陣隨即可以產生但卻無法做任何的調整，以下我們的方法可以逐步的調整每個字的權重，透過一個圖型與分群的概念，我們將設計一個調整權重的方法，如此一來便可以藉調整字的權重來改進實驗的效果。

Co-cluster[1]的好處在於我們可以同時將文章與字做分群，此演算法的架構在於利用文章來帶動字的分群，亦可利用字來帶動文章的分群，我們的想法在於將字賦予權重之後將原本的頻率矩陣改變成帶有權重意義的頻率矩陣，透過這樣子的方式我們可以分析以及改變每個字的權重，最後在 3.5 節我們會說明如何動態的改變字的權重以及如何評估它的效能。

3.3.2 公式定義

本小節將會對公式以及我們設計的方法做更完整的解說，在不失一般性的條件下，我們先取兩群來說明首先我們給定兩個群中心$\mathbf{C_{center_1}}$以及$\mathbf{C_{center_2}}$，並假設有兩個點$\mathbf{wp_i}$以及$\mathbf{wp_i}$，$\{\mathbf{i}, \mathbf{j}\} \in \{\mathbf{1} \dots \dots \mathbf{n}\}$，如果說$\mathbf{wp_i}$比較靠近$\mathbf{C_{center_1}}$則依據分群的概念，$\mathbf{wp_i}$會分到$\mathbf{Cluster_1}$。如果說存在一點$\mathbf{wp}$與$\mathbf{Cluster_1}$以及$\mathbf{Cluster_2}$都相似，則點座標將會靠近$\mathbf{C_{center_1}}$以及$\mathbf{C_{center_2}}$，換言之這個點將會介於$\mathbf{C_{center_1}}$以及$\mathbf{C_{center_2}}$之間，因此我們定義一個計算距離的差距來衡量彼此之間的相似度如下列公式：

$$\Delta\mathbf{dist_i} = \left| |\mathbf{wp_i} - \mathbf{C_{center_1}}| - |\mathbf{wp_i} - \mathbf{C_{center_2}}| \right| \qquad (\mathbf{3.3})$$

透過這個公式我們可以了解假設兩點$\mathbf{wp_i}$以及$\mathbf{wp_j}$，兩群中心$\mathbf{C_{center_1}}$以及$\mathbf{C_{center_2}}$如圖 3-8，根據公式(3.3)計算，得$\Delta\mathbf{dist_i} = |\mathbf{4} - \mathbf{8}| =$

$\mathbf{4}$，$\Delta\mathbf{dist_j} = |\mathbf{2}-\mathbf{3}| = \mathbf{1}$，$\Delta\mathbf{dist_i} > \Delta\mathbf{dist_j}$，並且可以發現當數字越大代表此點較靠近某群，相對的如果值越小代表這點離$\mathbf{C_{center_1}}$以及$\mathbf{C_{center_2}}$都相當的靠近，因此我們認為值的大小與重要性成正比。

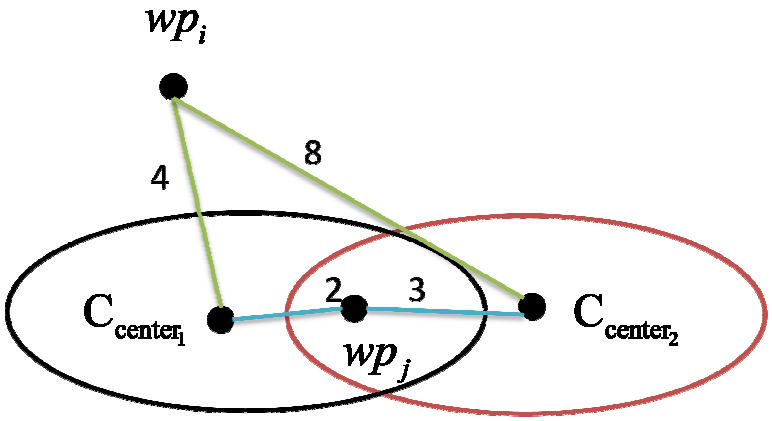


圖3-8 探討點與群中心的距離差所產生的關係性

為了避免$\Delta\mathbf{dist}$的值可能很散亂，在此我們透過 logistic function 將原先的值轉到{0,1}區間，logistic function 的圖如 3-9，我們可以發現當$\Delta\mathbf{dist}$越大時會越靠近 1，根據曲度的斜率我們可以控制當$\Delta\mathbf{dist}$的值越大(越小)我們該有多慢(多快)速度的增長，在此我們採用最原始型態的 logistic function 如下式子：

$$\mathbf{w_i} = \frac{\mathbf{1}}{\mathbf{1} + \mathbf{e}^{-\Delta\mathbf{dist_i}}} \qquad (\mathbf{3.4})$$

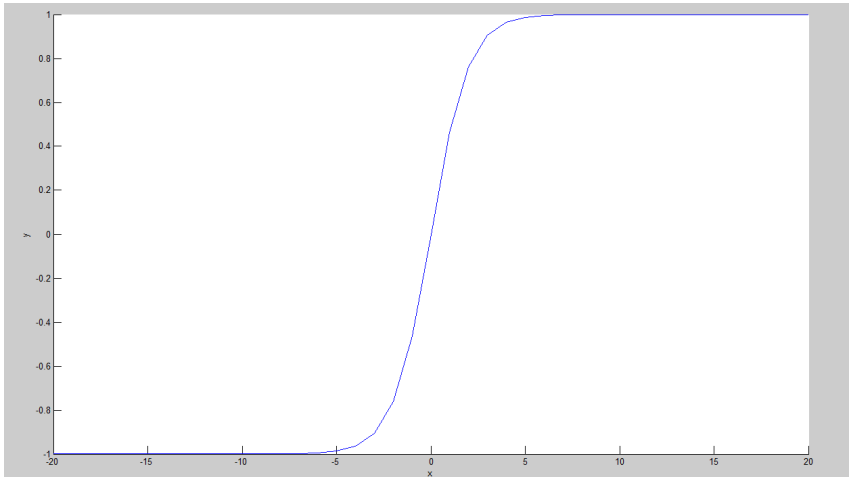


圖3-9 logistic function示意圖

由式子(3.3)以及(3.4)計算出來的$\mathbf{w_i}$即代表$\mathbf{wp_i}$所應有的權重，在此我們將進一步的推廣到多群，在分群的效能評估方式常常用兩兩比較的關係性來判斷分群的效能，F1 value[39]方法採用此種概念，在此我們效仿分群的評估方式，採用兩兩的計算，因此根據公式(3.3)、(3.4)我們可以改寫如下，其中 times 代表所執行的次數：

$$\Delta\overline{\mathbf{dist_i}} = \left(\sum_{\mathbf{r=1}}^{\mathbf{k}}\sum_{\mathbf{s=r+1}}^{\mathbf{k}}\left|\left|\mathbf{wp_i} - \mathbf{C_{center_r}}\right| - \left|\mathbf{wp_i} - \mathbf{C_{center_s}}\right|\right|\right) \Big/ \mathbf{times} \quad (\mathbf{3.5})$$

其中 k 代表群數$\{r, s\} \in \{\mathbf{1} \ldots\ldots \mathbf{k}\}$，$\mathbf{i} \in \{\mathbf{1} \ldots\ldots \mathbf{n}\}$，而 times 是為了平均所有的$\Delta\mathbf{dist_i}$所以最後會除以總共的次數，times 的計算方式為：$\mathbf{times} = \sum_{\mathbf{r=1}}^{\mathbf{k}}\sum_{\mathbf{s=r+1}}^{\mathbf{k}}\mathbf{1}$，最後我們將得到的$\Delta\overline{\mathbf{dist_i}}$透過公式(3.4)可以改寫成下式子：

$$\mathbf{w_i} = \frac{\mathbf{1}}{\mathbf{1} + \mathbf{e}^{-\Delta\overline{\mathbf{dist_i}}}} \quad (\mathbf{3.6})$$

最後我們令一個對角矩陣 T，使其值為$\mathbf{T(j,j)} = \mathbf{w_j}$，$\mathbf{j} \in \{\mathbf{1} \ldots\ldots \mathbf{n}\}$，假設一個矩陣$\mathbf{A_{m\times n}}$，**其中$\mathbf{A_{ij}}$代表裡面的值**，$\mathbf{A} \times \mathbf{T}$代表對於文章 i 其所對應第 j 個字乘以它所對應的權重$\mathbf{w_j}$，所得到的矩陣是賦有權重的一個新的矩陣，此矩陣包含了新的文章與字的關係，透過這個關係我們依舊可以再進一步做co-cluster的分群，再透過 Weighted Term Matrix Construction Algorithm 的評估，我們可以在得到更新一層的關係，在 3.5 節我們將會做更詳細的說明。

3.3.3 演算法

給定 n 個 word point $\mathbf{wp_i}$，$\mathbf{i} \in \{\mathbf{1} \ldots\ldots \mathbf{n}\}$以及 k 的群中心$\mathbf{C_{center_1}} \ldots\ldots \mathbf{C_{center_k}}$，

