國立交通大學

管理學院（資訊管理學程）碩士班

碩士論文

運用潛在語意索引的自動化文件分類

Automatic Classification of Text Documents by Using Latent Semantic Indexing

研 究 生：汪若文

指導教授：劉敦仁博士

中華民國九十三年七月

運用潛在語意索引的自動化文件分類

Automatic Classification of Text Documents by Using Latent Semantic Indexing

研 究 生：汪若文　　Student: Juo-Wen Wang

指導教授：劉敦仁博士　　Advisor: Dr. Duen-Ren Liu

國立交通大學

管理學院（資訊管理學程）碩士班

碩士論文

A Thesis

Submitted to Institute of Information Management

College of Management

National Chiao Tung University

in Partial Fulfillment of the Requirements

for the Degree of

Master

in

Information Management

July 2004

Hsinchu, Taiwan, Republic of China

中華民國九十三年七月

運用潛在語意索引的自動化文件分類

研究生：汪若文　　指導教授：劉敦仁博士

國立交通大學管理學院（資訊管理學程）碩士班

摘　要

在資訊擷取的領域中，資訊的搜尋與瀏覽是兩項非常重要的課題。雖然資訊的搜尋提供使用者快速找到所需資料的方法，但過渡依賴文字比對的檢索方式，無法有效處理同義字與一字多義等問題，加上使用者有時不見得能下達良好的搜尋條件，可能導致使用者無法找到真正所需的資料。因此要提供良好的資訊服務，除了提供資訊的搜尋外，透過良好的分類機制，提供資訊瀏覽的服務，是相當重要而具互補效果的功能。要提供相關的文件瀏覽服務，良好的文件分類是非常重要且基本的工作。

文件的分類可分爲兩個步驟：首先將文件以適當的適當的數學形式加以表述，其次是利用適當的分類演算法對文件進行自動分類。文件的分類是一種概念化的工作。傳統以向量空間法對文件進行表述，難以擺脫對於文字的直接依賴。潛在語意索引 (latent semantic indexing) 的目的在於發掘潛藏文件中的語意概念，而語意概念正好是文件分類的關鍵所在，因此將此技術應用於文件的分類，應有不錯的成效。

本研究嘗試使用潛在語意索引技術進行文件的表述，配合中心向量法與 *k*-NN 兩種分類演算法進行自動化文件分類，探討其可行性與效果。另外並以向量空間法配合上述兩種分類演算法作爲對照，比較兩者的分類效果。

本研究探討的是單一分類的問題。研究結果顯示，利用潛在語意索引技術進行文件的表述，配合適當的分類演算法，可以得到穩定的分類結果，因此將潛在語意索引運用於自動化文件分類是可行的。但在本研究中，無論是搭配中心向量法或 *k*-NN 法，運用潛在語意索引的分類正確率都不及向量空間法。至於潛在語意索引技術是否較適合運用於多分類的問題，或是潛在語意索引技術與其他分類演算法搭配可得較佳分類結果，則有待進一步的研究探討。

關鍵詞：自動分類、資訊擷取、潛在語意索引、向量空間法

Automatic Classification of Text Documents by Using Latent Semantic Indexing

Student: Juo-Wen Wang Advisor: Dr. Duen-Ren Liu

Institute of Information Management, College of Management

National Chiao Tung University

ABSTRACT

Search and browse are both important tasks in information retrieval. Search provides a way to find information rapidly, but relying on words makes it hard to deal with the problems of synonym and polysemy. Besides, users sometimes cannot provide suitable query and cannot find the information they really need. To provide good information services, the service of browse through good classification mechanism as well as information search are very important.

There are two steps in classifying documents. The first is to present documents in suitable mathematical forms. The second is to classify documents automatically by using suitable classification algorithms. Classification is a task of conceptualization. Presenting documents in conventional vector space model cannot avoid relying on words explicitly. Latent semantic indexing (LSI) is developed to find the semantic concept of document, which may be suitable for the classification of documents.

This thesis is intended to study the feasibility and effect of the classification of text documents by using LSI as the presentation of documents, and using both centroid vector and *k*-NN as the classification algorithms. The results are compared to those of the vector space model.

This study deals with the problem of one-category classification. The results show that automatic classification of text documents by using LSI along with suitable classification algorithms is feasible. But the accuracy of classification by using LSI is not as good as by using vector space model. The effect of applying LSI on multi-category classification and the effect of combining LSI with other classification algorithms need further studies.

Keywords: automatic classification, information retrieval, latent semantic indexing, vector space model

誌謝

從沒想過會在工作多年之後重回校園讀書。回想當年，幾乎是以逃離與解脫的心情從校園進入職場，怎知面對工作上對自我能力的要求與挑戰，自覺學識之不足，只好硬著頭皮重作學生。

經過幾年的努力掙扎，終於得以畢業。首先要感謝的是指導教授劉敦仁博士，經由他的指引，我才能在漫無頭緒之中找到研究的方向。也要感謝他在研究方法上諸多指點與建議，以及對於我那般拖拉遲緩所表現的寬容。

其次要感謝王朝煌教授、陳恭教授與陳安斌教授三位口試委員，他們對於本論文的指導與建議，讓我獲益甚多，他們對我的鼓勵與嘉勉，更讓我銘感五內。

能在工作之餘進修學習，要特別感謝國科會精密儀器發展中心的長官與同仁，他們的支持與鼓勵，讓我無工作上的後顧之憂。感謝資管所的老師、同學與實驗室的伙伴，有他們的指導與相伴，讓我忘卻了求學的辛苦，也讓我在學習的旅途中增添了許多的樂趣。還要感謝美國田納西大學計算機科學系的 Michael W. Berry 教授，他慷慨提供其開發的 GTP 軟體，讓我的研究得以順利進行。

此外，要感謝我的父母親，他們養我、育我，期盼我做個有用的人，我總算沒有太過辜負他們。

最要感謝的是靜芬，謝謝她這些年來的包容，如果不是她的鼓勵與鞭策，我根本不可能會想再回學校讀書。也因爲她的相伴與照顧，我才能順利完成學業。

目錄

中文摘要 i
英文摘要 ii
誌謝 iii
目錄 iv
圖目錄 vii
表目錄 viii
第一章　緒論 1
1.1　研究背景與動機 1
1.2　研究目的 2
1.3　論文架構 3
第二章　文獻探討 4
2.1　IR 模型與自動化文件處理 4
2.2　自動化文件分類的概念 6
2.3　向量空間法 7
2.3.1　向量空間法的文件表述 7
2.3.2　文件的比較 10
2.3.3　文件的檢索 10
2.4　潛在語意索引技術 11
2.4.1　LSI 運用的 SVD 技術 11
2.4.2　LSI 的幾何描述 13
2.4.3　運用 LSI 進行文件與索引詞的比較 13
2.4.4　虛擬文件的表述 14
2.4.5　LSI 的應用探討 15
2.5　自動化文件分類法 16
2.5.1　中心向量分類法 16
2.5.2　*k*-NN 分類法 17
2.5.3　其他分類法 19

2.6 運用 LSI 於文件分類的研究 19
第三章 研究方法 21
3.1 研究方法概述 21
3.2 文件資料 21
3.2.1 Inspec 文件集 21
3.2.2 字根的處理 25
3.3 軟硬體說明 26
3.3.1 LSI 軟體 26
3.3.2 其他軟硬體環境 27
3.4 向量正規化與 **S** 矩陣的探討 28
3.4.1 文件向量的正規化 28
3.4.2 LSI 中測試文件的處理 30
3.4.3 SVD 運算中的 **S** 矩陣 30
3.5 實驗步驟 31
3.5.1 實驗步驟概述 31
3.5.2 文件資料的整理與前處理 31
3.5.3 選取訓練文件與測試文件 31
3.5.4 對訓練文件進行 LSI 的 SVD 處理 32
3.5.5 分類的訓練 32
3.5.6 測試文件的分類 33
3.5.7 中心向量分類法的實驗變因設計 33
3.5.8 *k*-NN 分類法的實驗變因設計 37
3.5.9 實驗結果的評估 38
第四章 研究結果與討論 39
4.1 分類正確率 39
4.1.1 利用中心向量法的分類正確率 39
4.1.2 利用 *k*-NN 法的分類正確率 40
4.2 分類效率 42
4.2.1 利用中心向量法的分類效率 42
4.2.2 利用 *k*-NN 法的分類效率 43
4.3 綜合討論 44
4.3.1 LSI 與傳統向量空間法的分類結果比較 44

4.3.2 各種實驗變因的影響 45
4.3.3 中心向量法與 *k*-NN 法的比較 46
第五章 結論與建議 47
5.1 研究結論 47
5.2 研究限制 48
5.3 未來研究建議 48
參考文獻 50

圖目錄

圖 2-1 資訊擷取模型 (information retrieval models) 的分類架構圖 5
圖 2-2 索引詞出現頻率與索引詞 rank order 的關係示意圖 8
圖 2-3 向量空間法概念圖 9
圖 2-4 兩文件向量相似度的比較 10
圖 2-5 對一個索引詞—文件向量矩陣進行 SVD 處理 12
圖 2-6 索引詞—文件向量經過降階 SVD 處理 12
圖 2-7 文件群集的概念示意圖 16
圖 2-8 *k*-NN 分類法的示意圖 18
圖 3-1 本研究所用文件資料集 Inspec on Disc 的分類架構示意圖 22
圖 3-2 向量正規化與否對於向量加總的影響 29

表目錄

表 3-1 Inspec 文件集的分類架構 23
表 3-2 Inspec 文件集所選 38 類別各擁有的文件數量 25
表 3-3 LSI 搭配中心向量法的實驗變因組合 36
表 3-4 LSI 搭配 *k*-NN 法的實驗變因組合 37
表 4-1 LSI 搭配中心向量法所進行之分類實驗的結果 39
表 4-2 傳統向量空間法搭配中心向量法所進行之分類實驗的結果 40
表 4-3 LSI 搭配 *k*-NN 法所進行之分類實驗的結果 41
表 4-4 傳統向量空間法搭配 *k*-NN 法所進行之分類實驗的結果 41
表 4-5 LSI 搭配中心向量法進行分類實驗時，每份測試文件平均所花費的時間 42
表 4-6 傳統向量空間法搭配中心向量法進行分類實驗時，每份測試文件平均所花費的時間 42
表 4-7 LSI 搭配 *k*-NN 法進行分類實驗時，每份測試文件平均所花費的時間 43
表 4-8 傳統向量空間法搭配 *k*-NN 法進行分類實驗時，每份測試文件平均所花費的時間 44

第一章　緒論

1.1　研究背景與動機

在傳統資訊擷取 (information retrieval, IR) 的領域中，資訊的搜尋 (search) 與瀏覽 (browse) 一直是兩項非常重要的課題。資訊的搜尋指的是利用適當的搜尋條件，找出符合所需的資訊。資訊的瀏覽則是將資訊經過適當的整理，以結構化的方式提供給使用者，讓使用者透過結構化的路徑，瀏覽其感興趣的資訊。

隨著資訊技術的進步與網際網路的普及，許多期刊與雜誌開始走向數位化或電子出版的型態。所謂電子化的出版並不是將出版品以數位型態呈現即可，完善的電子出版系統應該考慮如何有效運用資訊的技術，提供使用者良好的文件瀏覽與檢索服務，甚至針對使用者個人的專長或興趣背景，進行資訊過濾處理 (information filtering)，提供主動式的個人化文件推薦服務。

目前許多國外的期刊，尤其是科技性的期刊，已經提供電子期刊的服務。通常較簡單的電子期刊服務是提供逐期的目錄，讓使用者瀏覽，然後查閱其感興趣的文章。較完善的服務則提供良好的檢索介面，讓使用者可選擇性地針對文章標題、作者、摘要、出版時間等各種屬性的資料欄位進行檢索，搜尋其所需的資訊。而提供給使用者的資訊通常是文章的摘要或全文。

除了電子期刊資料庫外，快速成長的大量網頁資料也成了 IR 領域積極處理的對象。如何在浩瀚的網頁中找到有用的或符合需求的資訊，是龐大網頁資料能否展現其潛在價值的重要關鍵。而目前多數的網頁資料搜尋都是透過全文檢索 (full-text index) 的技術，找出符合使用者查詢條件的網頁。

不論是電子期刊的文件檢索或網頁資料的搜尋，如此的資料搜尋基本上都是根據文字的比對，雖然比對的進行與結果可以非常精確，但卻非完美。因爲各種語言的文字都具有同義字 (synonymy) 與一字多義 (polysemy) 的問題，透過文字的精確比對，無法找出含有同義字的其他資料，也可能會找出含有相同文字而意義卻不相同的無關資料。此外，使用者有時不見得能下達良好的搜尋條件，又或者使用者與文件資料的作者對於

字彙的使用缺乏一致性，都可能導致使用者無法找到真正所需的資料。

因此，要提供良好的資訊服務，除了提供資訊的檢索與搜尋外，透過良好的分類機制，提供資訊瀏覽的服務，是相當重要而具互補效果的功能。要提供相關的文件瀏覽服務，良好的文件分類是非常重要且基本的工作。

傳統的文件分類多仰賴人工，也就是由對文件內容涉及之知識領域熟稔的專家進行。人工分類需要耗費不少人力資源，且不同的專家可能有不同的的主觀判斷，而影響文件的分類結果。隨著電腦資訊技術的發展，許多依賴電腦的自動化文件分類方法應運而生。自動化的文件分類方法有些是統計分析文件所含的關鍵字，有些是利用機率的分析，有些則利用類神經網路等人工智慧進行學習，其中以統計分析關鍵字最爲常見。

但文件的分類是一種概念化的工作，而單純的字彙並不能完全與類別相互對映，加上關鍵字所隱含的同義字與一字多義等問題，使得利用關鍵字的統計分析進行文件分類有其難以避免的缺陷。雖然有其他研究嘗試以自然語言的分析，掌握字彙、文詞與文件資料的語意，進行相關的處理，但若要透過自然語言的分析，尋找出文件所隱含的類別概念，以目前的資訊技術而言仍不夠成熟。

在這種兩難的情況下，潛在語意索引 (latent semantic indexing, LSI) 分析技術似乎提供了一個相當不錯的解決方法。LSI 係從傳統向量空間法出發，也是以文件的關鍵字爲基礎，但利用適當的數學方法對文件與關鍵字的關係進行轉換，而得以獲取隱藏在文件關鍵字中的語意成分。

原本 LSI 係應用於文件的檢索，主要目的在解決關鍵字的同義字問題，並獲取較佳的處理效率。經過許多相關的研究，證實其在文件資料的檢索與資訊過濾 (information filtering) 方面有不錯的效果，而且執行效率也較傳統向量空間法爲佳。因爲 LSI 技術係將關鍵字轉換成語意概念，而語意概念正好又是文件分類的關鍵所在，因此，將 LSI 應用於文件的分類，應有不錯的成效。

1.2 研究目的

隨著資訊技術的快速發展，企業組織越來越強調知識管理的重要。而在企業組織內，最重要的知識來源便是各式各樣的文件。因此不論是企業的知識管理，或是各種電子期刊，亦或是網際網路上大量的網頁，所面對的都是文件資料。而要善用這些文件資

料，除了透過良好的搜尋檢索方法，將文件資料妥善地分類，提供使用者良好的文件瀏覽服務，也是相當重要的。

本研究嘗試對於自動化文件分類的問題進行探討，並尋求可能的新方法。本研究將由傳統的 IR 技術出發，並以 LSI 技術作爲傳統向量空間法的替代模式，配合適切的分類演算法，探討如何將 LSI 技術應用於自動化的文件分類，並與傳統向量空間法進行對照，進一步分析運用 LSI 於自動化文件分類的可行性。