最後輸出是一個對角矩陣 T，根據式子(3.5)、(3.6)我們寫出本實驗的演算法如下：

=== Algorithm 3. Weighted Term Matrix Construction Algorithm ===

intput wp_i，$C_{center_1} \ldots\ldots C_{center_k}$，**k is the number of cluster**

output T

1. initial T **is a zero − matrix with dimention** $n \times n$

2. times $= \sum_{r=1}^{k} \sum_{s=r+1}^{k} 1$

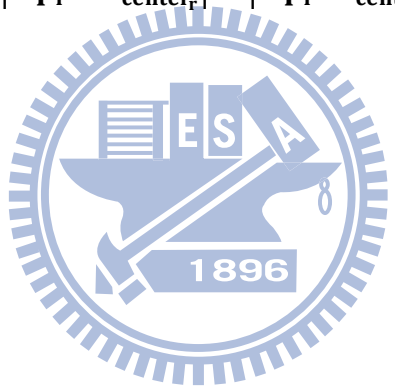
3. for $i = 1$ **to** n

4. $\overline{\Delta dist_i} = \left(\sum_{r=1}^{k} \sum_{s=r+1}^{k} \left| \left| wp_i - C_{center_r} \right| - \left| wp_i - C_{center_s} \right| \right| \right) / times$

5. $T(i,i) = \frac{1}{1 + e^{-\overline{\Delta dist_i}}}$

6. end

7. return T



3.4 Merge function

3.4.1 定義及解說

在 3.2 節我們有提到，雖然我們用了 outlier detection 把 outlier 過濾後，保留剩下的文章再分群可以達到還不錯的效果，但是面對 outlier 的文章，我們不能夠將它刪除，這也是有別於先前的 data mining[14]的傳統的方法，給定一個矩陣$A_{m \times n}$做 co-cluster[1]之後我們將會得到一個降維後的座標系統 $Z_{(m+n) \times L}$，其中 Z 的前$m \times L$的維度代表作標點，後面$n \times L$，L 代表每個文章或是字的維度，這個座標系統 Z 我們為了之後會用到先把它叫做Z_{old}，透過 outlier detection 以後我們取得 innerdoc 進行 co-cluster 之後我們會得到一個新的座

標系統 Z，對 Z 做 k-means 之後每個 innerdoc 皆會有自己的 index，代表該點標定為哪個群，此 index 我們叫做$\mathbf{index_{new}}$，代表去除 outlier 後新的矩陣進行 co-cluster 後得到分群的 index。

當我們去除 outlier 後得到新的矩陣進行 co-cluster 得到的座標系統 Z 與$\mathbf{Z_{old}}$是兩個不同的座標系統，但我們可以確定$\mathbf{innerdoc \subseteq alldoc}$，alldoc 代表原本語料庫的所有文章，根據上面的條件，我們可以對每個 innerdoc 找到其在原本的座標系統$\mathbf{Z_{old}}$中對應的座標點，所以每個 innerdoc 都會有根據去除 outlier 後進行 co-cluster 之新座標系統所產生的$\mathbf{index_{new}}$以及在尚未去除 outlier 之舊座標系統的$\mathbf{Z_{old}}$，但是屬於 outlierdoc 的文章只保留在舊座標系統所得到的 index 以及座標$\mathbf{Z_{old}}$，根據 3.1 節我們得知舊座標系統往往會因為 outlier 的影響造成分群效果不佳，故在舊座標系統得到的 index 是錯誤率較高的，但是如何讓 outlierdoc 的文章產生較正確的 index，我們必須由 innerdoc 的座標點來修正 outlierdoc 的 index。

該如何利用 innerdoc 的座標點來修正 outlierdoc 的 index 首先必須計算在$\mathbf{index_{new}}$之中哪些點是屬於第 k 群，將之歸類好以後再去找其在$\mathbf{Z_{old}}$所對應的座標點，並且計算群中心，在此我們先舉個例子來做解釋，假設存在 10 個點$\mathbf{D_i}$，其所對應的 index 為$\mathbf{idx_i}$，$\mathbf{i \in \{1 \ldots\ldots 10\}}$，尚未進行 outlier detection 以及其實際的 label 其值如下表：

文章	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9	D_{10}
Index	1	1	1	1	1	1	1	2	2	2
實際	1	1	1	1	2	2	2	2	2	2

表 3-2 未做 outlier detection 的文章所對應的 index

其所對應的座標點分佈圖如下：

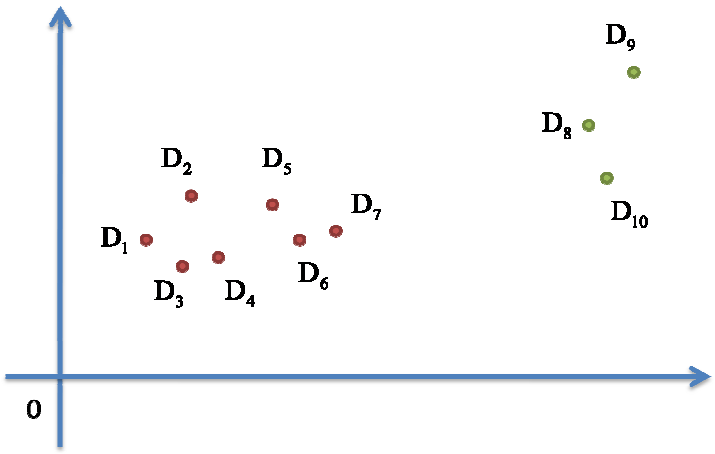


圖3-10 各點尚未進行outlier detection之分佈圖

透過去除 outlier 之後進行 co-cluster，我們成功的將正確率提高以及找出 outlier 的點其值以及分佈圖如下：

文章	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9	D_{10}
$Index_{new}$	1	1	1	1	2	2	2	?	?	?
實際	1	1	1	1	2	2	2	2	2	2

表 3-3 做完 outlier detection 再進行 co-cluster 後的文章所對應的 index

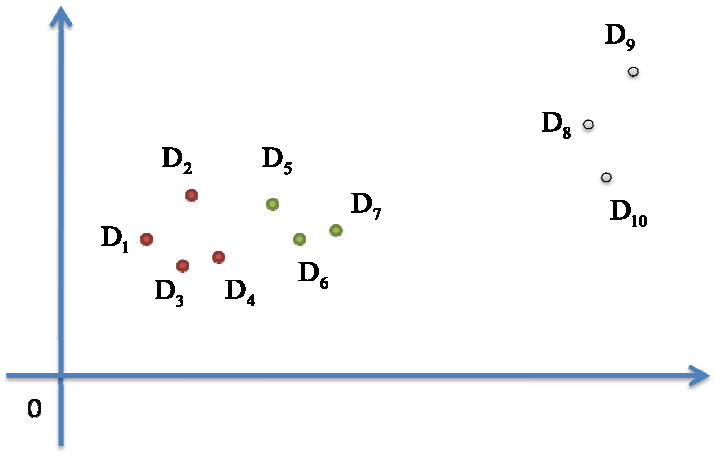


圖3-11 各點進行outlier detection再進行co-cluster後之分佈圖

對於D_8、D_9、D_{10}這三個點由於其 index 錯誤率較高，所以我們取出

$\mathbf{Index_{new}}$之中 label = 1 的點算出其幾何平均得到群中心$\mathbf{C_{center_1}}$，同樣的我們也對 label = 2 的點計算幾何平均得群中心$\mathbf{C_{center_2}}$，必須注意的一點是我們在此的座標系統都是原先尚未進行 outlier detection 的座標系統，因為我們必須在與$\mathbf{D_8}$、$\mathbf{D_9}$、$\mathbf{D_{10}}$存在座標點的系統下討論相似度，在將$\mathbf{D_8}$、$\mathbf{D_9}$、$\mathbf{D_{10}}$的座標分別與$\mathbf{C_{center_1}}$以及$\mathbf{C_{center_2}}$兩群中心進行相似度的計算，在此我們取歐式距離的計算方式，最後取最小者當作最相似者並且標定與之相同的 label 如圖 3-12，最後的值如下表：

文章	$\mathbf{D_1}$	$\mathbf{D_2}$	$\mathbf{D_3}$	$\mathbf{D_4}$	$\mathbf{D_5}$	$\mathbf{D_6}$	$\mathbf{D_7}$	$\mathbf{D_8}$	$\mathbf{D_9}$	$\mathbf{D_{10}}$
$\mathbf{Index_{new}}$	1	1	1	1	2	2	2	2	2	2
實際	1	1	1	1	2	2	2	2	2	2