1.3 論文架構

本論文旨在說明了本研究進行的動機與背景，探討相關的文獻，說明研究進行的方法以及研究所得結果，並就研究結果進行探討，提出未來可能的繼續研究方向。

本論文共分九章，各章內容簡述如下。

第一章是緒論。說明本研究的背景與動機、研究的目的，以及本論文的架構。

第二章是文獻探討。主要整理相關的文獻，說明自動化文件處理與自動化文件分類的基本概念，並介紹相關的理論與研究。包括可用於文件表述的傳統向量空間法以與由其變形衍生的潛在語意索引技術，以及已獲得廣泛研究的多種分類演算法。

第三章是研究方法。本研究運用潛在語意索引技術，配合中心向量法與 *k*-NN 兩種分類演算法，進行自動化的文件分類，並以傳統向量空間法做爲對照，探討運用潛在語意索引技術的可行性。本章詳述研究方法的概念與施行方式，包括文件資料的整理、研究使用的軟硬體、重要的研究考量，以及詳細的執行步驟。最後並說明本研究結果的評估方法。

第四章是研究結果與討論。本章針對前一章所提之研究方法與執行步驟所進行之自動化文件分類的實驗，整理各項實驗數據與結果，並對所得結果進行討論。

第五章是結論與建議。本章整理本研究的主要發現與結論，並對可能的改進方向與未來可行的相關研究提出建議。

最後則附上本研究所參考的各項文獻資料。

第二章　文獻探討

在過去二十年裡，隨著網際網路的興起與 WWW (world wide web) 的發展，資訊擷取 (information retrieval, IR) 的領域也因應實際的需求而有快速的發展。除了 IR 領域傳統的目標：資訊的索引 (indexing) 與搜尋 (search)，文件的分類 (classification and categorization)、資訊的過濾、文件探勘 (text mining)、視覺化處理等也都成了研究的重點。本研究探討文件的分類問題，本章將從自動化文件處理出發，整理並回顧自動化文件分類相關文獻，說明文件分類的理論與技術。

2.1 IR 模型與自動化文件處理

自動化文件處理在 IR 領域中是很重要的課題，因爲所謂資訊擷取所處理的資訊，都可視爲廣義的文件。IR 所處理的文件有許多種類型，除了一般文字類型的文件外，還有影像、聲音、影片等多媒體型態的資訊與地理資訊等，雖然不同類型的文件可能有不同的處理技術，但對於文字類型文件的處理技術仍然是 IR 的核心技術。

傳統 IR 系統通常處理的工作是利用索引詞 (index term) 對文件進行索引與搜尋。其目的是快速找出符合使用者下達條件的文件，也就是根據使用者的查詢條件 (query)，找出與其相關的文件。

參見圖 2-1，針對上述 IR 傳統的工作，IR 領域中三個傳統的模型分別爲布林 (Boolean)、向量 (vector) 與機率 (probabilistic) [1]。在布林模型中，文件與查詢條件係以索引詞的集合 (set) 來表達，因此可說此爲集合理論的 (set theoretic) 模型。在向量模型中，文件與查詢條件係表述成 t 維空間中的向量，因此也可說此爲代數的 (algebraic) 模型。而在機率模型中，文件與查詢條件係根據機率理論加以表述與處理，因此稱爲機率模型。

隨著 IR 相關研究的推展與資訊技術的發展，有些新興的模型陸續被提出。例如在集合理論模型中有模糊理論 (fuzzy theory) 與擴充的布林 (extended Boolean) 模型；在代數模型中有一般化向量 (generalizedvector) 模型、潛在語意索引 (latent semantic

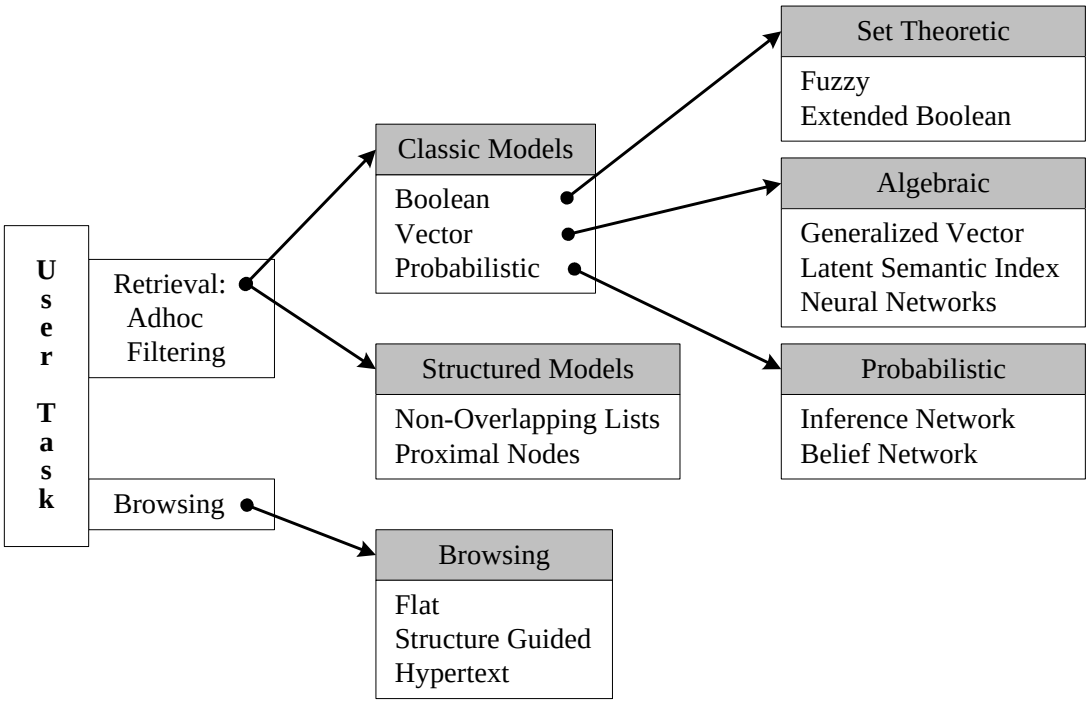

圖 2-1 資訊擷取模型 (information retrieval models) 的分類架構圖。[1]

index) 技術與類神經網路 (nueral network)；在機率模型中則有推論網路 (inference network) 與 Belief 網路 (Belief network)。

利用這些模型與相關的技術，可以處理使用者一項重要的課題：文件的檢索與過濾。所謂文件的檢索是指找出符合查詢條件的文件，文件的過濾是指對文件進行篩選，只將使用者感興趣的文件提供給使用者。但除了這一項課題，使用者還會有另一項重要的課題，就是文件的瀏覽。欲提供文件的瀏覽服務，可以透過結構化的架構將文件予以分類整理，並利用超連結文件 (hypertext) 提升服務的品質 [2]。

文件的分類除了可以有效地整理文件資料外，更可以在使用者不確定適當索引詞(無法下達正確查詢條件) 的情況下，讓使用者透過概念化的分類架構，找到其感興趣或所需要的文件。因爲要下達適切的查詢條件有時並不容易，對某些缺乏訓練的使用者也有相當的困難度，因此將文件予以適當地分類，可彌補文件檢索之不足。

雖然對使用者的應用而言，文件分類與文件檢索是不同的工作，但文件分類所仰賴的許多技術卻是源自於文件檢索使用的技術。後面幾節將陸續討論。

2.2 自動化文件分類的概念

在傳統 IR 的領域中對文件的自動化分類有許多相關的研究。一般認爲文件的分類和文件的群集 (cluster) 有密切的關係，Rijsbergen 強調文件分類的主要應用便是在文件群集 (document clustering) [3]。所謂文件的群集是將類似的文件聚集在一起，而文件的分類則是依據預先定義好的類別，將各文件分派至適當的類別中。

分類的類別基本上可視爲不同意義或概念 (concept) 的集合，隸屬相同類別的文件彼此間應該具有近似的意義或概念，而隸屬不同類別的文件，其文件中所表達的概念則較無關連性。要決定不同的文件是否具有近似的意義或概念，一般是比較文件的相似度，如果兩份文件的相似度越高，越有可能隸屬在相同的類別裡。

運用資訊技術進行自動化的文件分類大致可分成兩個步驟：

1. 特徵的選取 (feature selection) 與文件的表述 (presentation)：利用文件所具有的特徵，將文件以某種適當的數學形式表達，如此才能運用電腦進行文件的處理，而使用的數學形式必須能有效表達出文件所蘊含的意義與概念。
2. 利用適當的分類器 (classifier) 進行自動化的分類：所謂分類器亦可稱爲分類演算法 (classification algorithm)，其可將前一步驟所產生的結果進行適當的運算處理，然後決定文件所屬的類別。

就第一個步驟而言，最直接且常見的方法是利用文件中所含的「文字」(word)，經過適當的處理與權重計算，即可用以表達文件。這部分的研究多以向量空間 (vector space) 的方法爲基礎 [4, 5]，並利用字根 (stemming) 或同義詞等文字處理技術適度改良 [3, 5]。

至於分類器的研究相當多，方法也很多 [6, 7]。比較傳統的方法是利用向量空間法結合 tf-idf (term frequency－inverted document frequency) 的技術 [5]，其他還有 *k*-nearest neighbor (*k*-NN) 法 [8]、Naïve Bayes 分類法 [9, 10]、決策樹 (decision tree, DTree) 分類法 [9, 11]、類神經網路 (neural network, NNet) 分類法 [12, 13]、線性最小平方根法 (linear least squares fit, LLSF) [14]、基因演算法 (genetic algorithm) [15]、supporting vector machine [16] 等。

2.3 向量空間法

2.3.1 向量空間法的文件表述

要達成對文件的表述，必須先將文件所具有的特徵 (feature) 擷取出來。就英語系統而言，要表達一份文件最直接也最簡單的方法是從文件中的「文字」(word) 著手，因爲文字直接傳達了文件所欲表達的文意。在傳統 IR 中，便以文件所含有的文字作爲文件的特徵，而最常見的方法是 Salton 所提出的「向量空間」(vector space) 技術 [4, 5]。在向量空間的方法中，文件的特徵是文件中所含有的關鍵字 (keyword)，透過關鍵字可以適度表達出文件所蘊含的意義與概念。在向量空間法中，將文件所包含的關鍵字以適當的方式構成一向量，該向量即用於表述該文件。首先須將文件的關鍵字萃取出來，而關鍵字的萃取當然是透過電腦自動化進行。此一技術已發展得相當成熟，Salton 便依據此一技術，於康乃爾大學 (Cornell University) 開發出一套稱爲 SMART 的系統 [4, 5]，作爲文件檢索研究之用。

在向量空間的技術中，從文件中萃取出的關鍵字因爲可用於對文件進行索引，因此也稱爲索引詞 (index term)。根據索引詞在各文件中分布的情況，可以計算各個索引詞的權重值。

在實際的處裡中，首先得決定任一索引詞在一份文件中的重要度，一般以 rank order 表示，rank order 越小，表示重要度越高。由 Zipf 定律 (Zipf's law) [17]，任一索引詞在一份文件中的重要度 (亦即 rank order) 與其在該文件中出現的頻率 (term frequency, tf) 有密切的關係，兩者的乘積約爲一常數，以數學形式可以表達如下：

$$\text{frequency} \times \text{rank} \cong \text{constant} \tag{2-1}$$

也就是說，一索引詞在一份文件中出現的頻率越高，其 rank order 越小，重要度便越高；而出現頻率越低，重要度則越低。但有些在文件中出現頻率很高的字 (如介係詞、冠詞、副詞等) 對文件所蘊含的意義並沒有特別的貢獻，並不適合作爲索引詞，一般稱爲虛字 (stop word) 或非索引詞。

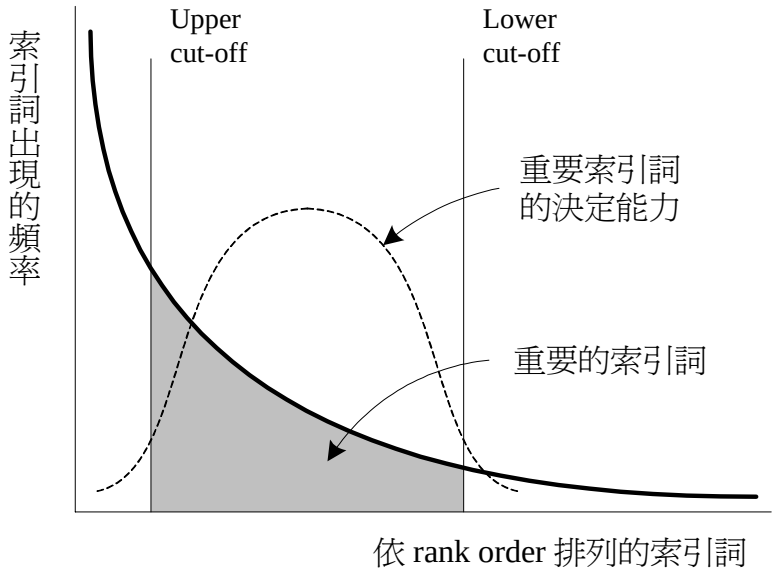

圖 2-2 索引詞出現頻率與索引詞 rank order 的關係示意圖。

基於此一現象，Luhn 提出了一個概念：利用字詞出現的頻率，可以測量該字詞的重要性 [18]。如圖 2-2 所示，字詞的頻率與字詞的 rank order 呈現雙曲線的關係。出現頻率過高的字詞意味著過於普通的字詞 (stopword)，對文件所含意義的影響並不大；出現頻率過低的字詞則意味著太稀少的字詞，對文件所含意義的影響也不大。因此可針對字詞出現的頻率，設定高低門檻 (upper cut-off and lower cut-off)，只保留出現頻率位於此高低門檻之間的字詞，也就是索引詞。而索引詞對文件所含意義具有的決定能力 (resolving power) 如同一個左右對稱的波形，波峰的位置即爲高低頻率門檻的中間。

除了考慮索引詞在一份文件中的重要度外，還要考慮索引詞在所有文件中的重要度。在大量的文件中，如果一個索引詞同時出現在大部分的文件中，則該索引詞用以區分文件意義的功能不大，其重要度也就不高。因此一個索引詞的重要度還需要考慮如下所示的反向文件頻率 (inverse document frequency, idf) [19]：

$$\text{idf} = \log_2 \frac{n}{fd_k} + 1 = \log_2(n) - \log_2\left(fd_k\right) + 1 \qquad (2\text{-}2)$$

其中 n 是文件的總數，fd_k 表示索引詞 k 在所有 n 份文件中出現的頻率 (n 份文件中有幾份含有索引詞 k)。

結合上述兩項考量，當使用索引詞表達文件的意義時，可以用反向文件頻率權重法來表達該索引詞在一份文件中的權重值 [3]。反向文件頻率權重法的基本概念即是以頻

率出發，一個索引詞在一份文件中出現的頻率越高，其重要性越高；但一個索引詞若出現在越多的文件中，其重要性則越低。首先計算索引詞 k 在文件 i 中出現的頻率 f_{ik}，然後計算索引詞 k 在所有 n 份文件中出現的頻率 fd_k (n 份文件中有幾份含有索引詞 k)，則索引詞 k 在文件 i 中的權重值計算如下：

$$w_{ik} = f_{ik} \cdot \left(\log_2 \frac{n}{fd_k} + 1 \right) \tag{2-3}$$

算出全部索引詞在任一份文件中的權重值後，便可以建立文件向量 (document vector) 來表示一份文件，文件向量中的各個元素是各索引詞在該文件中的權重值，例如下式即表示文件 i 的文件向量：

$$\mathbf{d}_i = (w_{i1}, w_{i2}, \ldots, w_{ik}) \tag{2-4}$$

該文件向量共有 k 個維度，對映至 k 個索引詞。如圖 2-3 所示，向量空間法係以各索引詞爲不同維度建構成一向量空間，而文件則依其內含之索引詞及對應之權重關係，形成此向量空間中的向量。

決定了各文件的文件向量後，便完成了以數學形式對文件的表述，透過此種表述，可以利用電腦的運算，對文件進行相關的處理，包括文件的檢索、比較與群集等。

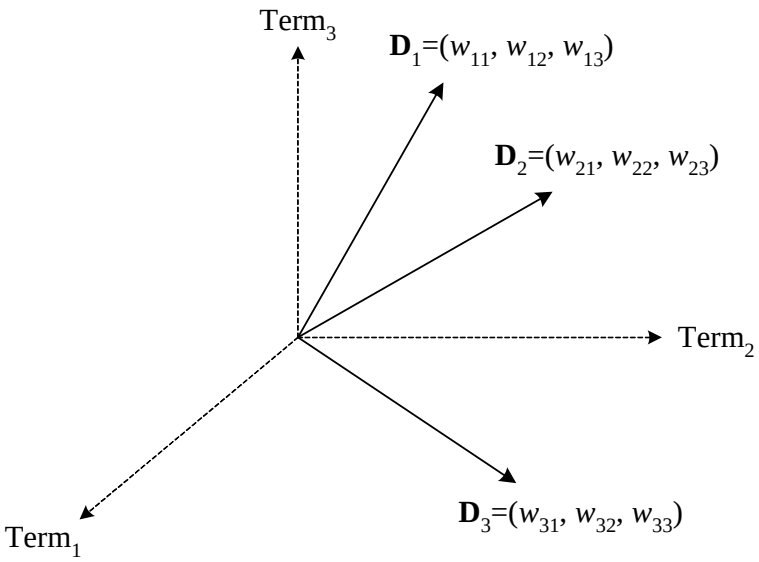

圖 2-3 向量空間法概念圖。

2.3.2 文件的比較

在傳統的向量空間法中，文件的分類是依據文件彼此間的相似度來進行的。建立了文件向量後，可以利用兩文件向量之間的夾角來決定兩份文件的相似度。如果兩文件向量間的夾角越小，表示兩份文件的相似度越大 [5]。爲計算方便，一般並不直接計算兩文件間的真正夾角，而是計算兩文件向量夾角的 cosine 函數值。若將此 cosine 函數值定義爲兩文件的相似度，則

$$\mathrm{SIM}(\mathbf{d}_i, \mathbf{d}_j) = \frac{\mathbf{d}_i \cdot \mathbf{d}_j}{|\mathbf{d}_i||\mathbf{d}_j|} = \frac{\sum_k (w_{ik} \cdot w_{jk})}{\sqrt{\sum_k (w_{ik})^2} \cdot \sqrt{\sum_k (w_{jk})^2}} \tag{2-5}$$

此函數值越大，表示兩向量夾角越小，兩文件的相似度也就越高。如圖 2-4 所示，左圖的兩向量夾角較大，夾角 cosine 值較小，兩向量相似度較低；右圖的兩向量夾角較小，夾角 cosine 值較小，兩向量相似度較高。

一旦可利用適當的數學形式表達出任意兩文件的相似度，就可以分析所有文件中各文件之間的相似度，將相似度高的文件群聚在一起，如此即可達成文件群集的目的。

2.3.3 文件的檢索

Salton 發展向量空間法最主要的目的便是要解決文件檢索的問題。當利用索引詞完成了文件的表述後，只要能夠將文件與使用者的查詢條件進行比較，就能夠進行文件的檢索。爲了讓使用者的查詢條件能與文件進行比較，必須讓查詢條件具備與文件相同的

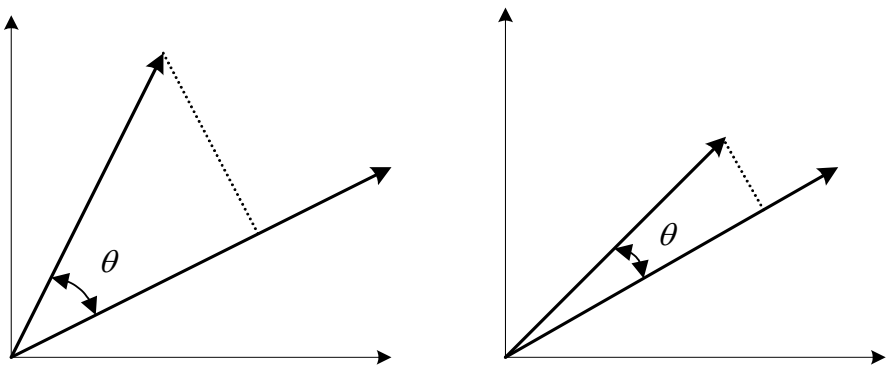

圖 2-4 兩文件向量相似度的比較。

表述形式，也就是與文件向量相同的數學形式。其方法是將查詢條件表述成以索引詞作爲建構元素的向量，該向量的維度是索引詞的數量，與文件向量相同，因此也稱爲虛擬文件 (pseudo-document)。

將查詢條件表述成虛擬文件後，可將虛擬文件的文件向量與真正的文件向量進行相似度的計算。如果設定一門檻值，將相似度高於該門檻值的文件視爲與查詢條件相關，也就是符合使用者所欲查詢的文件。如此即可達成文件檢索的目的。

2.4 潛在語意索引技術

使用向量空間方法進行文件檢索的最大優點在於處理容易，而且可以具體實行。但文件相似與否在於彼此的文意是否相近，這種比較是一種概念化的工作。向量空間法依賴的是索引詞的統計分析，而各種語言文字都具有同義字 (synonymy) 的特質，許多文件可能在表達相同概念時使用的是不同的詞彙，如此會造成單純的索引詞統計無法完全轉換成文意的表述。也就是說，單單考慮文件中出現的索引詞，並不能完全反映出隱含在文件中的語意概念，因此會造成文件檢索上的誤差。

爲了解決此一問題，Dumais 等人提出了潛在語意索引 (latent semantic indexing, LSI) 的方法 [20]。LSI 可說是向量空間法的擴展 [20–27]，其假設文件的關鍵詞之間潛藏著某種結構，可以反映出文件中的語意概念。只要找出這種潛藏的結構，即使文件中並不含有特定的索引詞，還是可以對文件進行概念化的檢索。

2.4.1 LSI 運用的 SVD 技術

LSI 從向量空間法出發，先利用文件中的索引詞建構文件向量，然後運用 singular value decomposition (SVD) 方法 [28] 將文件向量轉換成另一種等價的形式。此種等價的形式與原來構成文件向量的索引詞已無明顯的關聯性，但在數學運算上卻與原來的文件向量相同。假設有 d 份文件、t 個索引詞，利用文件向量，可以將文件與索引詞的關係以一個 $t \times d$ 的矩陣 $\mathbf{X}$ 表示，透過 SVD 的運算，$\mathbf{X}$ 可被拆解 (decomposed) 成三個矩陣的乘積 (如圖 2-5 所示)：

$$\mathbf{X} = \mathbf{T}_0\mathbf{S}_0\mathbf{D}_0^T \tag{2-6}$$

其中 t 是索引詞的數目，d 是文件的數目，m 是矩陣 $\mathbf{X}$ 的 rank 值，$m \le \min(t,d)$。$\mathbf{T}_0$ 與 $\mathbf{D}_0$ ($\mathbf{D}_0^T$ 是 $\mathbf{D}_0$ 的倒置矩陣) 的各欄是正交的 (orthogonal)，而且是單位長度 (unit-length columns)，也就是 orthonormal 矩陣，分別稱爲 left singular vector 矩陣 ($t \times m$) 與 right singular vector 矩陣 ($m \times d$)，$\mathbf{S}_0$ 是個對角的 (diagonal) 矩陣 ($m \times m$)，稱爲 singular value 矩陣。$\mathbf{S}_0$ 只有左上至右下對角線上的元素有值，且其值依欄列值的增加由大而小遞減。

如果將 $\mathbf{S}_0$ 矩陣中高欄列中的值 (也就是較小的值) 忽略，只取前 k 欄列，形成一個新的 k 階矩陣 $\mathbf{S}$，$\mathbf{T}_0$ 與 $\mathbf{D}_0$ 的欄數也從 m 減少爲 k，而形成 $\mathbf{T}$ 與 $\mathbf{D}$ 兩個新的矩陣，計算 $\mathbf{T}$、$\mathbf{S}$ 與 $\mathbf{D}$ 的乘積，可得一新的矩陣 $\hat{\mathbf{X}}$。因爲所忽略的值很小，新的矩陣與原有的矩陣在數學上近似，可得 (如圖 2-6 所示)：

$$\mathbf{X} \approx \hat{\mathbf{X}} = \mathbf{TSD}^T \tag{2-7}$$

其中 t 是索引詞的數目，d 是文件的數目，m 是矩陣 $\hat{\mathbf{X}}$ 的 rank 值，

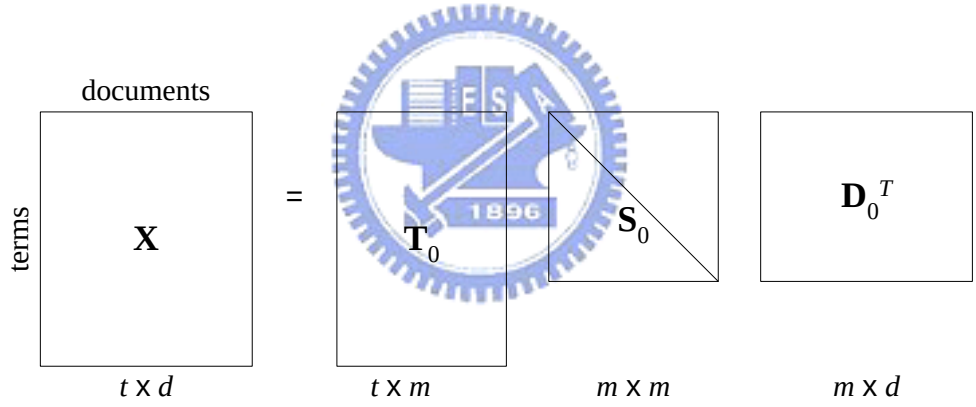

圖 2-5 對一個索引詞—文件向量矩陣進行 SVD 處理。

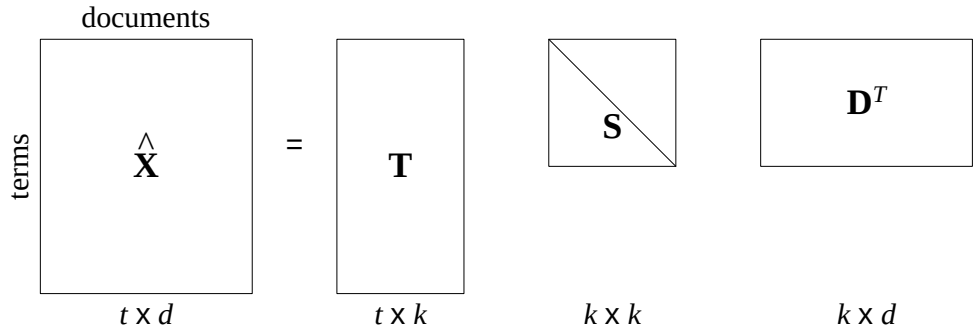

圖 2-6 索引詞—文件向量經過降階 SVD 處理。

$m \le \min(t, d)$，k 則是在降階模式中所選擇的適當維度 (number of dimensions)，$k \le m$。**T** ($t \times k$) 與 **D** ($k \times d$, $\mathbf{D}^T$ 是 **D** 的倒置矩陣) 的各欄是正交的 (orthogonal)，而且是單位長度，也就是 orthonormal 矩陣，**S** 是個對角的 (diagonal) 矩陣 ($k \times k$)，稱爲 singular value 矩陣。**S** 只有左上至右下對角線上的元素有值，且其值依欄列值的增加由大而小遞減。

2.4.2 LSI 的幾何描述

原本 t 維的文件向量經過 SVD 與降階處理後，形成的 left singular vector 與 right singular vector 可分別視爲索引詞與文件在 k 維空間中的另一種表達形式，而對角矩陣 **S** 是該 k 維空間各座標軸的比例因子 (scaling factor)。

LSI 透過這種 SVD 與降階的處理，可以將原本 t 維的文件向量簡化成 k 維，除了可簡化後續文件向量相關的計算外，也將原來與索引詞直接相關的文件向量轉換成與索引詞無直接相關的形式，可用以表達潛藏文件中的語意概念。

2.4.3 運用 LSI 進行文件與索引詞的比較

運用 LSI 技術可以進行文件與索引詞相關的比較，最基本的比較有三種，分別是索引詞與索引詞相關性的比較、文件與文件相似度的比較，以及索引詞與文件之間的比較。以下分別簡要說明。

(1) 索引詞與索引詞相關性的比較

因爲 $\hat{\mathbf{X}}$ 可視爲由索引詞所構成的文件向量，因此 $\hat{\mathbf{X}}$ 的列向量 (row vector) 可視爲各索引詞在所有文件中所表現出的意涵，$\hat{\mathbf{X}}$ 中兩個列向量的內積可用以表達相對應之兩索引詞的相關性。透過 $\hat{\mathbf{X}}$ 與其倒置矩陣的乘積可算出所有列向量的內積，因爲 **S** 是對角矩陣，而 **D** 是正交矩陣，可得：

$$\hat{\mathbf{X}}\hat{\mathbf{X}}^T = \mathbf{TS}^2\mathbf{T}^T = (\mathbf{TS})(\mathbf{TS})^T \tag{2-8}$$

這表示 $\hat{\mathbf{X}}\hat{\mathbf{X}}^T$ 的 (i, j) 元素值 (cell value) 可以由 **TS** 矩陣的第 i 列與第 j 列的內積而得。也就是說，**TS** 矩陣的每一列可以表達不同的索引詞，而兩列的內積結果就表達了兩個索引詞的相關性。另外要注意的是，因爲 **S** 是個對角 (diagonal) 矩陣，因

此 **TS** 矩陣與 **T** 矩陣所分別表達出的索引詞向量的差別僅在於兩者的座標存在 **S** 比例的關係。

(2) 文件與文件相似度的比較

文件與文件相似度的比較和索引詞相關性的比較很類似，差別僅在於用以表達文件意涵的是 $\hat{\mathbf{X}}$ 矩陣中的欄向量 (column vector)，而非列向量。因此 $\hat{\mathbf{X}}$ 中兩個欄向量的內積可用以表達相對應之兩文件的相似度。透過 $\hat{\mathbf{X}}$ 的倒置矩陣與 $\hat{\mathbf{X}}$ 的乘積可算出所有欄向量的內積，因爲 **S** 是對角矩陣，而 **T** 是正交矩陣，可得：

$$\hat{\mathbf{X}}^T\hat{\mathbf{X}} = \mathbf{DS}^2\mathbf{D}^T = (\mathbf{DS})(\mathbf{DS})^T \tag{2-9}$$

這表示 $\hat{\mathbf{X}}^T\hat{\mathbf{X}}$ 的 (i, j) 元素值可以由 **DS** 矩陣的第 i 列與第 j 列的內積而得。也就是說，**DS** 矩陣的每一列可以表達不同的文件，而兩列的內積結果就表達了兩份文件的相似度。另外要注意的是，因爲 **S** 是個對角 (diagonal) 矩陣，因此 **DS** 矩陣與 **D** 矩陣所分別表達出的文件向量的差別僅在於兩者的座標存在 **S** 比例的關係。

(3) 索引詞與文件的比較

索引詞與文件的比較和前面兩者有較大的差異。不同於索引詞間的比較或文件之間的比較分別取決於 $\hat{\mathbf{X}}$ 列向量的內積以及欄向量的內積，欲比較索引詞與文件的相關性，只需要找出 $\hat{\mathbf{X}}$ 矩陣中與該索引詞及文件相對應的元素值即可。譬如要比較第 i 個索引詞與第 j 份文件的相關性，只需要找出 $\hat{\mathbf{X}}$ 矩陣中 (i, j) 元素的值即可。因爲 $\hat{\mathbf{X}}$ 矩陣係由 **T**、**S**、**D** 三個矩陣所定義：

$$\hat{\mathbf{X}} = \mathbf{TSD}^T = \left(\mathbf{TS}^{1/2}\right)\left(\mathbf{DS}^{1/2}\right)^T \tag{2-10}$$