表 3-4 進行相似度計算後的文章所對應的 index

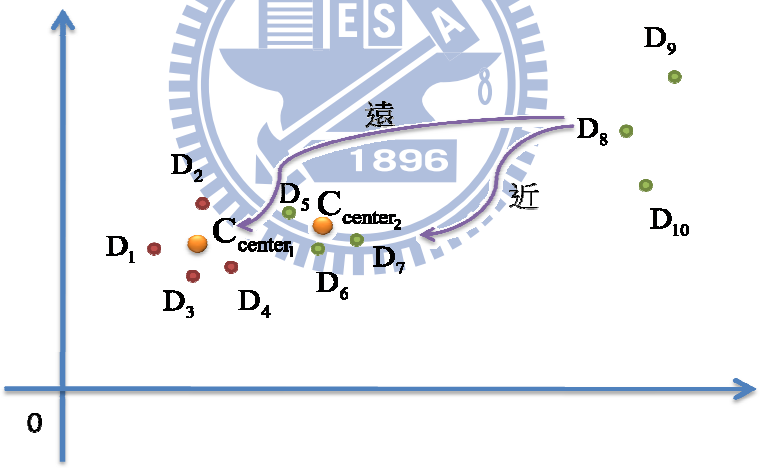


圖3-12 進行相似度計算後的各點之分佈圖

整個 Merge 的流程圖如下：

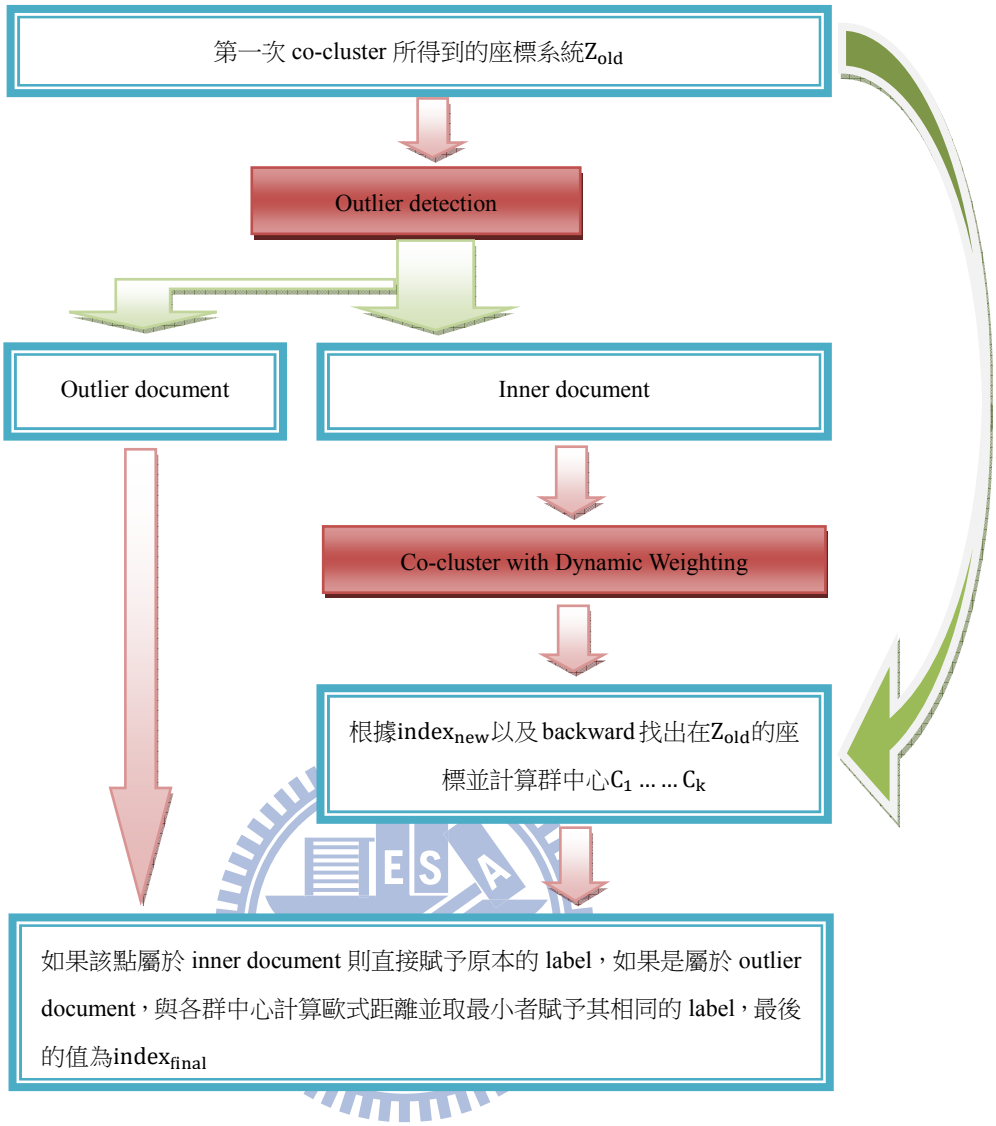


圖3-13 Merge的流程圖

3.4.2 演算法

根據圖 3-13 的流程圖，以下的演算法為整個程式執行運作的算式。

============== Algorithm 4. Merge function ==============

intput Given inner document clustering $index_{new}$、Z_{old}、innerdoc、outlierdoc

output Clustering result $index_{final}$

1. **begin**
2. **Compute K inner document cluster centers $C_1, \ldots, C_k$ on Z_{old} using innerDoc and $index_{new}$ information**
3. $index_{final} = index_{new}$
4. **Assign each document d_o in outlierDoc to one of the clusters $C_1, \ldots, C_k$ according to**

   $$C^* = \underset{C \in \{C_1, \ldots, C_k\}}{\operatorname{argmin}} |d_o - C|$$ **and update clustering $index_{final}$**

5. **return $index_{final}$**
6. **end**

3.5 Co-cluster with Dynamic Weighting

3.5.1 流程架構

本小節將針對本實驗的主要核心的流程做說明，給定一個矩陣$\mathbf{A_{m\times n}}$，其中 m 代表文章的篇數，n 代表所有字的數量，根據 3.2 節提到的 outlier detection，我們針對矩陣 A 進行 outlier 的去除動作，進而得到 inner document 以及 outlier document。得到的 inner document 接下來會在進行一次 co-cluster，得到新的座標點後取字的座標$\mathbf{wp_i}$，$\mathbf{i \in \{1 \ldots\ldots n\}}$，經過 3.3 節的 Weighted Term Matrix Construction Algorithm 對所有的字賦予權重之後，透過式子(3.5)、(3.6)產生矩陣 T，我們將這個 T 的矩陣乘以 A 來對矩陣 A 做更新，即$\mathbf{A = A \times T}$再對更新的矩陣再進行 co-cluster 同樣的依舊可以得到新座標點$\mathbf{wp_i}$，然而新的$\mathbf{wp_i}$有著新的分佈，這樣子的分佈我們亦可以在探討每一個字的權重，所以也同樣的透過式子(3.5)、(3.6)再進行更新而得到 T，如此一來每個字可以經過每次的 co-cluster、Weighted Term Matrix Construction Algorithm 進行動態的調整，本次的權重會根據上一次給定的分群所產生的分佈進行調整，最後 output 的結果是 inner document 的 index。我們將根據 3.4 的 merge function 與 outlier document 進行合併，最後的結果再來做評估，整個 Co-cluster with Dynamic Weighting 的流程架構圖如下：