因此 $\hat{\mathbf{X}}$ 矩陣的 (i, j) 元素值可由 $\mathbf{TS}^{1/2}$ 矩陣的第 i 列與 $\mathbf{DS}^{1/2}$ 矩陣的第 j 列的內積而得。

2.4.4 虛擬文件的表述

LSI 的的用途是資訊的檢索，因此它必須提供有效的機制，依據使用者下達的查詢條件，找出相關的文件資料。爲了便於將使用者下達的查詢條件 (query) 與其他文件進

行比較，可將查詢條件以類似 $\mathbf{X}$ 矩陣或 $\hat{\mathbf{X}}$ 矩陣中的文件向量表述成以索引詞爲建構元素的向量，此種向量稱爲虛擬文件 (pseudo-document)。也就是說，虛擬文件的維度是索引詞的數量，每個維度的值是該維度對應的索引詞在查詢條件中出現的頻率。

要進行查詢條件或虛擬文件與其他文件資料的比較，得先將查詢條件表述爲索引詞的向量 $\mathbf{X}_q$，然後推導出另一種表述形式 $\mathbf{D}_q$：

$$\mathbf{D}_q = \mathbf{X}_q^T \mathbf{T} \mathbf{S}^{-1} \tag{2-11}$$

依前節所述，LSI 中文件與文件相似度的比較取決於 $\mathbf{DS}$ 矩陣中列向量的內積，因此若要將查詢條件與文件進行相似度的比較，應該將 $\mathbf{D}_q$ 乘上 $\mathbf{S}$：

$$\mathbf{D}_q \mathbf{S} = \mathbf{X}_q^T \mathbf{T} \mathbf{S}^{-1} \mathbf{S} = \mathbf{X}_q^T \mathbf{T} \tag{2-12}$$

此種表述形式與前述 $\mathbf{DS}$ 矩陣中的各個列向量擁有相同的維度與座標軸，因此可以用於與 $\mathbf{DS}$ 矩陣的列向量進行內積計算，也就是可與各文件進行相似度的比較。如此即可找出能滿足與查詢條件相關的文件。因爲 $\mathbf{S}$ 是個正交 (orthogonal) 矩陣，$\mathbf{S}^{-1}$ 也仍是個正交矩陣，因此 $\mathbf{D}_q$ 矩陣可視爲是 $\mathbf{X}_q^T \mathbf{T}$ 在各個維度上，依 $\mathbf{S}$ 矩陣對角線上的數值而有所伸展或縮短。

2.4.5 LSI 的應用探討

LSI 可說是傳統向量空間法的變形與延伸。透過降階 SVD 的運算處理，將原本數千甚至上萬的維度有效地縮減，可以大幅減少相關 IR 處理所耗費的時間，提升運算與處理的效率。

在使用 LSI 時，降階後的 k 值 (降階後的維度) 如何選擇並沒有一定的規則。k 值如果取得太大，LSI 就與傳統的向量空間法沒有太大差別；k 值如果取得太小，則可能因爲捨棄了過多的資訊而導致明顯的誤差。儘管如此，由許多相關的研究顯示，k 值選爲 100 左右一般皆可得到不錯的結果 [21–26]。

原本 LSI 技術是要處理文件索引的問題，但因爲其轉換出的文件向量可視爲文件內容概念的表達，因此應該非常適合運用在文件的分類上。而這樣的分類法不再直接以關鍵詞來構成文件向量，亦可改善傳統向量空間法的缺點。

2.5 自動化文件分類法

2.5.1 中心向量分類法

以向量空間法的概念而言，兩份文件的相似度越高，兩份文件的文件向量就越接近。一旦可利用適當的數學形式表達出任意兩文件的相似度，就可以分析所有文件中各文件之間的相似度，將相似度高的文件群聚在一起，如此即可達成文件群集的目的。而若從分類的概念來看，相同類別的文件，其相似度應該比較高。因此相同類別的文件，其文件向量會形成一個群集，各個不同的類別就形成各個不同的群集 [3, 5]。圖 2-7 所示即爲文件群集的概念示意圖。圖中所示有六個群集，各個群集內含數份文件，並建構出該群集的中心向量。有些群集有所交集，彼此之間共享文件，意味著有些文件並不只隸屬於單一個群集。

如果如果能掌握已分類好的文件，將隸屬同一類別的文件群集在一起，便能找出其共有的特徵，也就是該分類類別的特徵。此種分類法稱爲 Rocchio 分類法 [6, 7]。要找出同一類別文件共有的特徵，最簡單的方法就是計算該類別中所有文件向量的中心向量 (centroid vector)，然後將待分類的文件與各類別的中心向量相互比較，看看該文件與哪一類別的中心向量相似度最高，即可決定該文件隸屬於哪個類別。因爲使用類別的中心向量作爲該類別的特徵，因此也稱爲中心向量分類法。

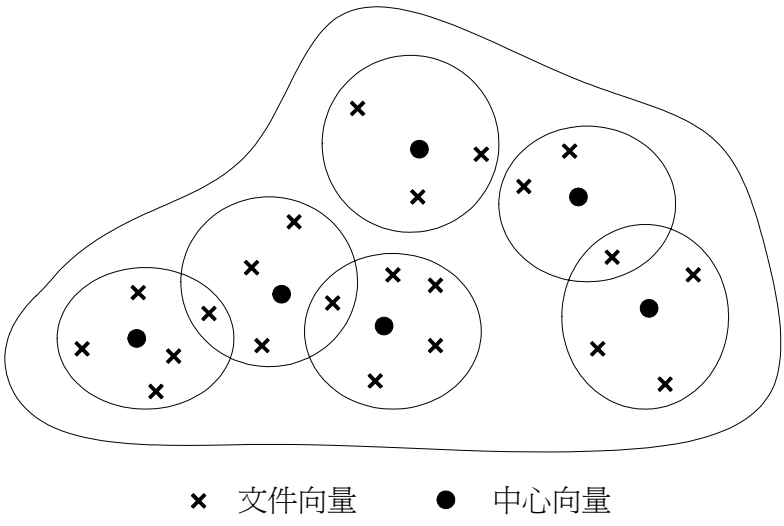

圖 2-7 文件群集的概念示意圖。

使用類別中心向量的分類法，其最大的好處在於概念簡單。而且當各類別的中心向量確定後，待分類的文件只需要與各類別的中心向量比較，實際執行的比較次數與文件類別的數量相同，相較於文件的數量而言，此次數相當少，因此在分類的運算效率上非常高。

利用類別的中心向量進行文件的分類，雖然簡單也很有效率，但通常無法得到較高的分類正確率。主要的原因在於隸屬於同一類別的訓練文件並不全然有良好的群集效果。也就是說，有些訓練文件會偏離其所屬類別的群集，正如同雜訊一般，在計算其所屬類別的中心向量時，反而會影響用以代表該類別的中心向量。

另外還有一個影響中心向量法的因素在於文件並不一定是只能隸屬於單一的類別，在大部分的情況中，文件可以隸屬於多個不同的類別，這使得各個類別的文件無法形成良好的群集效果，因此也無法得到具有類別代表性的中心向量。

2.5.2 *k*-NN 分類法

爲了排除群集效果不佳的訓練文件，避免其影響分類的正確率，有不少改良的分類法被提出，*k*-NN (*k*-nearest neighbor) 便是其中一種 [8]。*k*-NN 法並不利用各類別的特徵來進行分類，而是對每一份測試文件，找出與其相似度最高的 k 份訓練文件。因爲該 k 份文件與測試文件的相似度最高，因此對於測試文件應隸屬於哪個類別具有關鍵性的角色。分析該 k 份文件所屬類別以及與測試文件的相似度，將該相似度視爲一種權重值，並將 k 份文件中同屬相同類別的文件的權重值加總，最後視哪個類別的權重值最高，就將測試文件分類至該類別。

圖 2-8 是 *k*-NN 分類法的示意圖。圖中 A、B、C 代表三個不同類別的文件，圓點則是待分類的測試文件。爲了簡化說明，圖中所示的 *k*-NN 法並不考慮權重的處理，因此簡化成只考慮不同類別的文件數量。圖中最小的圓圈表示 $k = 1$ 的狀況，此時待分類的文件歸屬於 A 類別。中間的圓圈表示 $k = 3$ 的狀況，此時待分類的文件歸屬於 B 類別，因爲與待分類文件最相似的三份文件中，有一份屬 A 類別，而有兩份屬 B 類別。至於最大的圓圈表示 $k = 6$ 的狀況，此時待分類的文件歸屬於 C 類別。因爲與待分類文件最相似的六份文件中，歸屬 C 類別的文件數量最多。

嚴格來說，*k*-NN 法並不需要訓練的階段，因爲只要建立了訓練文件的文件向量後，無需其他額外的處理，就可以進行測試文件的分類。

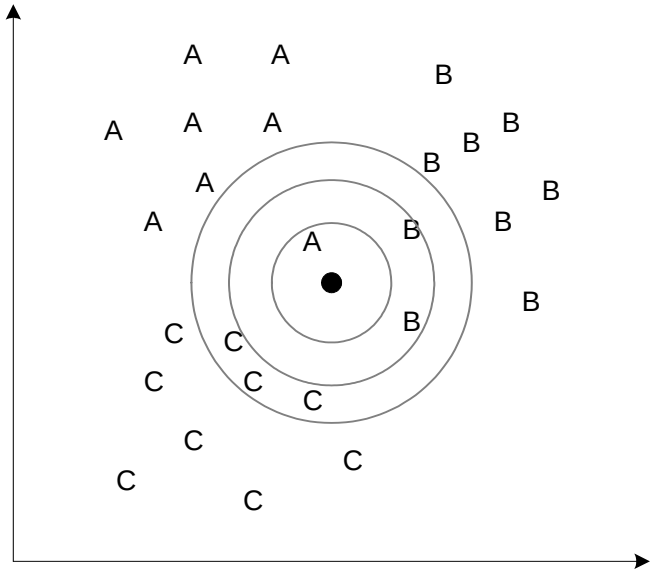

圖 2-8 *k*-NN 分類法的示意圖。

k-NN 法也可以應用於多分類的問題中。也就是每份文件不一定只隸屬單一個類別，而可以同時隸屬多個不同的類別。通常的處理方式是設定一門檻值，將與測試文件最相近的 *k* 份文件經相似度權重值加總計算後，視各類別的權重值是否高於設定的門檻值，如果是的話，便認定該測試文件隸屬該類別。因爲權重值高於設定門檻值的類別可能不只一個，所以同一份測試文件可能同時隸屬多個不同的類別。

使用 *k*-NN 法時，*k* 值的選取並沒有一定的規則。有不少的研究探討如何選取適當的 *k* 值，以獲得最佳的分類效果，但沒有明確的結果。有些研究利用嘗試多個不同的 *k* 值，觀察分類的效果，決定最終使用的 *k* 值 [6, 7]。也有研究針對不同的類別，使用不同的 *k* 值，而非固定的 *k* 值 [29]。也有相關文獻建議使用 cross validation 的方法找出最佳的 *k* 值 [30]。不過一般而言，*k* 值的選取不宜過大，最多爲兩位數。*k* 值很大時，*k* 值的變化對分類的結果影響不大；*k* 值很小時，則可能因爲局部的文件分布不均衡，使得 *k* 值小幅的變動即會造成分類結果有較大的差異。

使用 *k*-NN 法的好處是只取最具關鍵性的部分文件作爲分類的依據，避免群集效果不佳的文件造成干擾，因此通常可以得到較佳的分類正確率。但 *k*-NN 法只考慮與測試文件接近的部分訓練文件，完全忽略了其他訓練文件可能產生的影響。此外，*k*-NN 有運算處理效率上的缺點。因爲在 *k*-NN 法中，待分類的測試文件必須與每一份訓練文件進行比較，且將比較後的結果進行排序，才能找出與其相似度最高的 *k* 份文件，當訓練文件的數量很龐大時，這會是相當耗費時間的工作。

2.5.3 其他分類法

除了前述的中心向量法與 *k*-NN 法外，其他還有許多應用也相當廣泛的分類法，例如決策樹 (decision tree, DTree) 演算法 [9, 11]、運用機率分析的 Naïve Bayes 分類法 [9, 10]、運用回歸分析的線性最小平方根法 (linear least squares fit, LLSF) [14]、運用類神經網路 (neural network, NNet) 的分類法 [12, 13] 、基因演算法 (genetic algorithm) [15]、supporting vector machine [16] 等。這些分類法各有其特色與優缺點，經過許多相關的研究，證實均有不錯的分類效果，也有文獻對這些分類法的分類效果作了相當深入的研究 [5, 6, 31, 32]。

在文件分類的研究中，爲了簡化問題，有些只考慮單一分類的狀況，也就是假設每份文件只隸屬於單一類別。但也有不少研究在探討多重分類的問題。此外，大部分的研究所處理的分類架構都是平坦式 (flat) 的單層架構，但也有些研究探討多層的分類架構 [33–35]。

2.6 運用 LSI 於文件分類的研究

有人曾對運用 LSI 於文件自動化分類進行過研究，但似乎仍缺乏有系統且廣泛的研究。

Dasigi 等人結合 LSI 與類神經網路進行文件分類的研究 [36]，其使用 Reuters-22173 [37] 文件資料集，並選用其階層式分類架構中第一層的八個類別。因爲 Reuters-22173 文件資料集中的每份文件並不全然只屬於單一類別，因此 Dasigi 等人的研究並不侷限於單一分類的問題。不過其研究的重點在於使用「information fusion」，也就是結合不同的文件特徵選取方式，探討其對於文件分類的影響，而非單純地探討將 LSI 運用於文件分類的效果。

Zeliknovitz 等人結合 LSI 與 noisy-or 運算法進行文件分類的研究 [38]，但其研究重點在於探討使用大量背景資訊進行 LSI 處理對於分類效果的影響，亦非單純地探討將 LSI 運用於文件分類的效果。其使用了四種不同的文件資料集。對於只有兩種類別的物理學論文資料，該研究得到高於 90% 的分類正確率；對於有七種類別與動物相關的網頁資料，該研究所得的分類正確率約爲 60%；對於有四種類別與電腦資訊相關的網頁資料，該研究所得的分類正確率隨著訓練文件數量的不同，約爲 40% 到 80%；至

於有 20 種類別的網路新聞群組 (newsgroup) 資料，該研究所得的分類正確率則約爲 40% 到 60%。

這些研究結果雖可提供相當程度的參考，但因爲都不是單純地探討將 LSI 技術運用於自動化文件分類，且缺乏與其他一般分類法的比較。因此不足以充分顯露出運用 LSI 的分類效果。

第三章　研究方法

本研究的目的在探討運用潛在語意索引 (LSI) 的技術進行文件分類的效果。本章將敘述本研究所處理的方法，說明使用的文件資料與文件資料的準備，並將說明相關軟體與研究工具的準備，以及配合使用的分類法。

3.1　研究方法概述

文件分類涉及了兩個部分：文件的表述與分類演算法。本研究探討運用 LSI 技術進行文件分類的效果，因爲 LSI 算是傳統向量空間法的延伸與變形，因此可應用於第一個部分，也就是文件的表述。本研究使用 LSI 技術達成文件的表述，並以傳統向量空間法所進行的文件表述作爲對照，然後配合已經獲得廣泛研究的分類法，進行文件的分類，再比較兩者的分類結果。

在分類法的選擇上，本研究使用了兩種方法。第一種是中心向量法。選擇此方法是因爲其原理簡單，運算快速，可以很快地獲得 LSI 與傳統向量空間法的比較結果。第二種是 *k*-NN 法。*k*-NN 法是種已經經過廣泛研究證明效果不錯的分類法，配合 *k*-NN 法，可以較客觀地評析兩種文件表述應用於自動化文件分類的效果。