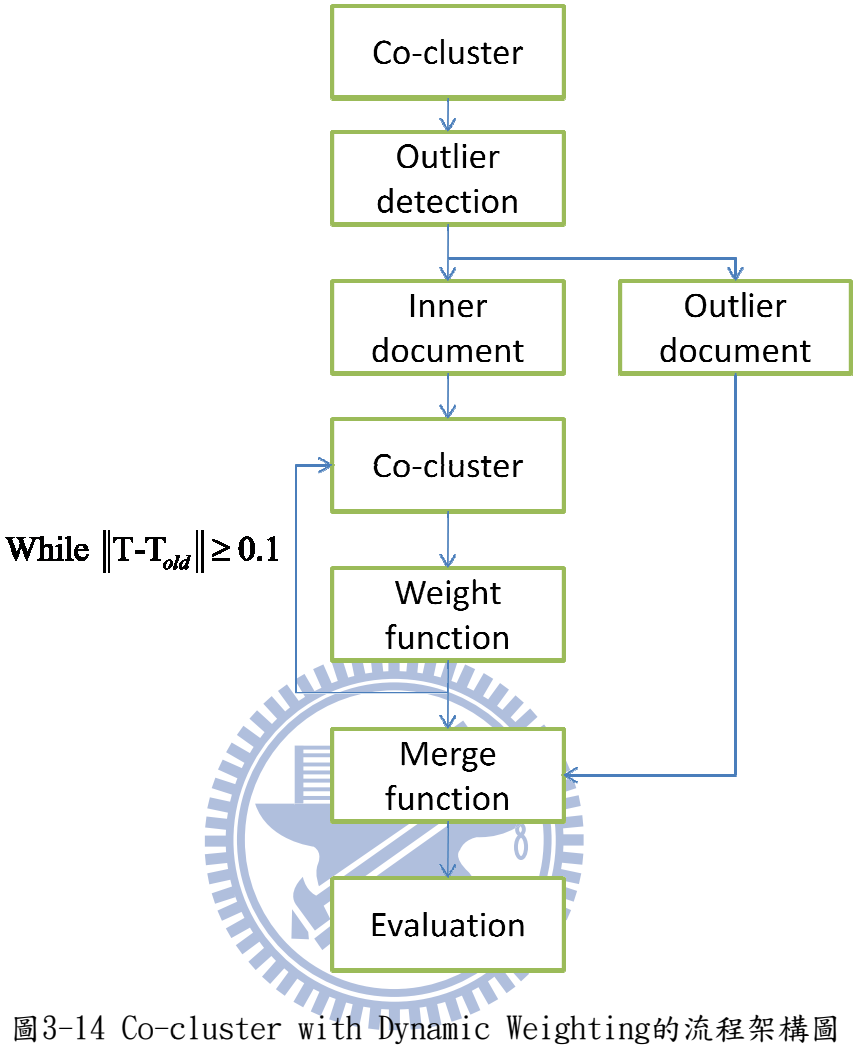


圖3-14 Co-cluster with Dynamic Weighting的流程架構圖

3.5.2 演算法

本小節將列出本實驗的主要演算法，我們將會總合 3.2、3.3、3.4 的方法，最後在 3.5.3 我們將評估效果。

======== Algorithm 5. Co − cluster with Dynamic Weighting ======

1. Initial $T_{initial} = T_{old} = T$，$r = 1$
2. Given A，from $A_n = D_1^{-\frac{1}{2}} A D_2^{-\frac{1}{2}}$
3. $[index, Z] = co - cluster(A)$，and set $Z_{old} = Z$
4. $[innerdoc, outlierdoc] = Outlier\ detection(Z_{old})$
5. $A \leftarrow innerdoc$
6. While$(r > 0.1)$
7. $[index, Z] = co - cluster(A)$，and set $Z_{new} \leftarrow Z$，
   which is $L -$ dimension data ，and $L = \lceil \log_2 k \rceil$
8. $T = weight\ function(Z_{new})$
9. $A \leftarrow A \times T$
10. $r = \|T - T_{old}\|$
11. $T_{old} \leftarrow T$
12. end
13. Run $k -$ means algorithm on the $L -$ dimension data Z to obtain the desired $k -$ way multipartition，$L = \lceil \log_2 k \rceil$，and set index $\rightarrow$ $index_{new}$
14. $index_{final} = merge\ function(index_{new}, Z_{old}, innerdoc, outlierdoc)$

第四章、實驗結果與討論

4.1 實驗資料

我們的實驗選取了三個語料庫，在文件分析的實驗中著名的兩個語料庫為20newsgroups[1]以及 Reuters-21578[2]，第三個為 Dhillon 發表在 KDD2001 年的論文[1]中的語料庫(Classic3[3])。

(1). 20newsgroups 的語料庫中包含了 20,000 篇新聞的文章，其中分成了 20 個類別，文章中不少類別討論的話題很接近例如：comp.sys.ibm.pc.hardware 以及 comp.sys.mac.hardware 都是在討論有關 hardware 方面的新聞，用詞上也都比較接近，同樣的也有許多文章討論的話題相對的比較懸殊例如：rec.motorcycles 以及 soc.religion.christian。本實驗取的語料庫為 20newsgroups-bydate 的版本，此版本根據日期作排列並且除了去除了在文章中組別的編列文字之外(Xref, Newsgroups, Path, Followup-To, Date)，同時也去除了一些重複的文章，經過這樣以上的步驟，實際所剩下的文章數量為 18,846 篇文章。

(2). Reuters-21578 的語料庫包含了 21578 篇文章，是路透社(Reuters)新聞專線在 1987 年所蒐集的文件集並編撰定義了它們各自的類別，在許多 information retrieval 的研究上常常用到該語料庫，語料庫中類別與類別之間的篇章數相當懸殊，最大的類別群(earn)可以到 3753 篇，而最小群可以到 1 篇，而本實驗取篇章數最大的 10 個類別來做實驗。

(3). Dhillon 在 KDD2001 年所發表的論文之中的語料庫我們選取了其中的一個(以下稱 Classic3)來做實驗，其中 Classic3 包含了 MEDLINE(經由醫療的期刊中的摘要部分，共計 1033 筆摘要)、CISI(從 information retrieval 論

[1] http://people.csail.mit.edu/jrennie/20Newsgroups/20news-bydate.tar.gz
[2] http://www.daviddlewis.com/resources/testcollections/reuters21578/reuters21578.tar.gz
[3] ftp://ftp.cs.cornell.edu/pub/smart/

文中的摘要部分，共計 1460 筆摘要)以及 CRANFIELD(是有關於空氣動力學系統的 1400 筆摘要)。

4.2 實驗設計

4.2.1 前處理

首先 Reuters-21578 的語料庫是由 XML 的型式撰寫，本系統將保留<title>以及<body>內的文字，接下來對所有的語料庫我們會做下列的處理方式：

(1). Stop Words 移除

在文章之中，會有許多的介系詞、動詞、語助詞之類的文字，像是 a、the、an 等等，這些字通常對於分群和分類沒有幫助，也就是 Stop Words，這些字會造成分群的一些 Noise，因此我們將一些常見的 Stop words 建成一個 table，在將資料做處理的時候，會參照這個表，將不必要的 stop words 去除掉，有利於我們的分群和分類。

(2). 統一字母的大小寫一律改小寫、去除特殊符號

在文章中，作者使用的字可能會有大小寫不同的情況，電腦會將大小寫看成是不同的字，因此我們將其所有的文字都改成小寫，以便判斷字的相同。文章中還會有許多雜訊的特殊符號，例如標點符號、羅馬符號等等，這些對於文章的分類、分群是沒什麼幫助的，因此我們會把這些符號刪除，保留對文章有意義的字。

(3). 字頻的次數

文章中出現了許多的稀有字，在所有的字裡面這些字的個數其實已經涵蓋了一半以上，然而這些稀有字因為出現的次數過少，其實對於文章並沒有太大的意義，但是過量卻足以影響整個實驗的效果，面對這樣的問題，我們系統將只保留字頻總次數超過 30 次以上的字，比方說一個字”A”出現在所有的

文章的總次數只有 29 次，那這個字將會被刪除，這樣的做法一方面可以減少稀有字對分群的影響更可以降低矩陣的維度以加快計算的速度。

4.2.2 F1 cluster evaluation measure

Clustering 的部分，本論文採用 F1 cluster evaluation measure[39]來評估分群結果的好壞。當系統在做分群時，系統並不會知道哪個群是哪個類別，系統只會將所有的資料分成指定的群數，所以計算效能時，是將分群的結果和真實的類別作比較，此方法 Ramage[40]等人亦使用過，比較方是下列四種情況：

1. True Positives(TP)：

   系統將兩篇文章分在同一群，而這兩篇文章真實也是在同個類別內。

2. False Positives(FP)：

   系統將兩篇文章分在同一群，但是這兩篇文章真實不是在同個類別內。

3. True Negatives(TN)：

   系統將兩篇文章分在不同群，而這兩篇文章真實也不在同個類別內。

4. False Negatives(FN)：

   系統將兩篇文章分在不同群，但是這兩篇文章真實是在同個類別內。

F1 cluster evaluation measure 的算法定義如下：

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 * precision * recall}{precision + recall}$$

4.2.3 比較方法

實驗除了本實驗的方法外，為了不失公平性，以下幾種是我們實作以用來跟我們的方法做比較，實作上我們採用$\mathbf{C^{\#}}$來做前處理的部分，方法主軸的部分採用matlab 實作：

(1).K-means：

K-means[4]是一個被大家廣為比較並列為基準的一個方法，原理很簡單，給定一個集合其中包含了 n 個點$\mathbf{x_1 \ldots x_n}$，其中此 n 個點為 d-dimension 的實數向量,k-means 演算法目的將 n 個點分成 k 群$(\mathbf{k \leq n}), \mathbf{S = \{S_1, S_2, \ldots, S_k\}}$以盡量減少點與其所在的群的群中心距離的平方合(within-cluster sum of squares)的方式， 本實驗採用 matlab 內部的 K-means 方法：[Index]=kmeans(matrix,k)，其中我們給定矩陣 matrix 以及目標群 k 經過此方法會得到各點所對應的 label。

(2).Co-cluster

本實驗採用 Dhillon[1]的方法，在 2.3 節有此方法詳細的介紹。

(3).PLSA：在 PLSA 的實作方面，我們參考此方法[38]，從本質上來說，PLSA 是基於混合式分解(mixture decomposition)進而得到潛在的類別模組(latent class model) ，它的標準程序為利用 Expectation Maximization(EM)[8]演算法藉由淺在的變數模組(latent variable models)估計最大概似(maximum likelihood)，此演算法包含了兩個步驟 E-step 以及 M-step，E-step 會根據目前估計的參數，來計算潛變量 z 的後驗機率，在 M-step 參數會根據先前 E-step 所獲得的後驗機率來進行更新，當我們給定 word $\mathbf{w \in W = \{w_1, w_2, \ldots, w_M\}}$在文章$\mathbf{d \in D = \{d_1, d_2, \ldots, d_N\}}$，E-step 的公式如下：

$$P(z_k|d_i, w_j) = \frac{P(w_j|z_k)P(z_k|d_i)}{\sum_{l=1}^{K} P(w_j|z_l)P(z_l|d_i)} \quad (4.1)$$

估計$P(d_i) \propto n(d_i)$可以被獨立進行的，而 M-step 用來重新估計參數的算式如下：

$$P(w_j|z_k) = \frac{\sum_{i=1}^{N} n(d_i, w_j)P(z_k|d_i, w_j)}{\sum_{m=1}^{M}\sum_{l=1}^{N} n(d_i, w_m)P(z_k|d_i, w_m)} \quad (4.2)$$

$$P(z_k|d_i) = \frac{\sum_{j=1}^{M} n(d_i, w_j)P(z_k|d_i, w_j)}{n(d_i)} \quad (4.3)$$

4.3 實驗成果

本節將本實驗的結果一一呈現出來，首先在表 4-1 是 20newsgroups 的分群結果，其中包含 2 群與 4 群的結果，本實驗 F1-value 取最大值作為最後的分數，其中我們的方法所得到的結果相較於其他方法來的好，表 4-2 是 Reuters-21578 的實驗顯示結果，

	k-means	PLSA	Co-cluster	Co-clustering with Dynamic Weighting
talk.politics.guns & talk.politics.mideast	66.63%	76.76%	66.63%	93.63%
rec.autos & rec.motorcycles	66.62%	69.65%	64.39%	75.85%
alt.atheism & comp.graphics	66.75%	96%	94.02%	96%
rec_autos & rec_baseball	65.50%	62.08%	89.47%	91.63%
alt.atheism & comp.graphics & rec_baseball	50.17%	94.39%	64.91%	94.09%

表 4-1 20newsgroups 的分群結果，值為 F-1value

	k-means	PLSA	Co-cluster	Co-clustering with Dynamic Weighting
crude & money-fx	65.37%	62.63%	62.56%	96.66%
corn & ship	67.40%	64.96%	67.43%	80.64%
corn & crude	65.49%	79.78%	90.46%	95.02%
grain & interest	67.56%	97.25%	97.26%	97.47%
ship & trade	59.02%	58.49%	59.05%	83.80%
grain & ship	63.50%	85.59%	70.65%	86.01%
trade & wheat	69.53%	95.16%	78.58%	92.55%
crude & grain & money-fx	48.75%	64.41%	66.31%	67.45%
money-fx & ship & trade & wheat	39.58%	53.32%	52.15%	52.85%
grain & interest & money-fx & ship & trade	32.66%	53%	43%	46.45%
crude & earn & grain & interest & money-fx	45.85%	59.48%	59.94%	60.02%

表 4-2 Rruters-21578 的分群結果，值為 F-1value

	k-means	PLSA	Co-cluster	Co-clustering with Dynamic Weighting
cisi & med	67.34%	97.62%	93.68%	96.65%
med & cran	60.68%	98.41%	99.20%	99.36%
cisi & cran	66.18%	98.61%	97.58%	97.65%
Classic3	55.84%	97.14%	95.00%	96.38%

表 4-3 Classic3 的分群結果，值為 F-1value

4.4 實驗討論

4.4.1 效果與評估

本節我們將探討 Co-cluster with Dynamic Weighting 方法的實際的效果，首先我們以 talk.politics.guns 以及 talk.politics.mideast 的實驗結果做分析，根據圖 3-6 以及圖 3-7，我們可以發現在去除 outlier 之後經過 co-clustering 的分群可以明顯的感覺到相較於先前尚未經過去除 outlier 的分群，效果有顯著的差異，我們在經過一次 co-clustering 以後根據 weight function 取得 weight 矩陣 T 之後將之乘以 A 矩陣得到了新的矩陣之後，我們進行了第二次 iteration 得到了更新的結果如圖 4-1，其對映實際的群分佈圖為圖 4-2，相較於圖 3-6 以及圖 3-7，可以感覺經過第二次的調整，系統分群的效果圖顯示正確的點相較之下有些許的變多，接著我們根據之前的實驗步驟，再進行第三次的 iteration，所得到的結果如圖 4-3，實際點的分佈圖如 4-4，我們可以發現在第三次的 iteration 可以明顯的看出點與點之分佈，明顯的比第一次的圖(圖 3-6、圖 3-7)密集很多，在此我們利用 F1-value[39]的方法作效能的評估，詳細的 F1-value 的計算方法我們將會在第四章做說明，然而在第一次 iteration 的調整下，系統的 F1-value 為 90.27%，第二次 iteration 的調整，系統 F1-value 為 93.08%，第三次 iteration 的調整下，系統的 F1-value 為 93.71%。