3.2　文件資料

3.2.1　Inspec 文件集

本研究使用的文件資料係取用自 Inspec on Disc 中收集整理的摘要資料。Inspec on Disc 收錄了許多與自然科學及工程相關的文獻摘要資料，本研究選用了 1989 年至 1999 年期間電腦科學相關文獻摘要資料 (Abstracts of Computer Science Selections)。本論文將以 Inspec 文件集稱之。在此 Inspec 文件集中，共有 34281 份文件資料。其分類架構爲三階層式，第一階有五個類別，第二階有 43 個類別，其中兩個第二階的類別

又細分爲數個第三階的類別。此 Inspec 文件集的分類架構示意圖如圖 3-1 所示，完整的分類內容與說明則如表 3-1 所示。本研究僅考慮單階層的分類架構，而且使用第二階的分類類別。部分第二階的類別雖又細分爲數個第三階的類別，因爲僅爲少數情況，而非一般性的情況，因此在本研究中，將那些第三階的類別全部併入其所隸屬的第二階類別中，不再予以細分。

因爲本研究在探討運用 LSI 與傳統向量空間法在文件自動分類上的效果，因此爲了簡化所處理的分類問題，僅考慮單一類別的情況，也就是說，只考慮每份文件只能隸屬單一個類別，而不能同時隸屬多個不同的類別。

依此準則，本研究先從 Inspec 文件集中選出只隸屬於單一類別的文件，至於隸屬多個類別的文件則捨棄不用。爲了確保每個類別中有足夠的文件數量，不致因文件數過少而影響分類效果，本研究將所含文件數量不足十篇的類別也捨棄，僅選取至少含有十份文件的類別。經此篩選出的類別共有 38 個，而選用的文件數量爲 13827。所選 38 類別各擁有的文件數量如表 3-2 所示。

原始的 Inspec 文件資料共有 11 個檔案，每一年的資料各爲一個檔案。每份文件除了文件編號以及實際的內容外，還包括了許多輔助性資料，例如文件類型、標題、作者、期刊或研討會名稱、出版者、類別、關鍵字等資訊。本研究先對 Inspec 文件資料進行前處理，將每份文件儲存爲單一的檔案，如此有利於後續各項實驗步驟的進行。另外本研究也將各文件中大部分的輔助性資料移除，只保留文件編號、文件標題與實際的文件內容。文件編號單獨位於文件中的第一行，文件標題位於第二行，文件內容則位於文件標題之後。

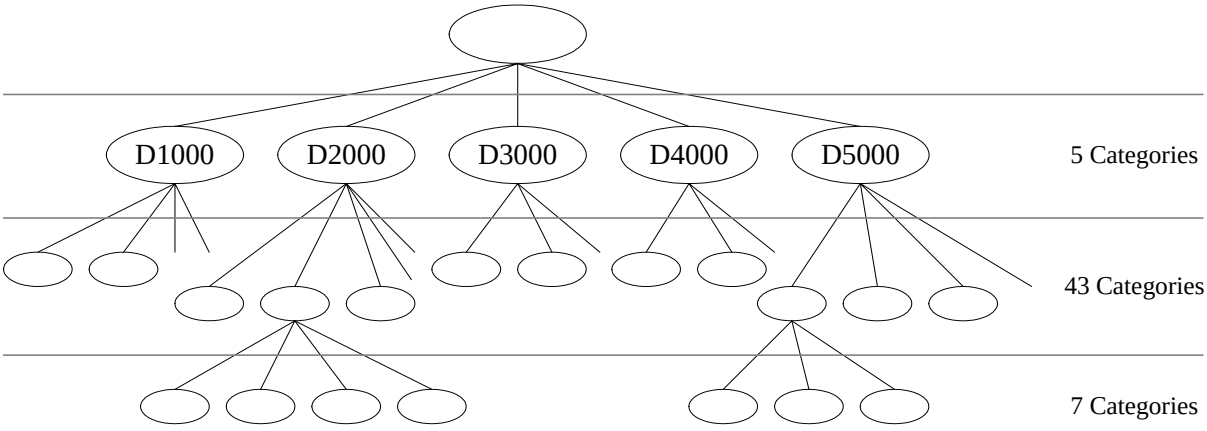

圖 3-1 本研究所用文件資料集 Inspec on Disc 的分類架構示意圖。

表 3-1 Inspec 文件集的分類架構。

類別編號	說明
D1000	General & Management aspects of Information Technology (inc. contracts, planning)
D1010	IT consultancy services
D1030	IT training requirements
D1040	Human aspects of IT (inc. ergonomics, health hazards, home working)
D1050	Legal requirements of IT (inc. liability, regulation, taxation)
D1060	Security aspects of IT (inc. computer crime)
D2000	Applications of Information Technology
D2010	Business and professional IT applications (inc. desktop publishing, expert systems, word processing)
D2020	Design and graphics IT applications
D2030	Education and training IT applications
D2040	Emergency services IT applications
D2045	Farming and horticultural IT applications
D2050	Financial applications of IT
D2050B	IT in accounting
D2050E	IT in banking (inc. smart cards)
D2050F	IT in financial markets (inc. commodities, foreign exchange, stock markets)
D2050G	IT in insurance
D2060	Health care applications of IT
D2070	Industrial and manufacturing applications of IT (inc. CIM, FMS, MAP)
D2080	Information services and database systems in IT
D2090	Leisure industry, travel and transport applications of IT
D2105	Media-TV, radio, press applications of IT
D2110	Personnel applications of IT
D2115	Property market and building industry applications of IT
D2120	Public administration and law applications of IT (inc. government)
D2130	Public utilities' applications of IT (inc. electricity, gas and water suppliers)
D2140	Marketing, retailing and distribution applications of IT (inc. EFTPOS, smart cards)

表 3-1 (續) Inspec 文件集的分類架構。

類別編號	說明
D3000	General Information Technology systems and equipment
D3010	Copiers and copy boards for business automation
D3020	Furniture and office environment for business automation
D3025	Mailroom systems for business automation
D3030	Microform equipment for business automation
D3035	Monitoring and alarm systems for business automation
D3045	Records management systems for business automation (inc. document image processing, shredders)
D3050	Video systems for business automation
D3060	Voice equipment, dictation for business automation
D4000	Office automation-communications (inc. general communication industry topics, telecommunication networks)
D4010	Television systems for office automation
D4020	Electronic mail systems for office automation
D4040	Facsimile systems for office automation
D4045	Mobile communications systems for office automation (inc. cellular radio and telephones, radiopaging)
D4060	Teleconferencing systems for office automation
D4070	Telephone systems for office automation
D4080	Telex for office automation
D4090	Viewdata and teletext for office automation (inc. ideotext)
D5000	Office automation – computing (inc. general computing industry topics)
D5010	Computers and work stations for office automation
D5010B	Portable computers for office automation
D5010D	Computer selection guides for office automation
D5010G	Terminals for office automation
D5020	Computer networks and intercomputer communications in office automation
D5030	Printers and other peripherals for office automation
D5040	Supplies, stationery and storage media for office automation
D5050	Word processing equipment for office automation (inc. desktop publishing)

表 3-2 Inspec 文件集所選 38 類別各擁有的文件數量。

分類號	文件數	分類號	文件數	分類號	文件數	分類號	文件數	分類號	文件數
D1010	20	D2050	3650	D2120	161	D3045	383	D4070	277
D1030	62	D2060	489	D2130	40	D3050	15	D5010	1429
D1040	174	D2070	494	D2140	919	D3060	75	D5020	902
D1050	35	D2080	444	D3010	273	D4010	16	D5030	496
D1060	641	D2090	149	D3020	200	D4020	75	D5040	190
D2010	670	D2105	184	D3025	66	D4040	262	D5050	115
D2020	274	D2110	96	D3030	255	D4045	79		
D2030	105	D2115	75	D3035	144	D4060	123		

經前處理後的文件，內含字數最多的文件含有 304 個字，最少的則含有 12 個字。每份文件平均含有 81.6 個字。

本研究依循傳統 IR 的方法，將文件區分成訓練文件 (training documents) 與測試文件 (testing documents)，利用訓練文件找出各類別文件的特徵與分類演算法相關的參數，然後將建構好的分類法用於測試文件，比較其測試後的分類類別與原本正確的分類類別，經統計後計算分類的正確率。本研究係以隨機方式，選取各類別中 80% 的文件 (共有 11075 份文件) 作爲訓練文件，將剩餘的 20% 文件 (共有 2752 份文件) 作爲測試文件。爲求研究的結果具有統計上更高的可性度，本研究對相同的文件集選取了兩組不同的分割方式，也就是兩組不同的訓練與測試文件，分別進行相同的分類處理，並比較兩者之間的差異。

3.2.2 字根的處理

以英文的語言特性而言，視使用的文意情境，許多字都有不同的變形，如單複數、形容詞與副詞等不同詞性，或是動詞的不同時態等。在傳統 IR 的研究中，爲了更精確掌握不同變形的字詞所具有相同的文意，通常會先對字詞進行字根的處理，也就是說，將具有相同字根 (stem) 的字視爲相同的字。

在 Dumais 等人關發表的 LSI 首篇文獻中，作者表示 LSI 並不直接使用字詞的外在形式，而是掌握字詞的內涵，可以發掘文件中潛在的語意，因此利用 LSI 處理的文件，可以無需進行字根的處理。但在其他 LSI 相關的研究中，似乎對此問題進行過深

入的探討，且大部分的 LSI 研究仍會適度地使用字根演算法。

本研究的目的在比較 LSI 法與傳統向量空間法對於文件分類的效果，傳統向量空間法直接利用字詞的外在形式，因此字根的處理有其必要性。爲了兼顧傳統向量空間法語 LSI 兩者的特性與其對於字根處理的一般原則，本研究對於文件的剖析都進行字根的處理。

本研究使用的字根處理方法是 M. F. Porter 所提出的「字尾移除演算法」(algorithm for suffix stripping) [39]，也就是一般 IR 領域所熟知的 Porter 字根演算法 (Porter's stemming algorithm)。可進行 Porter 字根處理的程式一般也稱爲 Porter's stemmer。本研究使用了官方網站所提供的 Perl 版本軟體 [40]，進行文件字根的前置處理。

3.3 軟硬體說明

3.3.1 LSI 軟體

LSI 技術的核心是 SVD 的相關運算，雖然已有許多數學相關的套裝軟體提供 SVD 的運算功能，網路上也有免費的 SVD 軟體可供下載使用，但對於處理文字與文件的 LSI 而言，仍須額外撰寫不少程式才足供使用。

本研究透過網際網路，尋找到一套可完全提供 LSI 相關運算與處理功能的軟體：General Text Parser (GTP) [41]。該軟體係由美國田納西大學計算機科學系 (Department of Computer Science, University of Tennessee) 的 Michael W. Berry 教授等人所開發，雖爲免費提供，但並非於網路上公開下載。本研究經 Berry 教授的同意與授權，取得了 GTP 原始程式碼。

GTP 係以 C++ 語言寫成，提供了對文件集一般性的剖析 (parse)，也能提供 IR 應用所需的矩陣 decomposition。GTP 提供了傳統向量空間法對於文件的剖析功能，可以將文件中全部的索引詞擷取出來。GTP 不只能處理單一的檔案，也能對整個目錄所有的檔案同時處理，包括各層子目錄在內。GTP 軟體附了一個 common_words 檔案，內含許多常見字，也就是虛字 (stop word)，使用者可選擇性地決定是否使用該檔案將常用字排除在擷取的索引詞之外。

GTP 對文件進行剖析時，會將擷取出的索引詞以及其在該文件中出現的頻率寫入

資料檔中。使用者可以選擇性地設定索引詞的 local 與 global 權重形式，其中 local 權重形式有 tf (term frequency)、log 及 binary 三種，global 權重形式則可以是 normal、idf、idf2 或 entropy。預設的 local 權重形式是 tf，且不使用 global 權重。本研究依循傳統 IR 的處理方式，選擇 tf 作爲 local 權重形式，而以 idf 作爲 global 權重形式。

GTP 擷取出索引詞及其出現頻率後，會根據使用者設定的 local 及 global 權重形式，計算最終的索引詞權重值。至此，GTP 即可以各索引詞爲不同的維度，決定出該文件的文件向量，並可將結果寫入資料檔中。

經由使用者的指示，GTP 也可以將產生的文件向量正規化 (normalization)，亦即將每份文件的文件向量調整成單位長度。

當 GTP 對文件進行剖析時，使用者可以設定額外的過濾器 (filter)，將文件先以設定的過濾器處理，再交由 GTP 進行剖析。使用者如果需要進行字根的處理，便可將前述的 Porter's stemmer 作爲額外的過濾器，對文件進行前置處理。

GTP 剖析完文件，產生了文件—索引詞向量後，可以對該向量的矩陣進行 SVD 處理，產生前一章所述的 **T**、**S** 與 **D** 矩陣。因爲 **S** 矩陣是個對角矩陣，假設降階後該矩陣爲 k 階，**S** 矩陣其實是個內含 k 個 singular values 的檔案，每個 singular value 一行，共有 k 行。另外 GTP 會產生兩個檔案：lalv2 與 larv2，lalv2 對應的是 **T** 矩陣，larv2 則對應 **D** 矩陣。lalv2 中包含了組成 **T** 矩陣的向量，larv2 中包含了組成 **D** 矩陣的向量，各向量係依其對應之升冪的 singular values 依序寫入檔案中，也就是說，對應小的 singular value 的向量寫在前面，對應到大的 singular value 的向量則寫在後面。兩個檔案都是 binary 格式，所有存在檔案中的值都是 double 型別。

本研究取得的 GTP 軟體爲 Sun Solaris 版本，可在 Solaris 2.6 版的系統上以 GNU C/C++ 編譯器順利編譯並執行。

3.3.2 其他軟硬體環境

爲了將利用訓練文件所產生的所有實驗數據有效儲存，以利後續測試文件分類研究，以及各項統計之用，本研究使用了 SQL 資料庫儲存各項實驗數據。本研究使用的 SQL 資料庫系統爲開放程式碼 (open source) 的 MySQL [42]，版本爲 3.1.16。

本研究所有的程式都以 Perl 撰寫 [43]。Perl 是一種直譯式命令稿語言 (interpret script language)，雖然在執行效率上不及 C 一類的編譯式語言，但並不會差距太大。而

Perl 對於字串型態的資料有很強的處理能力，也具有很大的處理彈性，在文件資料的處理上相當便利。此外 Perl 有豐富的模組，包括高階的資料庫存取介面 (database interface, DBI)，可以很便利地與 MySQL 溝通。

本研究取得的 GTP 軟體爲 Sun Solaris 的版本，因此文件的前處理與 GTP 的執行，都是在 Sun 的機器上進行，型別是 Ultra 10 Model 300，使用的作業系統爲 SunOS 5.6，也就是 Solaris 6，CPU 爲一顆 299 MHz SUNW UltraSPARC-Iii，配備 768 MB 記憶體。至於實際的文件分類訓練與測試，則都是在 IBM eServer xSeries 205 電腦上進行，使用的作業系統是 RedHat Linux 9.0，系統核心 (kernel) 爲 2.4.20-8 版，CPU 爲一顆 2.8 GHz Intel Pentium 4，配備 1 GB 記憶體。

3.4 向量正規化與 S 矩陣的探討

3.4.1 文件向量的正規化

以傳統向量空間法建構文件向量時，文件向量各維度的值是依對應的索引詞在該文件中出現的頻率，加上適當的權重處理而成。因爲各文件的長度 (所含字詞數量) 不同，有些差距很大，使得各文件向量的長度也就不盡相同。

如果要比較文件與文件之間的相似度，或是比較查詢條件 (虛擬文件) 與各文件之間的相似度，所需計算的是向量與向量之間夾角的 cosine 值，此值與向量間的夾角有關，而與向量的長度無關。參見公式 2-5，不論兩向量的長度爲何，在計算兩向量間夾角的 cosine 值時，都會除以兩向量的長度，因此爲了計算與處理方便，通常都會對文件向量進行正規化 (normalization) 的處理，也就是將每個文件向量的長度都調整爲單位長度。

但在中心向量法的分類處理中，各類別的中心向量係將該類別中全部的文件向量加總而成，而向量相加的結果不僅與向量間的夾角有關，也與向量的長度有關。如圖 3-2 所示，兩向量未經正規化處理與經正規化處理後，其相加的結果是不相同的。左圖所示爲兩個經過正規化的向量，皆爲單位長度；右圖則是兩向量未經正規化的狀態，一個大於單位長度，另一個小於單位長度。由圖中可看出，兩組向量加總之後所得的向量並不相同。

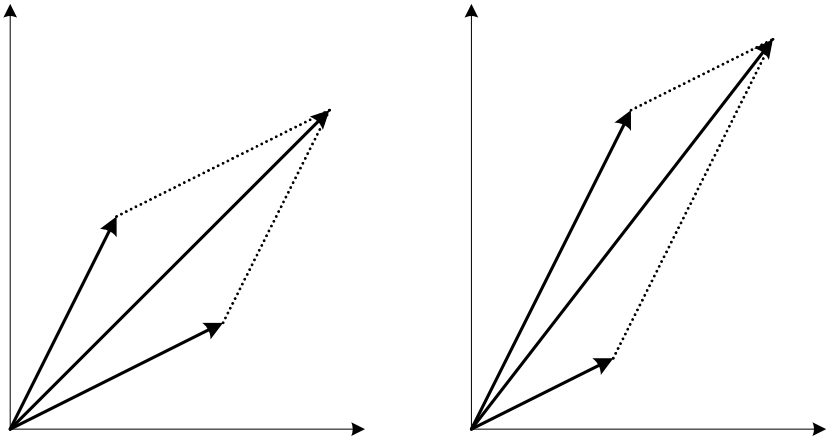

圖 3-2 向量正規化與否對於向量加總的影響。

因爲中心向量法爲本研究使用的分類法之一，因此本研究也探討文件向量正規化與否對於文件分類的影響。

對於傳統向量空間法，會先產生文件向量，然後依不同類別求出其中心向量，也就是計算該類別中全部文件向量的總和。在計算中心向量之前，各文件向量是否進行正規化的處理，對於所得的中心向量將有直接的影響。本研究將針對此兩種情況分別加以探討。

至於 LSI 所面臨的情況就更複雜了。因爲 LSI 會對原始的文件向量進行 SVD 降階處理，進行 SVD 降階處理前的文件向量是否需要正規化，對後續的分類結果可能產生影響。文件向量經過 SVD 降階處理後，LSI 所運用的文件向量是 **DS** 矩陣中的列向量，因此計算中心向量時，係取 **DS** 矩陣中的列向量加以計算。因此是否先將 **DS** 矩陣中的列向量正規化，也是需要探討的問題。

不論使用傳統向量空間法或 LSI 法，在訓練階段計算出來的類別中心向量，在測試階段只需要與測試文件進行相似度的比較，因此中心向量是否正規化並不會影響分類的結果。在本研究中，中心向量都會進行正規化的處理。

至於使用 *k*-NN 法時，因爲測試文件必須與每一份訓練文件進行相似度的比較，其中僅涉及文件向量間的夾角，與其長度無關。因此就傳統向量空間法而言，各文件向量是否正規化並不會影響分類的結果。在本研究中，各文件向量都會進行正規化的處理。但對於 LSI 而言，測試文件進行 SVD 降階處理前是否先進行正規化的處理，似乎可能影響後續分類的結果，因此本研究將分別探討。至於 SVD 降階處理後實際用於分類

演算法的文件向量，本研究都會進行正規化的處理。

3.4.2 LSI 中測試文件的處理

使用 LSI 技術處理完訓練文件後，在分類測試的階段中，必須將測試文件轉換成可以和其他訓練文件進行相似度比較的形式，也就是 **DS** 矩陣的形式。可以將測試文件視爲是一組查詢條件，將測試文件先以傳統向量空間法表達成索引詞構成的向量 $\mathbf{X}_{\text{test}}$，然後利用公式 (2-12) 轉換成 **DS** 矩陣的形式：

$$\mathbf{D}_{\text{test}}\mathbf{S} = \mathbf{X}_{\text{test}}^{T}\mathbf{T}\mathbf{S}^{-1}\mathbf{S} = \mathbf{X}_{\text{test}}^{T}\mathbf{T} \tag{3-1}$$

經過了這樣的轉換，不管是在中心向量法或 *k*-NN 法的分類處理中，都可以將測試文件與各訓練文件進行必要的相似度比較或相關的運算處理。

3.4.3 SVD 運算中的 S 矩陣

在 LSI 的概念中，文件與文件的相似度原本係由 **X** 矩陣中與文件對應的兩組列向量內積而決定，經過 SVD 運算的降階處理後，轉換成了 **DS** 矩陣中與文件對應的兩組向量內積。因此可視 **DS** 矩陣中的列向量爲轉換後的文件向量。但因爲 **S** 是個 orthonormal 矩陣，因此若將 **S** 矩陣視爲一個座標轉換的機制，座標中各個維度隨著 S 矩陣中對應的值而在座標尺度上伸展或短縮，但只是座標尺度上的變化，各維度座標軸的方位並沒有改變。由此可知，**DS** 矩陣中的列向量可視爲是 **D** 矩陣中列向量經過 S 矩陣的座標轉換所得的結果，因爲此座標轉換的特性，**D** 矩陣就某種程度的意義而言，已足以代表 SVD 降階處理後的文件向量。

爲了瞭解在 LSI 方法中，**DS** 矩陣與 **D** 矩陣何者應用於文件分類有較佳的效果，本研究分別對兩者進行實驗，並探討其對於分類結果的影響。

同樣的，在處理測試文件時，對應於訓練文件以 **DS** 矩陣的形式表述，測試文件亦轉換以 $\mathbf{D}_{\text{test}}\mathbf{S}$ 形式表述，也就是 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 的形式。但如果相訓練文件直接以 **S** 矩陣表述，則測試文件亦應以 $\mathbf{D}_{\text{test}}$ 形式表述，也就是 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}\mathbf{S}^{-1}$ 的形式。因此本研究除了探討 **DS** 矩陣與 **D** 矩陣何者對文件分類有較佳的效果外，也會探討對應於兩者的 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 與 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}\mathbf{S}^{-1}$ 對分類結果的影響。

3.5 實驗步驟

3.5.1 實驗步驟概述

本研究所進行的實驗係以下列步驟進行：

(1) 文件資料的整理與前處理。
(2) 選取訓練文件與測試文件。
(3) 對訓練文件進行 LSI 的 SVD 處理。
(4) 以傳統向量空間法及 LSI 分別建立的文件向量，配合中心向量法及 *k*-NN 法進行文件的分類訓練。
(5) 利用中心向量法與 *k*-NN 法，分別對測試文件進行分類。
(6) 對所得分類結果進行正確率與執行效率的比較。

以下各小節將對上述實驗步驟作詳細的說明。

3.5.2 文件資料的整理與前處理

此步驟先將 Inspec 文件集依年份組成的 11 個檔案，撰寫程式加以拆解成每份文件一個檔案，共有 34281 個檔案。同時將文件中不會使用到的資訊移除，包括文件型態、作者、出版時間、期刊或會議名稱等，只保留文件編號、文件標題與文件內容。

然後依 3.2 節所描述的準則，挑選出只隸屬於單一類別的文件，再將所包含的文件數不到 10 份的類別及其所含文件刪除，至此共剩餘 38 個類別、13827 份文件。這些是後續研究所會使用到的全部文件。

3.5.3 選取訓練文件與測試文件

以隨機方式，選取各類別中 80% 的文件作爲訓練文件，將剩餘的 20% 文件作爲測試文件。依此方式選取，訓練文件共有 11075 份，測試文件則有 2752 份。

本研究爲了增加統計上的資料與可信度，分別選取了兩組不同的訓練文件與測試文件，並分別進行完全相同的分類實驗。此兩組文件的選取分別稱爲文件分割 A 與文件分割 B。

爲了方便後續研究的進行，訓練文件全部存放在一個目錄中，測試文件則全部存放在另一個目錄中。相關的文件資訊，如文件編號、文件所屬類別、文件爲訓練文件或測試文件等，全部儲存至關聯式資料庫系統中。

3.5.4 對訓練文件進行 LSI 的 SVD 處理

選取了訓練文件後，接下來便利用 GTP (General Text Parse) 軟體對訓練文件進行 LSI 的 SVD 處理。

在執行 GTP 時，使用了該軟體所附的 common_words 檔案，將文件中的虛字 (非索引詞) 移除，而在索引詞權重的處理上，則使用 tf 作爲 local 權重形式，使用 idf 作爲 global 權重形式。另外也使用了 Poter's stemmer 做爲外部的過濾器，對訓練文件進行字根的處理。執行 GTP 後，經 SVD 降階處理後所得的矩陣階數 k 值，皆取 100。

依據 3.4.1 小節所討論的文件向量正規化的處理方式，本研究對訓練文件分別進行了兩次的 SVD 處理，也就是執行了兩次 GTP，其中一次會對未進行 SVD 降階處理前的原始文件向量加以正規化，另外一次則不進行正規化的處理。如此可得到兩組 GTP 的執行結果。分別將兩組結果所得的 singular value、left singular vector 與 right singular vector 全部儲存在關聯式資料庫系統中，供後續研究使用。

對文件向量執行 GTP 後，除了可以得到 SVD 降階處理後的各矩陣資料，還可以得到未降階前的原始文件向量，此文件向量代表了傳統向量空間法所得的文件向量。將此文件向量資料儲存在關聯式資料庫系統中，可供本研究對照組 (向量空間法) 之用。

3.5.5 分類的訓練

利用前一步驟以 LSI 技術及傳統向量空間法分別建立的文件向量，配合中心向量法及 k-NN 法進行文件的分類訓練。

(1) 中心向量法

針對每個類別，將該類別所含全部訓練文件的文件向量加總，再除以該類別所含訓練文件的數量，計算出該類別所有訓練文件的中心向量。因爲本研究使用的文件分別隸屬於 38 個類別，因此計算所得的中心向量也有 38 個。

(2) *k*-NN 法

k-NN 法：因爲 *k*-NN 法係直接使用訓練文件的文件向量，因此產生了訓練文件的文件向量後，在分類的訓練階段便無需再進行額外的處理。

3.5.6 測試文件的分類

利用前一步驟所得的訓練結果，分別配合中心向量法與 *k*-NN 法，進行測試文件的分類。

(1) 中心向量分類法

將測試文件的文件向量逐一與各類別的中心向量比較，計算兩者的相似度 (向量夾角的 cosine 值)，視測試文件與哪一個類別的中心向量相似度最高，即判定該測試文件隸屬於哪一個類別。

(2) *k*-NN 分類法

本研究主要在比較傳統向量法與 LSI 在文件分類上的差異，而非探討 *k*-NN 分類法的效能，因此並不試圖尋找 *k*-NN 法中最佳的 *k* 值。本研究分別選用 30 與 10 作爲 *k*-NN 法中的 *k* 值，也就是說，將 *k* 值設定爲 30 及 10。將測試文件的文件向量逐一與各訓練文件的文件向量進行比較，計算兩者的相似度 (向量夾角的 cosine 值)。對每一份測試文件而言，可得到 11075 個比較結果，將這些結果排序，分別取相似度最高的 30 及 10 個結果。針對此 30 及 10 個個計算所得的相似度，檢查其對應的各個訓練文件，如果有屬於相同類別的話，就將其相似度相加。最後視哪個類別所得的相似度最高，就判定測試文件隸屬於該類別。

3.5.7 中心向量分類法的實驗變因設計

根據 3.4 節所描述的研究考量因素，包括了相關向量的正規化問題，以及 **DS** 矩陣與 **D** 矩陣的取捨，本研究設計了多種控制變因的方式，以不同的組合進行分類的訓練與測試，以便比較在各種不同變因下所的的分類結果。以下即說明以中心向量法作爲分類法時，與 LSI 搭配的各種變因組合。

(1) LSI 與中心向量法變因組合 1

- 執行 GTP 時對原始的 (未降階前的) 文件向量進行正規化處理。
- 以 **DS** 矩陣的列向量作爲訓練文件的文件向量。
- 以 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 矩陣的列向量作爲測試文件的文件向量。
- 對 **DS** 矩陣中的列向量進行正規化處理。

(2) LSI 與中心向量法變因組合 2

- 執行 GTP 時對原始的 (未降階前的) 文件向量進行正規化處理。
- 以 **DS** 矩陣的列向量作爲訓練文件的文件向量。
- 以 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 矩陣的列向量作爲測試文件的文件向量。
- 對 **DS** 矩陣中的列向量不進行正規化處理。

(3) LSI 與中心向量法變因組合 3

- 執行 GTP 時對原始的 (未降階前的) 文件向量不進行正規化處理。
- 以 **DS** 矩陣的列向量作爲訓練文件的文件向量。
- 以 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 矩陣的列向量作爲測試文件的文件向量。
- 對 **DS** 矩陣中的列向量進行正規化處理。

(4) LSI 與中心向量法變因組合 4

- 執行 GTP 時對原始的 (未降階前的) 文件向量不進行正規化處理。
- 以 **DS** 矩陣的列向量作爲訓練文件的文件向量。
- 以 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 矩陣的列向量作爲測試文件的文件向量。
- 對 **DS** 矩陣中的列向量不進行正規化處理。

(5) LSI 與中心向量法變因組合 5

- 執行 GTP 時對原始的 (未降階前的) 文件向量進行正規化處理。
- 以 **D** 矩陣的列向量作爲訓練文件的文件向量。
- 以 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 矩陣的列向量作爲測試文件的文件向量。

- 對 **D** 矩陣中的列向量進行正規化處理。

(6) LSI 與中心向量法變因組合 6

- 執行 GTP 時對原始的 (未降階前的) 文件向量進行正規化處理。
- 以 **D** 矩陣的列向量作爲訓練文件的文件向量。
- 以 $\mathbf{X}_{\text{test}}^{T}\mathbf{TS}^{-1}$ 矩陣的列向量作爲測試文件的文件向量。
- 對 **DS** 矩陣中的列向量進行正規化處理。

(7) LSI 與中心向量法變因組合 7

- 執行 GTP 時對原始的 (未降階前的) 文件向量進行正規化處理。
- 以 **D** 矩陣的列向量作爲訓練文件的文件向量。
- 以 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 矩陣的列向量作爲測試文件的文件向量。
- 對 **D** 矩陣中的列向量不進行正規化處理。

(8) LSI 與中心向量法變因組合 8

- 執行 GTP 時對原始的 (未降階前的) 文件向量進行正規化處理。
- 以 **D** 矩陣的列向量作爲訓練文件的文件向量。
- 以 $\mathbf{X}_{\text{test}}^{T}\mathbf{TS}^{-1}$ 矩陣的列向量作爲測試文件的文件向量。
- 對 **D** 矩陣中的列向量不進行正規化處理。

(9) LSI 與中心向量法變因組合 9

- 執行 GTP 時對原始的 (未降階前的) 文件向量不進行正規化處理。
- 以 **D** 矩陣的列向量作爲訓練文件的文件向量。
- 以 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 矩陣的列向量作爲測試文件的文件向量。
- 對 **D** 矩陣中的列向量進行正規化處理。

(10) LSI 與中心向量法變因組合 10

- 執行 GTP 時對原始的 (未降階前的) 文件向量不進行正規化處理。

- 以 **D** 矩陣的列向量作爲訓練文件的文件向量。
- 以 $\mathbf{X}_{\text{test}}^{T}\mathbf{TS}^{-1}$ 矩陣的列向量作爲測試文件的文件向量。
- 對 **DS** 矩陣中的列向量進行正規化處理。