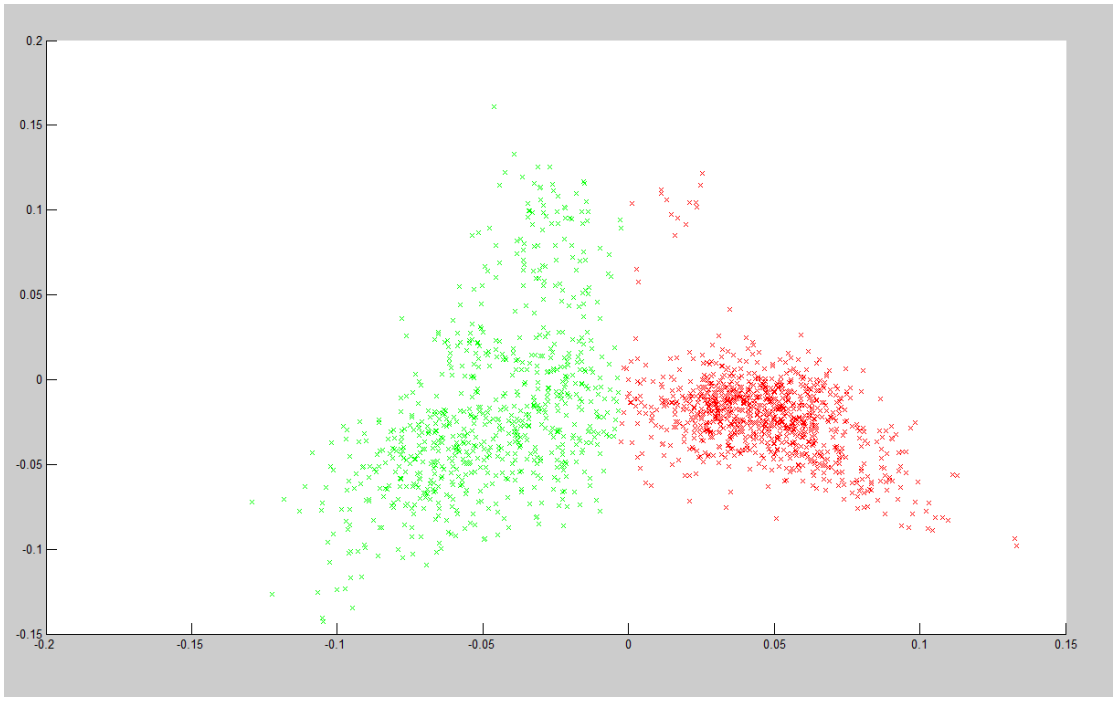


圖4-1 進行第二次iteration系統分群圖

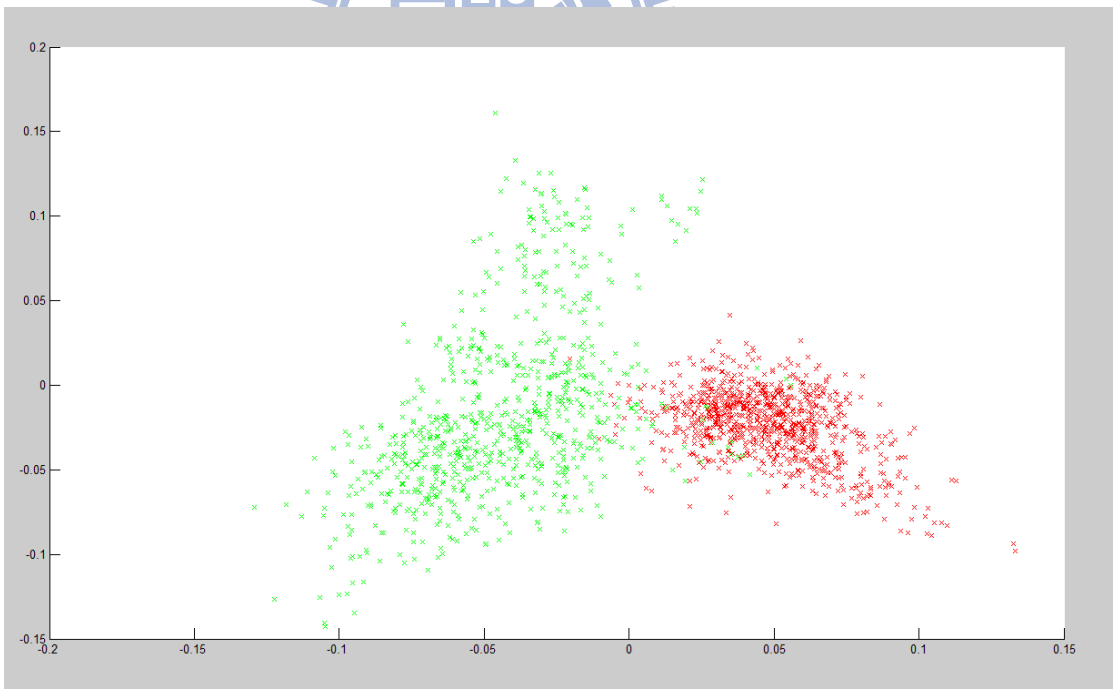


圖4-2 進行第二次iteration實際點分佈圖

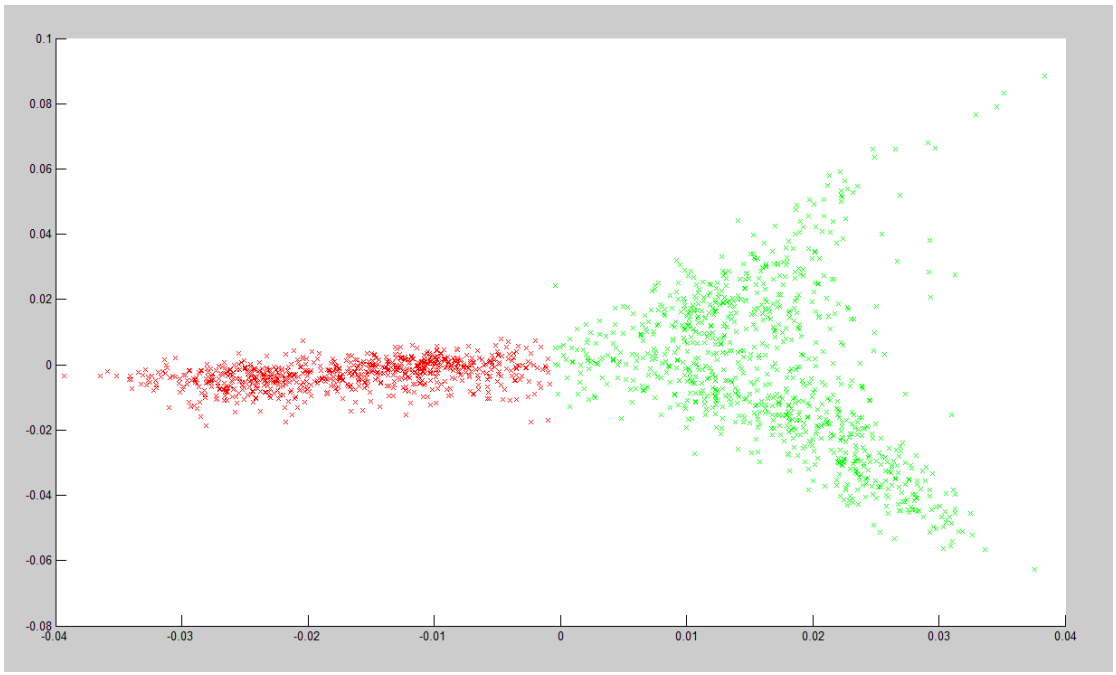


圖4-3 進行第三次iteration系統分群圖

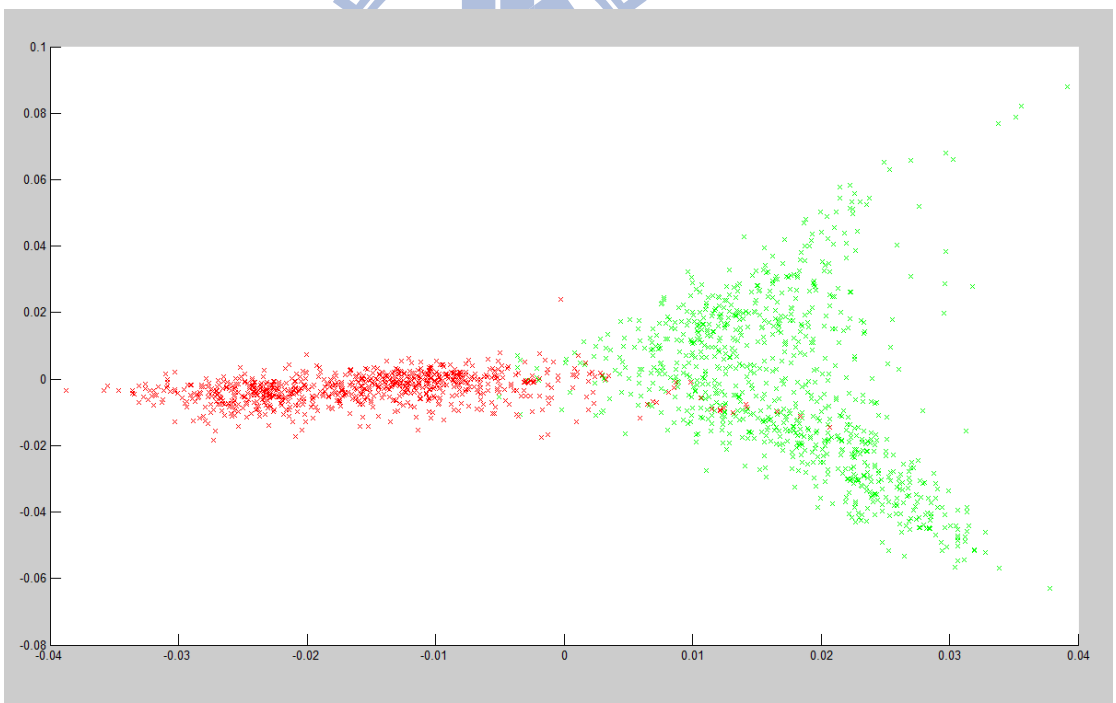


圖4-4 進行第三次iteration實際點分佈圖

Co-clustering 的演算法的基本架構在 Spectral clustering，在傳統的 spectral clustering 建立一個 semi-definite 的矩陣 **K，其中$K = GG^T$的型式**

的矩陣要找出 normalized cuts 必須要花上$\mathbf{O(n^3)}$的時間複雜度，同樣的 co-clustering 再進行 SVD 分解的同時必須花上$\mathbf{O(n^3)}$的時間複雜度，然而本系統的演算法再進行第一次的 co-clustering 需花上$\mathbf{O(n^3)}$，再進行 iteration 的計算每次必須花上$\mathbf{O(n^3)}$加上 weighted function 的總時間$\mathbf{O(nk^2 + n^3)}$其中$\mathbf{O(nk^2)}$為 weight function 的時間，$\mathbf{O(n^3)}$為矩陣相乘，然而在最後 merge function 的時間複雜度為$\mathbf{O(kn)}$，所以為了避免 iteration 的次數進行的太多而達到的效果又只有微量的進步，我們設立停止的條件為當$\mathbf{r = \|T - T_{old}\| \leq 0.1}$則迴圈終止。我們考慮到了當這一次的 T 的分佈與上一次 T 也就是$\mathbf{T_{old}}$的分佈達到很接近的情況下停止，下圖 4-5、圖 4-6 是我們針對 20newsgroups 的 corpus 中取 talk.politics.guns 以及 talk.politics.mideast 兩筆資料以及 Reuters21578 取 crude 及 money-fx 兩筆資料進行 Co-cluster with Dynamic Weighting 方法經過第一次、第二次以及第三次...等 iteration，根據 r 值的變化量並且針對內點的 F1-value 所作的圖。

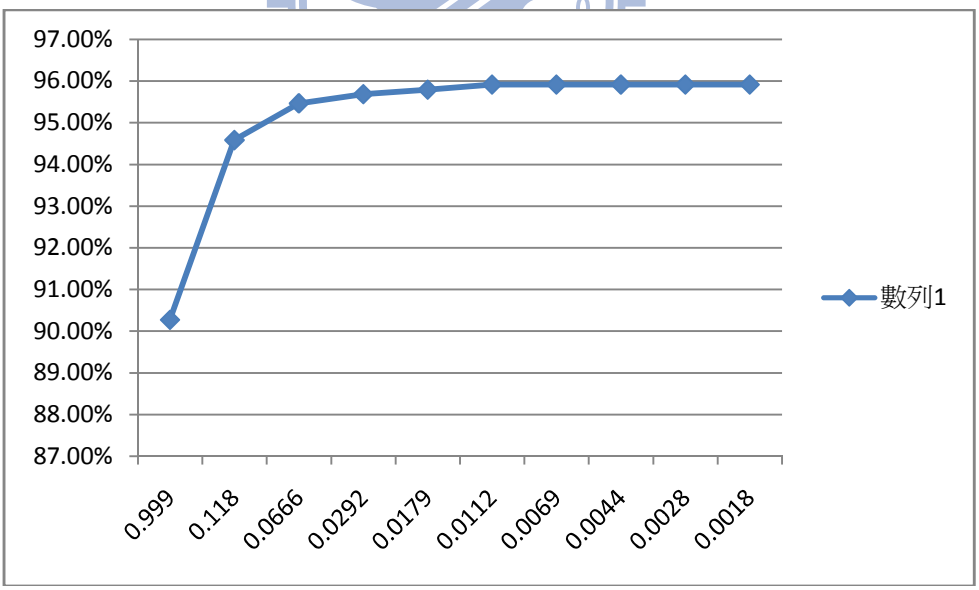


圖4-5 資料talk.politics.guns以及talk.politics.mideast各iteration r值的變化量與F1-value的圖表，其中橫軸為r值，縱軸為內點F1-value曲線圖

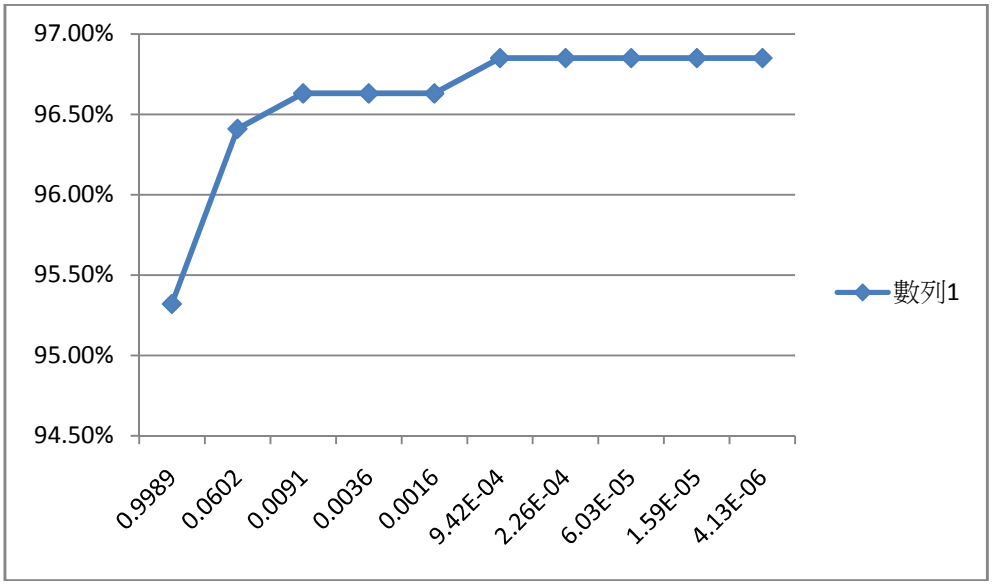


圖4-6 資料crude及money-fx各iteration r值的變化量與F1-value的圖表，其中橫軸為r值，縱軸為內點F1-value曲線圖

根據實驗的結果，我們可以知道在第一次 iteration 至第二次 iteration 的結果內點的 F1-value 變化量有明顯的提升，在二次到之後的 F1-value 的成長都在 1%之間，故我們定 r 取≤ **0.1**這樣即可讓我們高效率且高效能。

4.4.2 字的權重分析

本小節我們將探討字增加權重後的影響，我們將根據 talk.politics.guns & talk.politics.medist 去除 outlier 後文章與字的分佈輸出在一個二維的平面上如圖 4-7，橫軸為所有字的個數，縱軸為文章的篇數，我們可以看出文章與文章中依舊沒有看出很明顯的界限存在；另外使用本系統的方法，我們將字的權重取平均值以後，取大於平均值以上的字輸出文章與字的分佈圖如圖 4-8，以及大於兩倍平均值以上的字輸出文章與字的分佈圖如圖 4-9，根據圖的觀察，若字被賦予的權中越大，由點圖中可看出文章與文章可以看出有明顯的界限存在，我們可以相信被賦予較大權重的字都是較具有鑑別度的字。

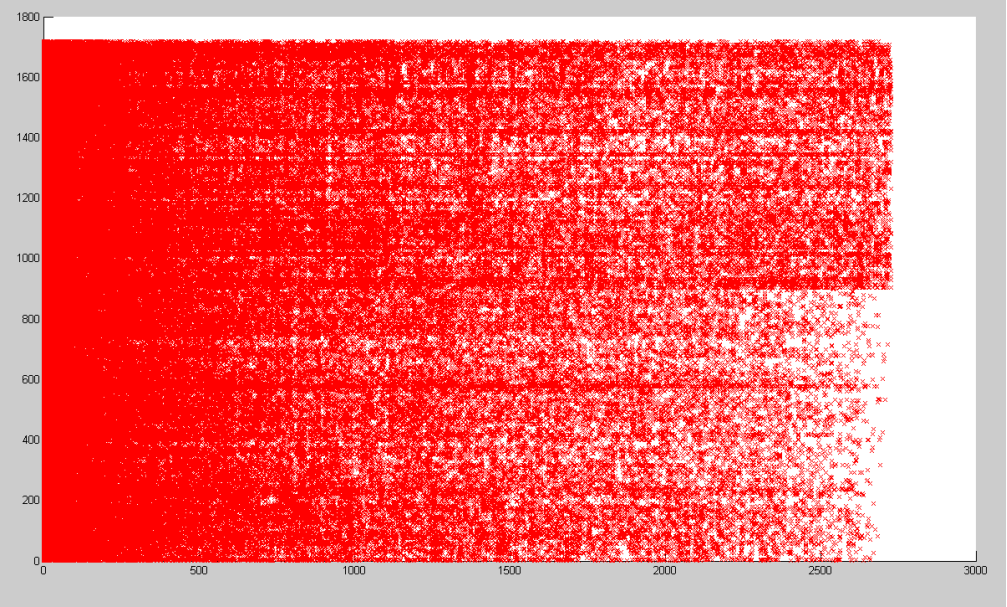


圖4-7經過outlier detection後文章與字的分佈圖

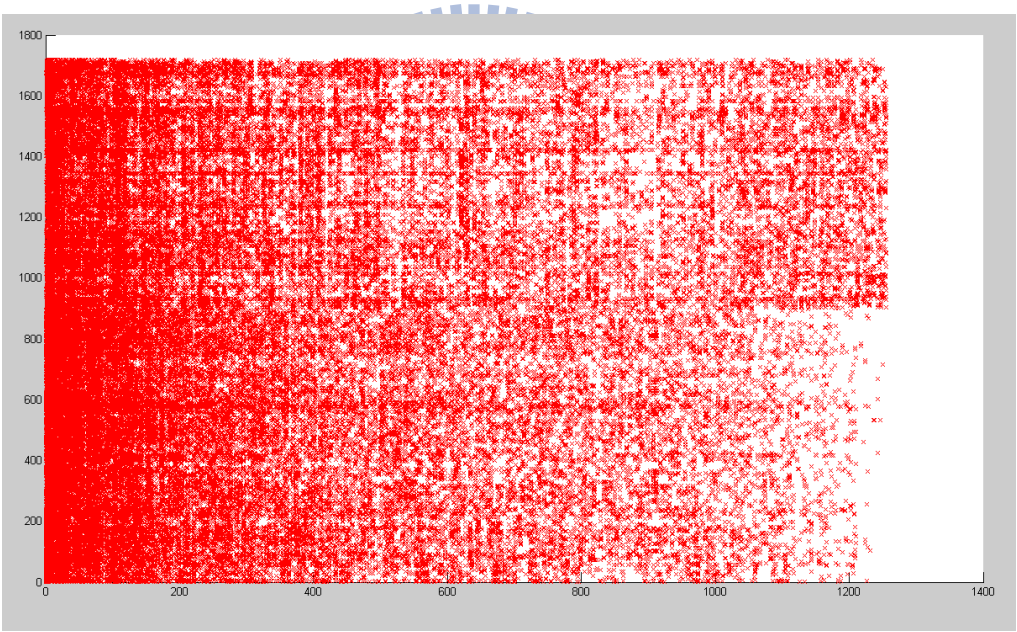


圖4-8權重大於平均值以上文章與字的分佈圖

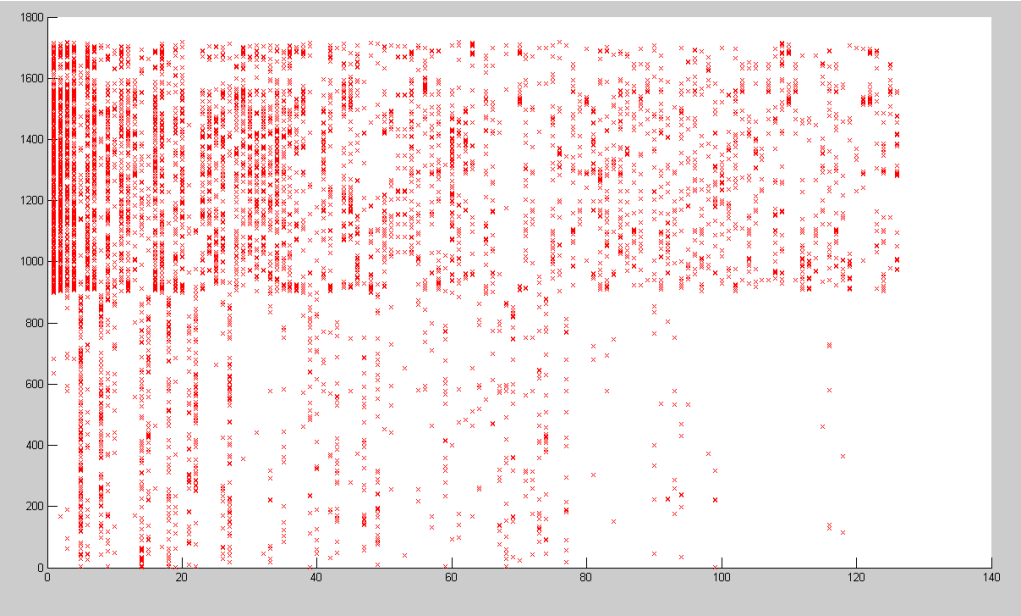


圖4-9權重大於兩倍平均值以上文章與字的分佈圖

第五章、結論與未來展望

5.1 研究總結

本系統經過研究與分析，我們將 spectral co-clustering 校能做更進一步的提升，透過一個加權的系統，使得分群的效果可以有顯著的提升，同樣的我們也分析了其實許多的文字在文章中雖然出現的頻率很高，但是是否可以做為好的特徵來增加分群的效果，同樣的也存在不少字具有好的特徵可以提升分群的效果，但是出現的次數相對的卻不多，本系統可以讓具有好的特徵的字眼加強它的權重，並降低了不具好特徵的字眼的權重已達到好的分群效果。