(11) LSI 與中心向量法變因組合 11

- 執行 GTP 時對原始的 (未降階前的) 文件向量不進行正規化處理。
- 以 **D** 矩陣的列向量作爲訓練文件的文件向量。
- 以 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 矩陣的列向量作爲測試文件的文件向量。
- 對 **D** 矩陣中的列向量不進行正規化處理。

(12) LSI 與中心向量法變因組合 12

- 執行 GTP 時對原始的 (未降階前的) 文件向量不進行正規化處理。
- 以 **D** 矩陣的列向量作爲訓練文件的文件向量。
- 以 $\mathbf{X}_{\text{test}}^{T}\mathbf{TS}^{-1}$ 矩陣的列向量作爲測試文件的文件向量。
- 對 **D** 矩陣中的列向量不進行正規化處理。

上述 12 種變因組合可以表 3-3 簡要表示。

表 3-3 LSI 搭配中心向量法的實驗變因組合。

變因 \ 組合	1	2	3	4	5	6	7	8	9	10	11	12
執行 GTP 時對原始文件向量進行正規化	✓	✓			✓	✓	✓	✓				
使用 **DS** 計算分類向量	✓	✓	✓	✓								
使用 **D** 計算分類向量					✓	✓	✓	✓	✓	✓	✓	✓
對 **DS** 或 **D** 進行正規化	✓		✓		✓	✓			✓	✓		
使用 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 處理測試文件	✓	✓	✓	✓	✓		✓		✓		✓	
使用 $\mathbf{X}_{\text{test}}^{T}\mathbf{TS}^{-1}$ 處理測試文件						✓		✓		✓		✓

至於在對照組傳統向量空間法的實驗中，根據 3.4 節的描述，只有兩組的變因組合：

(1) 傳統向量空間法與中心向量法變因組合 1

- 執行 GTP 時對原始的 (未降階前的) 文件向量進行正規化處理，如此可得到可供向量空間法使用而已正規化的文件向量。

(2) 傳統向量空間法與中心向量法變因組合 2

- 執行 GTP 時對原始的 (未降階前的) 文件向量不進行正規化處理，如此可得到可供向量空間法使用而未正規化的文件向量。

3.5.8 *k*-NN 分類法的實驗變因設計

至於在 *k*-NN 法的分類實驗中，各項所需實驗資料則取自中心向量法所產生的數據，並不需要進行額外的處理。

但在 *k*-NN 法中，測試文件直接與各訓練文件進行相似度的比較，因此沒有中心向量法中文件向量正規化與否對向量加總所造成的問題，所以在實驗變因的設計中，不需要考慮 **DS** 或 **D** 矩陣中列向量的正規化問題。在本研究中，處理 *k*-NN 法時，都會先對 **DS** 或 **D** 矩陣中的列向量進行正規化的處理。

表 3-4 LSI 搭配 *k*-NN 法的實驗變因組合。

變因 \ 組合	1	2	3	4	5	6	7	8	9	10	11	12
執行 GTP 時對原始文件向量進行正規化	✓	–		–	✓	✓	–	–			–	–
使用 **DS** 計算分類向量	✓	–	✓	–			–	–			–	–
使用 **D** 計算分類向量		–		–	✓	✓	–	–	✓	✓	–	–
使用 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 處理測試文件	✓	–	✓	–	✓		–	–	✓		–	–
使用 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}\mathbf{S}^{-1}$ 處理測試文件		–		–		✓	–	–		✓	–	–

由上述說明，在 *k*-NN 法中的實驗變因組合僅爲中心向量法的一半，其各種變因組合可以表 3-4 簡要說明。真正需要探討的只有 1、3、5、6、9、10 等變因組合。

至於使用傳統向量空間法搭配 *k*-NN 法時，因爲沒有向量加總時向量正規化的問題，因此原始的文件向量是否進行正規化對分類結果並無影響。加上不需要考慮 S 矩陣的影響，所以相較於搭配中心向量法時的兩組實驗變因組合，此時只有一組的變因組合，就是執行 GTP 時對原始文件向量進行正規化處理。

3.5.9 實驗結果的評估

依據前述步驟與方法，可以分別得到 2075 份測試文件的分類結果，也就是說，各測試文件一定會被判定隸屬於某一個類別。將此分類的測試結果與該測試文件真正的類別相比較，可知該測試文件的分類測試結果正確與否。

對於每種實驗變因的組合，計算分類正確的測試文件數與全部測試文件數的比例，亦即分類的正確率，即爲本研究用以評估分類效果的依據。

除了評估分類的正確率外，本研究也將比較各種實驗變因組合所花費的時間，也就是每一份測試文件進行分類所需的時間，據此評估分類法的效率。

第四章　研究結果與討論

本章針對前一章所提之研究方法與執行步驟所進行之自動化文件分類的實驗，分別依據兩組文件分割方式，以及傳統向量空間法、LSI 技術與中心向量分類法、*k*-NN 分類法的組合，整理各項實驗數據與結果。並對所得結果進行討論，包括研究所用方法的優缺點與可行性。

4.1 分類正確率

4.1.1 利用中心向量法的分類正確率

表 4-1 列出了運用 LSI 結合中心向量法所進行之分類實驗的結果，表中除了顯示 3.5.7 節所述各種實驗變因組合所的結果外，也同時列出了 A 與 B 兩組文件分割的結果。表中所示的分類結果都是如 3.5.9 節所述的分類正確率。

從 LSI 結合中心向量法的結果中可看出，實驗變因組合 5、7、9、11 所得的分類正確率較其他變因組合明顯較佳，約高出 8% 至 9%。配合表 3-3 所示的變因組合，可以觀察到以 **D** 矩陣的列向量作爲訓練文件的文件向量，計算各類別的中心向量，並以 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 矩陣的列向量作爲測試文件的文件向量，所得的分類結果較佳。

至於在實驗變因組合 5、7、9、11 中，7 與 11 兩組的結果又較 5 與 9 兩組爲佳，約高 1%，也就是說，對 **D** 矩陣的列向量不進行正規化的處理比進行正規化處理可得到較佳的結果，但差異相當小。至於 5 與 9 兩組以及 7 與 11 兩組，在 A、B

表 4-1 LSI 搭配中心向量法所進行之分類實驗的結果。

變因組合	1	2	3	4	5	6	7	8	9	10	11	12
文件分割 A (%)	51.4	50.3	50.7	49.7	58.1	50.0	58.9	49.4	57.9	49.0	58.8	48.6
文件分割 B (%)	52.4	51.5	51.8	50.9	57.5	49.5	58.3	49.6	57.5	49.4	58.9	49.9

表 4-2 傳統向量空間法搭配中心向量法所進行之分類實驗的結果。

變因組合	1	2
文件分割 A (%)	63.4	63.5
文件分割 B (%)	60.6	60.2

兩組文件分割中各有優劣，且差異都非常小，約 1%。由此可知，在執行 GTP 時，是否對原始的文件向量進行正規化的處理，對於分類的結果並沒有明顯的影響。

就各種實驗的變因組合而言，文件分割 A 與文件分割 B 所得的結果都非常接近，似乎意味著不同的訓練文件與測試文件的選取，對於 LSI 與中心向量法的組合不會造成太大的影響。

表 4-2 則列出了運用傳統向量空間法結合中心向量法所進行之分類實驗的結果。在此部分，實驗變因組合 1 所得的分類結果較變因組合 2 爲佳，但差距不到 1%，因此是否對訓練文件的文件向量進行正規化處理，似乎對分類的結果影響不大。

但不論是在變因組合 1 或變因組合 2 中，文件分割 A 與文件分割 B 所得的結果卻有約 3% 的差異，似乎意味著不同的訓練文件與測試文件的選取，對於傳統向量空間法與中心向量法的組合可能造成較大的影響。

比較表 4-1 與表 4-2 的實驗結果，可以發現使用中心向量法進行文件分類時，運用 LSI 技術所得的結果並不會比運用傳統向量空間法來得好。若以兩者所得的最佳結果來看，在文件分割 A 中，使用傳統向量空間法所得的分類正確率較使用 LSI 高約 5%，在文件分割 B 中，則傳統向量空間法所得的分類正確率較使用 LSI 高約 2%。

4.1.2 利用 *k*-NN 法的分類正確率

表 4-3 列出了運用 LSI 結合 *k*-NN 法所進行之分類實驗的結果，表中除了顯示 3.5.7 節所述各種實驗變因組合所的結果外，也同時列出了 A 與 B 兩組文件分割，以及 $k = 30$ 與 $k = 10$ 的結果。表中所示的分類結果都是如 3.5.9 節所述的正確率。

從 LSI 結合 *k*-NN 法的結果中可看出，不論是在文件分割 A 或文件分割 B 中，且不論 *k* 值爲 30 或 10 時，實驗變因組合 1 所得的結果都是最佳的。此組實驗變因是使用 **DS** 矩陣的列向量作爲訓練文件的文件向量，而使用 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 作爲測試文件的文

表 4-3 LSI 搭配 *k*-NN 法所進行之分類實驗的結果。

k 值	變因組合	1	2	3	4	5	6	7	8	9	10	11	12
30	文件分割 A (%)	68.8	–	68.8	–	67.9	67.9	–	–	67.2	67.2	–	–
	文件分割 B (%)	67.7	–	67.4	–	67.4	67.4	–	–	63.2	63.2	–	–
10	文件分割 A (%)	69.3	–	68.5	–	68.8	68.8	–	–	67.6	67.6	–	–
	文件分割 B (%)	68.0	–	67.6	–	67.4	67.4	–	–	66.6	66.6	–	–

表 4-4 傳統向量空間法搭配 *k*-NN 法所進行之分類實驗的結果。

k 值	變因組合	1	2
30	文件分割 A (%)	76.1	–
	文件分割 B (%)	75.8	–
10	文件分割 A (%)	78.1	–
	文件分割 B (%)	77.0	–

件向量，這與 LSI 的原始概念完全吻合。而此兩組實驗變因組合所得結果的相近，意味著執行 GTP 時是否對原始的文件向量進行正規化處理，對分類結果並不會造成明顯的影響。比較 *k* 值爲 30 或 10 的結果，當 *k* 爲 10 時，分類正確率較高，但與 *k* 爲 30 時的差異並不大。

另外，實驗變因組合 5 與 6 的結果相同，實驗變因組合 9 與 10 的結果也相同，顯示不論是以 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 或 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}\mathbf{S}^{-1}$ 作爲測試文件的文件向量，對分類的結果都不會造成影響。

在文件分割 A 中，各種實驗變因組合所得的結果差異不大，最好的與最壞的差距約爲 1.6%。而在文件分割 B 中，當 *k* 取 30 時，最好的結果與最壞的差距 4.5%，但 *k* 取 10 時，差距僅有 1.4%，似乎 *k* 爲 10 時有較穩定的結果。

表 4-4 則列出了運用傳統向量空間法結合 *k*-NN 法所進行之分類實驗的結果。在此部分，實際上只有一組變因組合，因爲在執行 GTP 時，是否對原始的文件向量進行正規化處理，對分類的結果並沒有影響。而觀察不同 *k* 值所得的結果，*k* 取 10 時的

分類正確率較 *k* 取 30 時約高 2%。

比較表 4-3 與表 4-4 的實驗結果，可以發現使用 *k*-NN 法進行文件分類時，運用 LSI 技術所得的結果也不會比運用傳統向量空間法來得好。若以兩者所得的最佳結果來看，在文件分割 A 與文件分割 B 中的差距分別約爲 7% 與 9%。

4.2 分類效率

4.2.1 利用中心向量法的分類效率

表 4-5 列出了運用 LSI 結合中心向量法進行分類實驗時，每份測試文件平均所花費的時間。表 4-6 則列出了運用傳統向量空間法結合中心向量法進行分類實驗時，每份測試文件平均所花費的時間。表中除了顯示 3.5.7 節所述各種實驗變因組合所的結果

表 4-5 LSI 搭配中心向量法進行分類實驗時，每份測試文件平均所花費的時間。

變因組合	1	2	3	4	5	6	7	8	9	10	11	12
文件分割 A	0.33	0.33	0.33	0.32	0.33	0.32	0.33	0.33	0.33	0.32	0.32	0.33
文件分割 B	0.32	0.33	0.32	0.32	0.32	0.32	0.33	0.32	0.34	0.33	0.32	0.32

單位：秒

註： 因每份測試文件花費時間極短，故先計算全部測試文件所費時間，再取每份文件的平均值。

表 4-6 傳統向量空間法搭配中心向量法進行分類實驗時，每份測試文件平均所花費的時間。

變因組合	1	2
文件分割 A	23.9	24.1
文件分割 B	24.3	23.4

單位：秒

外，也同時列出了 A 與 B 兩組文件分割的結果。

從此兩表可以明顯看出，在配合中心向量的分類法時，使用 LSI 進行測試文件分類所花費的時間較使用使用傳統向量空間法短了許多。後者所花的時間約爲前者的 73 倍。這樣的結果是很容易理解的。因爲使用中心向量法進行分類時，測試文件的分類只是很單純地與各類別的中心向量進行相似度比較，因此向量的維度有決定性的影響。LSI 因爲對原始的文件向量進行降階處理，使得文件向量的維度從數千甚至上萬，減少至一百，因此所花的時間可以大幅減少。

各種實驗變因組合在執行效率上基本上是相同的，這表示各種實驗變因組合對執行效率並沒有特別的影響。

4.2.2 利用 *k*-NN 法的分類效率

表 4-7 列出了運用 LSI 結合 *k*-NN 法進行分類實驗時，每份測試文件平均所花費的時間。表 4-8 則列出了運用傳統向量空間法結合 *k*-NN 法進行分類實驗時，每份測試文件平均所花費的時間。表中除了顯示 3.5.7 節所述各種實驗變因組合所的結果外，也同時列出了 A 與 B 兩組文件分割，以及 $k = 30$ 與 $k = 10$ 的結果。

在配合 *k*-NN 分類法時，使用 LSI 進行測試文件分類所花費的時間與使用使用傳統向量空間法相比，雖然較短，但並沒有很明顯的差距，向量空間法所花的時間約爲 LSI 的 1.35 倍。這與前述配合中心向量法進行文件分類的效率有截然不同的差異。雖然在 *k*-NN 法中，需將測試文件的文件向量與訓練文件一一比較，向量的維度對於文件相似

表 4-7 LSI 搭配 *k*-NN 法進行分類實驗時，每份測試文件平均所花費的時間。

k 值	變因組合	1	2	3	4	5	6	7	8	9	10	11	12
30	文件分割 A	145	–	137	–	140	138	–	–	135	137	–	–
	文件分割 B	135	–	141	–	126	134	–	–	134	138	–	–
10	文件分割 A	135	–	136	–	144	141	–	–	141	139	–	–
	文件分割 B	144	–	144	–	140	141	–	–	138	138	–	–

單位：秒

表 4-8 傳統向量空間法搭配 *k*-NN 法進行分類實驗時，每份測試文件平均所花費的時間。

k 值	變因組合	1	2
30	文件分割 A	189	–
	文件分割 B	187	–
10	文件分割 A	188	–
	文件分割 B	189	–

單位：秒

度的比較有絕對的影響，但在 *k*-NN 法中，因爲得將測試文件與訓練文件比較的結果進行排序，以便找出最相近的 *k* 份訓練文件，而排序所花的時間相較於文件比較所花的時間，對執行效率具有比較大的影響。不論是利用 LSI，或是使用傳統向量空間法，每份測試文件所需處理的排序狀況是相同的，造成兩者所花的全部時間沒有太大差異。至於使用不同 *k* 值的分類測試，因爲都需要對全部文件所得結果進行排序，故差異很小。