就整體的表現而言，本系統在分群的效果比起原先的 co-clustering 效果提升了許多，在群數不平均的條件下實驗顯示本系統相較於PLSA依舊是佔有優勢，在網路的社會上許多人會關注的新聞議題會根據新聞的類別而有多寡之分，一個較多人關注的議題相對的文章數量就會遠大於冷門的議題，本系統可以解決許多資料量大小不平衡的語料庫，這點也比較符合現實，但本系統依舊沒辦法解決某些問題，例如：本實驗會使用到 k-means 的分群，但如果這次的群中心找到的是錯誤率很高的群中心，是會影響到之後的結果，在非監督式的學習下這點是很難做到很好。

5.2 未來展望

未來我們希望推導成半監督式的學習法(semi-supervised)，在面對網路上成千上萬的文章，雖然非監督式的學習法可以節省很多的人力，但在效果上依舊是有限制，然而半監督式的學習法可以藉由少量的人力來達到更高的效果，有別於監督式的學習法(supervised)，半監督式學習法在訓練的資料上少很多，在效果上也比非監督式學習法好，此方法可以讓我們可以找到更好更正確的群中心，在面對跌代的計算上也不會因為找到效果較差的群中心而影響結果，使整體更為

穩定，同樣的面對大小不平衡的語料庫，也會因為給定訓練的文章，系統也會依據訓練的文章學習後來修正測試的文章。

參考文獻

[1] I. S. Dhillon, "Co-clustering documents and words using bipartite spectral graph partitioning," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '01. New York, NY, USA: ACM, 2001, pp. 269–274. [Online]. Available: http://doi.acm.org/10.1145/502512.502550

[2] G. Salton and M. J. McGill. Introduction to Modern Retrieval. McGraw-Hill Book Company, 1983.

[3] E. M. Voorhees. *The effectiveness and efficiency of agglomerative hierarchic clustering in document retrieval.* PhD thesis, Cornell University, 1986.

[4] I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine Learning*, 42(1):143–175, January 2001. Also appears as IBM Research Report RJ 10147, July 1999.

[5] H. Sch¨utze and C. Silverstein. Projections for efficient document clustering. In *ACM SIGIR*, 1997.

[6] T. Kohonen. *Self-organizing Maps*. Springer, 1995.

[7] R. V. Katter. Study of document representations: Multidimensional scaling of indexing terms. System Development Corporation, Santa Monica, CA, 1967.

[8] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the em algorithm," *JOURNAL OF THE ROYAL STATISTICAL SOCIETY, SERIES B*, vol. 39, no. 1, pp. 1–38, 1977. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.133.4884

[9] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: a review," *ACM Comput. Surv.*, vol. 31, pp. 264–323, September 1999. [Online]. Available: http://doi.acm.org/10.1145/331499.331504

[10] Y. Chen, L. Wang, and M. Dong, "Non-negative matrix factorization for semisupervised heterogeneous data co-clustering," *IEEE Trans. on Knowl. and Data Eng.*, vol. 22, pp. 1459–1474, October 2010. [Online]. Available: http://dx.doi.org/10.1109/TKDE.2009.169

[11] I. S. Dhillon, S. Mallela, and D. S. Modha, "Information-theoretic co-clustering," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '03. New York, NY, USA: ACM, 2003, pp. 89–98.

[12] Y. Cheng and G. M. Church, "Biclustering of expression data," in

Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology. AAAI Press, 2000, pp. 93–103. [Online]. Available: http://portal.acm.org/citation.cfm?id=645635.660833

[13] S. C. Madeira and A. L. Oliveira, "Biclustering algorithms for biological data analysis: A survey," *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, vol. 1, pp. 24–45, January 2004. [Online]. Available: http://dx.doi.org/10.1109/TCBB.2004.2

[14] S. Busygin, O. Prokopyev, and P. M. Pardalos, "Biclustering in data mining," *Comput. Oper. Res.*, vol. 35, pp. 2964–2987, September 2008. [Online]. Available: http://portal.acm.org/citation.cfm?id=1343112.1343210

[15] T. Hofmann, J. Puzicha, and M. I. Jordan, "Learning from dyadic data," in *Proceedings of the 1998 conference on Advances in neural information processing systems II*. Cambridge, MA, USA: MIT Press, 1999, pp. 466–472.

[16] X. Zhu, Z. Ghahramani, and J. D. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *ICML*, 2003, pp. 912–919.

[17] F. Wei, W. Li, Q. Lu, and Y. He, "Query-sensitive mutual reinforcement chain and its application in query-oriented multi-document summarization," in *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*, ser. SIGIR '08. New York, NY, USA: ACM, 2008, pp. 283–290.

[18] U. Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, pp. 395–416, December 2007. [Online]. Available: http://portal.acm.org/citation.cfm?id=1288822.1288832

[19] W. Li, W.-K. Ng, Y. Liu, and K.-L. Ong, "Enhancing the effectiveness of clustering with spectra analysis," *IEEE Trans. on Knowl. and Data Eng.*, vol. 19, pp. 887–902, July 2007.

[20] M. Filippone, F. Camastra, F. Masulli, and S. Rovetta, "A survey of kernel and spectral methods for clustering," *Pattern Recogn.*, vol. 41, pp. 176–190, January 2008.

[21] H. Zha, X. He, C. Ding, H. Simon, and M. Gu, "Bipartite graph partitioning and data clustering," in *Proceedings of the tenth international conference on Information and knowledge management*, ser. CIKM '01. New York, NY, USA: ACM, 2001, pp. 25–32.

[Online]. Available: http://doi.acm.org/10.1145/502585.502591

[22] T. Li, "A general model for clustering binary data," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, ser. KDD '05. New York, NY, USA: ACM, 2005, pp. 188–197. [Online]. Available: http://doi.acm.org/10.1145/1081870.1081894

[23] W. Li and A. McCallum, "Semi-supervised sequence modeling with syntactic topic models," in *Proceedings of the 20th national conference on Artificial intelligence - Volume 2*. AAAI Press, 2005, pp. 813–818. [Online]. Available: http://portal.acm.org/citation.cfm?id=1619410.1619463

[24] B. Long, Z. M. Zhang, and P. S. Yu, "Co-clustering by block value decomposition," in *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, ser. KDD '05. New York, NY, USA: ACM, 2005, pp. 635–640. [Online]. Available: http://doi.acm.org/10.1145/1081870.1081949

[25] C. Ding, T. Li, W. Peng, and H. Park, "Orthogonal nonnegative matrix t-factorizations for clustering," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '06. New York, NY, USA: ACM, 2006, pp. 126–135. [Online]. Available: http://doi.acm.org/10.1145/1150402.1150420

[26] B. Long, X. Wu, Z. M. Zhang, and P. S. Yu, "Unsupervised learning on k-partite graphs," in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, ser. KDD '06. New York, NY, USA: ACM, 2006, pp. 317–326.

[27] B. Mandhani, S. Joshi, and K. Kummamuru, "A matrix density based algorithm to hierarchically co-cluster documents and words," in *Proceedings of the 12th international conference on World Wide Web*, ser. WWW '03. New York, NY, USA: ACM, 2003, pp. 511–518.

[28] L. A. Park, C. A. Leckie, K. Ramamohanarao, and J. C. Bezdek, "Adapting spectral co-clustering to documents and terms using latent semantic analysis," in *Proceedings of the 22nd Australasian Joint Conference on Advances in Artificial Intelligence*, ser. AI '09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 301–311.

[29] T. K. Landauer, P. W. Foltz, and D. Laham, "Introduction to latent semantic analysis," *Discourse Processes*, vol. 25, pp. 259–284, 1998.

[30] G. Bisson and F. Hussain, "Chi-sim: A new similarity measure for the co-clustering task," in *Proceedings of the 2008 Seventh International Conference on Machine Learning and Applications*. Washington, DC, USA: IEEE Computer Society, 2008, pp. 211–217. [Online]. Available: http://portal.acm.org/citation.cfm?id=1491260.1491396

[31] K. M. Hall. An r-dimensional quadratic placement algorithm. *Management Science*, 11(3):219–229, 1970.

[32] W. E. Donath and A. J. Hoffman. Lower bounds for the partitioning of graphs. *IBM Journal of Research and Development*, 17:420–425, 1973.

[33] M. Fiedler. Algebraic connectivity of graphs. *Czechoslovak Mathematical Journal*, 23:298–305, 1973.

[34] A. Pothen, H. Simon, and K.-P. Liou. Partitioning sparse matrices with eigenvectors of graphs. SIAM *Journal on Matrix Analysis and Applications*, 11(3):430–452, July 1990.

[35] Rayleigh-Ritz theorem.

[36] L. W. Hagen and A. B. Kahng, "New spectral methods for ratio cut partitioning and clustering," *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 11, no. 9, pp. 1074–1085, 1992.

[37] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, 2000.

[38] T. Hofmann, "Unsupervised learning by probabilistic latent semantic analysis," *Mach. Learn.*, vol. 42, no. 1-2, pp. 177–196, 2001.

[39] C. D. Manning, P. Raghavan, and H. Schtze, *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008.

[40] D. Ramage, P. Heymann, C. D. Manning, and H. Garcia-Molina, "Clustering the tagged web," in *WSDM '09: Proceedings of the Second ACM International Conference on Web Search and Data Mining*. New York, NY, USA: ACM, 2009, pp. 54–63.