4.3 綜合討論

4.3.1 LSI 與傳統向量空間法的分類結果比較

從前述的研究結果中可以看出，運用 LSI 技術將 SVD 降階處理後的文件向量作爲文件的表述，配合適當的分類演算法，可以得穩定的分類結果，因此運用 LSI 技術進行自動化文件分類是可行的。但不論是配合中心向量法或 *k*-NN 法進行文件分類，運用 LSI 技術所得的結果都不及運用傳統向量空間法來得好，LSI 所得的分類正確率較低。

回顧 LSI 技術的出發點，其目的是希望解決同義字的問題，透過數學上 SVD 降階的處理，將過度依賴字詞形式的狀況轉換成較具彈性而能展現語意的狀況，藉此在進行文件資料檢索時，能找出更多與查詢條件相關的文件。因此 LSI 原本的目的是希望建立文件彼此間更多的關聯性，而不致在檢索時有所遺漏。

本研究所探討的是單一分類的問題，在單一分類的問題中，每份測試文件只能歸屬於單一個類別。雖然運用 LSI 技術展現語意的概念與分類的概念是一致的，但單一分類的概念似乎又與 LSI 希望找出文件間更多關聯性的意圖互相衝突。或許這是造成 LSI 在分類正確率上不及傳統向量空間法的原因。

如果前述的猜測是正確的，那麼 LSI 或許比較適合運用在多重分類的問題中，也就是每份文件可以同時歸屬於多個不同的類別。

雖然在分類正確率上 LSI 不及傳統向量空間法，但運用 LSI 的分類效率卻明顯優於傳統向量空間法，尤其在使用中心向量分類法時特別明顯。這是因爲 LSI 對文件向量進行降階處理，使得後續各種相關的數學計算得以簡化，而縮短花費的時間。

4.3.2 各種實驗變因的影響

本研究針對文件向量的正規化與否以及 LSI 技術中 **S** 矩陣的影響，設計了幾種不同的實驗變因組合。雖然在使用 *k*-NN 分類法時，不同的變因組合沒有太大的差別，但最佳的結果出現在變因組合 1 中。該變因組合係使用 **DS** 矩陣的列向量作爲訓練文件的文件向量，而以 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 矩陣的列向量作爲測試文件的文件向量，這完全符合 LSI 技術所使用的文件向量形式。

而在使用中心向量分類法時，不同的變因組合就有了比較明顯的差異。但比較特別的是，在本研究所得的結果中，分類正確率最佳的變因組合是以 **D** 矩陣的列向量作爲訓練文件的文件向量，而以 $\mathbf{X}_{\text{test}}^{T}\mathbf{T}$ 矩陣的列向量作爲測試文件的文件向量。這與 LSI 技術以 **DS** 矩陣的列向量表述訓練文件並不符合。

在 2.4.3 節中曾提及，因爲 **S** 矩陣是個對角的 (diagonal) 矩陣，因此 **DS** 矩陣與 **D** 矩陣的差異在於其列向量座標軸存在著某種比例上的關係，該比例正好對應到 **S** 矩陣中對角線上各元素的值。因爲此種比例關係是線性的 (linear)，因此 **DS** 矩陣與 **D** 矩陣在意義上就某種層面而言是相同的。

但爲什麼 **DS** 矩陣與 **D** 矩陣仍對中心向量分類法造成相當明顯的差異，其中的原因仍有待進一步的探討。

另外在使用中心向量分類法的實驗變因組合中，在向量加總前進行向量的正規化與否，並沒有對分類正確率造成明顯的差別。這必須對向量加總重新探討。兩向量如果原始長度差異不大，加總前是否先進行正規化處理對加總的結果影響不大；但如果兩向量

的原始長度差異很大，加總前是否先進行正規化處理，對於加總的結果就會產生較大的差異。

本研究使用的文件資料係摘要資料，文件的長度並不長，而且各文件長度的差異也不算大，因此文件向量是否進行正規化的處理，對於分類結果的影響就比較小。至於如果處理的文件長度差異很大時，文件向量正規化與否是否真的對分類結果造成明顯的影響，則需要使用不同的文件資料進一步探討。

4.3.3 中心向量法與 *k*-NN 法的比較

在本研究中，不論使用 LSI 技術或傳統向量空間法進行文件的表述，搭配 *k*-NN 法所得的分類正確率都明顯優於搭配中心向量法所得的分類正確率。這與過去文件分類相關研究所獲致的結論完全吻合。

k-NN 法已被證明是相當不錯的分類演算法，可以得到較高的分類正確率。但 *k*-NN 法在計算上需要耗費大量的時間，分類的效率是比較差的。中心向量法因爲使用過於簡化的模型，因此其分類正確率一般而言都不理想。但中心向量法所需要的數學計算相對其他分類演算法而言相當少，因此有相當高的執行效率。

單純就 *k*-NN 法所得的結果而言，*k* 值選用 30 時，與分類的類別數 38 較接近，而 *k* 值選用 10 則與各類別最少文件數 10 一致，後者所得的分類正確率較前者稍高，但差異有限。對於文件類別數與各類別所含文件數的選取準則，以及最佳的 *k* 值，仍有進一步探討的空間。

第五章　結論與建議

隨著網際網路與相關資訊技術的發展，文件資料累積的速度相當快速，加上各個組織對於知識管理的需求日益殷切，使得人們不斷地思考如何能更有效地運用龐大的文件資料。

雖然許多文件搜尋引擎應運而生，而其搭配的全文檢索或相關技術提供了人們快速尋找文件資料的途徑，但此一途徑受限於對字詞形式的過度依賴，只解決了部分的問題，並不足以提供全面有效利用文件資料的方法。

爲了彌補此一不足之處，提供概念化的文件瀏覽服務是必需的。而要提供良好的概念化文件瀏覽服務，則有賴於良好的文件分類。傳統的文件分類多仰賴專家的人爲判斷，雖然可信度高，但面對日益增加的龐大文件資料，卻有嚴重的效率問題。因此利用適當的資訊技術，對龐大的文件資料進行自動化的分類處理是無可避免的途徑，也是踏入知識管理領域成功與否的關鍵。

5.1　研究結論

本研究的目的在探討運用潛在語意索引 (LSI) 技術於自動化文件分類的可行性。由本研究所設計的實驗與獲致的結果，大致可獲得下列的結論。

一、 使用 LSI 技術進行文件的表述，搭配適當的分類演算法，可用於文件的自動化分類，且分類效果穩定。

二、 搭配中心向量法或 *k*-NN 法進行文件的自動化分類時，使用 LSI 技術作爲文件表述所得的分類正確率並不會比傳統向量空間法來得理想。但使用 LSI 可得較佳的執行效率。

三、 以 LSI 搭配 *k*-NN 法進行文件自動化分類較搭配中心向量法可得到明顯較佳的分類正確率，但搭配中心向量法時，卻明顯有著較佳的執行效率。

四、 以 LSI 技術進行文件的表述時，不論搭配中心向量法或 *k*-NN 法，對原始文件向

量進行正規化與否，都不會對分類結果造成明顯的差異。

五、在 *k*-NN 法中，*k* 值選用 10 所得分類正確率較 30 略佳，但差異不大。

5.2 研究限制

本研究使用的文件資料集係選用 Inspec on Disc 自 1989 年至 1999 年期間電腦科學相關文獻摘要資料 (Abstracts of Computer Science Selections)。該文件資料集的文件同質性相當高，而本研究選取其分類架構的第二階分類點，類別的數量不少，可能造成各類別之間的差異不明顯，而影響分類的正確率。如將本研究的方法應用於不同領域的資料分類，可能會有較佳的結果。

本研究僅處理單一類別的分類問題，因此在文件資料的選取上排除了同時隸屬多個不同類別的文件。但在現實世界中，許多分類問題都屬於多類別的，因此本研究所獲致的結果並不足以充分反映出現實世界的狀況。

5.3 未來研究建議

雖然本研究對於運用 LSI 技術於自動化文件分類有初步的結果，也證明了其可行性，但受限於時間與研究的規模，對於相關延伸問題無法再做深入的探討。因此針對本研究採用的方法與前一節所提本研究的限制，對於本研究未來可持續進行的研究方向與重點，提出下列幾點建議供後續研究參考。

一、 不僅探討單一分類的問題，還應探討多分類的問題。因爲一份文件同時隸屬多個不同的類別，比較符合現實世界的狀況。而 LSI 技術依其特性與目的，可能比較適合處理多分類的問題。

二、 選用不同的文件資料集進行相關的研究與探討。本研究僅使用了從 Inspec on Disc 選出的部分文件資料，未來的研究可選取更多不同型態的資料，以廣泛及深入探討運用 LSI 於自動化文件分類的可行性與效能。

三、 本研究在使用 *k*-NN 分類演算法時，並未對 *k* 值進行最佳化的處理。未來的研究可對 *k* 值進行更有系統的探討。

四、 本研究利用傳統向量空間法與 LSI 法對照，分別搭配中心向量與 *k*-NN 兩種分類演算法進行文件分類的實驗，未來的研究可考慮將 LSI 與其他不同的分類演算法搭配，例如類神經網路、機率法等，以便於更充分瞭解運用 LSI 於自動化文件分類的效能。

參考文獻

[1] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*, ACM Press Books, ACM & Addison-Wesley, 1999.

[2] 林頌堅, "自動化文件分類在資訊服務上的應用", *資訊傳播與圖書館學*, **5** (2), 87–102, 1998.

[3] C. J. van Rijsbergen, *Information Retrieval*, 2nd edition, Butterworths, London, 1979.

[4] G. Slaton and C. S. Yang, "On the Specification of Term Values in Automatic Indexing", *Journal of Documentation*, **29** (4), 351–372, 1973.

[5] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*, McGraw Hill, 1983.

[6] Y. Yang and X. Liu, "A Re-examination of Text Categorization Methods", *Proceedings of SIGIR-99, 22nd ACM International Conference on Research and Development in Information Retrieval*, 42–49, 1999.

[7] Y. Yang, "An Evaluation of Statistical Approaches to Text Categorization", *Information Retrieval*, Kluwer Academic Publishers, **1**, 69–90, 1999.

[8] Y. Yang, "Expert Network: Effective and Efficient Learning from Human Decisions in Text Categorization and Retrieval", in *17th Ann. Int. ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '94)*, 13–22, 1994.

[9] T. Mitchell, *Machine Learning*, McGraw Hill, 1996.

[10] 吳文峰, "中文郵件分類器之設計及實作", 逢甲大學資訊工程研究所碩士論文, 2002.

[11] J. R. Quinlan, "Induction of Decision Trees", *Machine Learning*, **1** (1), 81–106, 1986.

[12] E. Wiener, J. O. Pedersen, and A. S. Weigend, "A Neural Network Approach to Topic Spotting", in *Proceedings of the 4th Annual Symposium on Document Analysis and*

Information Retrieval (SDAIR '95), 317–332, 1995.

[13] H. T. Ng, W. B. Goh, and K. L. Low, “Feature Selection, Perceptron Learning, and a Usability Case Study for Text Categorization”, in *20th Ann. Int. ACM SIGIR Conference on Research and Development in Information Retrievel (SIGIR '97)*, 67–73, 1997.

[14] Y. Yang and C. G. Chute, “A Linear Least Squares Fit Mapping Method for Information Retrieval from Natural Language Texts”, in *Proceedings of the 14th International Conference on Computational Linguistics (COLING 92)*, 447–453, 1992.

[15] 莊慧美, “以智慧型計算方法探討文件分類”, 國立屏東科技大學資訊管理研究所碩士論文, 2000.

[16] 涂宜昆, “以單類支持向量機爲基礎之階層式文件分類”, 國立成功大學資訊工程研究所碩士論文, 2003.

[17] H. P. Zipf, *Human Behavior and the Principle of Least Effort*, Cambridge, Massachusetts, 1949.

[18] H. P. Luhn, “The Automatic Creation of Literature Abstracts”, *IBM Journal of Research and Development*, **2**, 159–165, 1958.

[19] K. Spark Jones, “A Statistical Interpretation of Term Specificity and Its Application in Retrieval”, *Journal of Documentation*, **28** (1), 11–20, 1972.

[20] S. T. Dumais, S. Deerwester, and R. Harshman, “Indexing by Latent Semantic Analysis”, *Journal of the American Society for Information Science*, **41** (6), 391–407, 1990.

[21] J. I. Hong, “An Overview of Latent Semantic Indexing”, http://www.cs.berkeley.edu/~jasonh/classes/sims240/sims-240-final-paper-lsi.htm.

[22] S. T. Dumais, “LSI Meets TREC: A Status Report”, in *The First Text Retrieval Conference (TREC1)*, 137–152, 1993.

[23] S. T. Dumais, “Latent Semantic Indexing (LSI) and TREC-2”, in *The Second Text*

Retrieval Conference (TREC2), 105–116, 1994.

[24] S. T. Dumais, “Latent Semantic Indexing (LSI): TREC-3 Report”, in *The Third Text Retrieval Conference (TREC3)*, 219–230, 1995.

[25] T. A. Letsche and M. W. Berry, “Large-Scale Information Retrieval with Latent Semantic Indexing”, *Information Science*, **100**, 105–137, 1997.

[26] P. Foltz, “Using Latent Semantic Indexing for Information Filtering”, in *Proceeding ACM Conference Office Information System (COIS)*, 40–47, 1990.

[27] 黃卓倫, “利用隱藏語意索引進行文件分段檢索之研究”, 國立台灣大學資訊管理研究所碩士論文, 1997.

[28] G. E. Forsythe, M. A. Malcolm, and C. B. Moler, *Computer Methods for Mathematical Computations*, Chap. 9, NJ: Prentice Hall, 1977.

[29] L. Baoli, Y. Shiwen, and L. Qin, “An Improved *k*-Nearest Neighbor Algorithm for Text Categorization”, *Proceedings of the 20th International Conference on Computer Processing of Oriental Languages*, Chenyang, China, 2003.

[30] M. Mullin and R. Sukthankar, “Complete Cross-Validation for Nearest Neighbor Classifiers”, *Proceedings of the International Conference on Machine Learning*, 639–646, June 2000.

[31] K. Aas and L. Eikvil, “Text Categorization: A Survey”, Technical Report, Norwegian Computer Center, 1999.

[32] F. Sebastiani, “Machine Learning in Automated Text Categorization”, *ACM Computing Surveys*, **34**, 1, 1–47, 2002

[33] J. W. T. Wong, W. K. Kan, and G. Young, “ACTION: Automatic Classification for Full-Text Documents”, *SIGIR Forum*, *ACM Special Interest Group on Information Retrievel*, **30** (1), 26–41, 1996.

[34] S. D’Alessio, K. Murray, R. Schiaffino, and A. Kershenbaum, “Category Levels in Hierarchical Text Categorization”, *Proceedings of the 3rd Conference on Empirical Methods in Natural Language Processing (EMNLP-3)*, 1998.

[35] S. Dumais and H. Chen, “Hierarchical Classification of Web Content”, *Proceedings of SIGIR 2000*, 256–263, 2000.

[36] V. Dasigi, R. C. Mann, and V. A. Protopopescu, “Information Fusion for Text Classification—An Experimental Comparison”, *Pattern Recognition*, **34**, 2413–2425, 2001.

[37] D. D. Lewis, “Representation and Learning in Information Retrieval”, Ph.D. Dissertation, University of Massachusetts, Amherst, 1992.

[38] S. Zelikovitz and H. Hirsh, “Using LSI for Text Classification in the Presence of Background Text”, *Proceedings of the 10th International Conference on Information and Knowledge Management (CIKM-2001)*, 113–118, 2001

[39] M. F. Porter, “An Algorithm for Suffix Stripping”, *Program*, **14** (3), 130–137, 1980.

[40] The Porter Stemming Algorithm, http://www.tartarus.org/~martin/PorterStemmer/

[41] Latent Semantic Indexing Web Site, http://www.cs.utk.edu/~lsi/

[42] MySQL Web Site, http://www.mysql.com/

[43] Perl Web Site, http://www.perl.com/