國立政治大學資訊管理研究所

碩士學位論文

指導教授：楊建民博士

雲端運算服務環境下運用文字探勘於語意註解網頁文件分析之研究

研究生：黃孝文

中華民國九十九年七月

致 謝

時光飛逝，兩年的研究所生活一眨眼便過去了，回顧過去這兩年時光，論文的完成絕非一己之力所能完成，本論文能順利完成，承蒙指導教授 楊建民博士在課業上、工作與為人處世各方面的悉心指導，論文撰寫過程中盡其所能地給予引導協助，由衷感謝。同時要感謝口試委員政治大學林我聰教授、李有仁教授與台北大學電子商務中心邱光輝主任提供非常多寶貴意見，在此致上最誠摯的敬意與感謝。

研究所的求學生涯中，有非常多的成果與收穫，這一切要感謝過程中周遭朋友所給予的協助，感謝同師門的婉瑋學姐、春美學姐、繼鴻學長、世傑學長、俊宏學長、鈞華學長、宜儒、少華、承翰、世蓉、榕芝以及學弟們，與你們共同生活在「數位創新研究管理計劃室」，除了砥礪學業之外，也讓我增添了很多美好回憶；謝謝我所愛的朋友們，讓我能暫時放下在論文撰寫過程中的不如意，恣意地開懷大笑；特別感謝我摯愛的女朋友，宛婷。感謝妳在背後的默默支持，分享我研究上獲得進展的喜悅，撫平我大大小小不順心的事情，妳是我一路上積極前進的動力，謝謝妳不離不棄的相伴。

最後謹以完成碩士學業的榮耀與喜悅獻給用心栽培我的父母與家人，因為你們的支持，讓我無後顧之憂的完成學業，也期許自己在今後人生的道路上，能夠不忘記自己的初衷，並且面對未知與困難無所畏懼。在此獻上最真誠的謝意，給所有曾經幫助過我的人，謝謝你們。

黃孝文

謹誌於 國立政治大學資訊管理研究所

2010 年 7 月

摘要

隨著網路的快速成長，資料探勘(Data Mining)及文字探勘(Text Mining)所須分析的資料集越來越龐大，透過單一機器執行資料探勘分析受限於記憶體大小及其計算能力，不僅運算時間大幅增加，分析資料集的檔案大小也因而受到限制；語意註解萃取出文件的重要內容，凸顯主題加強資料探勘及文字探勘的效果，而資料探勘、文字探勘和語意註解背後都牽涉到大規模的資料處理，透過雲端運算的技術使負載平衡，將運算工作分散至運算叢集中的每一台電腦，不僅加快運算和儲存的速度，更可降低整體的風險。

本研究使用 Hadoop 軟體實作雲端文字探勘平台，用於分散式文字探勘及結果分析，採用涵蓋21578篇新聞文件的路透社資料集(Reuters 21578)進行實證分析，依照 Mod Apte 切分法分為訓練資料集及測試資料集用以進行文件分類，文件分類的步驟分為數個部分，分別為進行資料格式轉換的資料前置處理、針對文件內容加註更詳盡的連結及描述的語意註解、用以產生分類預測模型的分類器(簡單貝氏分類器、餘集簡單貝氏分類器)與評估文件分類結果的評估器；路透社資料集經過去除停用字、附加語意註解資料及文本詞彙長度統計分類，再進行簡單貝氏分類器及餘集簡單貝氏分類器的訓練，比較測試資料集的分類正確率作為文件分類實證結果。

本研究根據實驗結果發現，探討去除停用字、語意註解、文件分類演算法及文本詞彙長度對於文件分類正確率的影響：(1)去除停用字使出現頻率高的停用字對於分類預測產生負面影響；(2)語意註解作為詮釋資料的取得方式，可增加文件分類的效果；(3)餘集簡單貝氏分類器，可用以減少偏斜資料對於分類預測結果的誤判；(4)文本詞彙長度較長的文章則會某種程度主導分類預測結果，造成誤判的產生，降低分類正確率；透過上述各影響因子的調整使文件分類的結果得到改善，使得文件分類正確率獲得較佳的效果。

本研究提出之系統以雲端運算環境運行文件分類演算法，使得大型資料集得以更為迅速取得分析結果，使用語意註解作為詮釋資料的來源，使得文件分類模型產生過程中有更多資訊可分析，使得機器判斷的正確程度獲得改善，亦可將文件轉換為語意網文件，供語意網搜尋引擎查詢檢索，未來應加入 Twitter 或 Facebook 等擁有大量非結構化資料的網站之資料，使本平台得以分析更大規模的資料，並且考慮資料集類別分佈的集中程度對分類正確率的影響程度，同時應實作效果更佳的分類演算法，進而改善系統整體的結果。

關鍵字: 雲端運算、文件分類、語意註解、簡單貝氏分類器

Abstract

Nowadays, businesses perform data mining and text mining need to handle large scale dataset. The computational resources of servers are often limited and lack of efficient to compute analytical jobs. But if they could run their data mining jobs under cloud computing clusters, they are able to get results very quickly on a large dataset without "out of memory" problems.

In this paper, a series of experiments are conducted to measure and analyze the accuracy of the classification algorithms implemented on Hadoop using Reuters-21578 dataset; the process of text mining consisted of four stages: (1)data preprocessing, (2)semantic annotation, (3)classifier, (4)evaluator. Reuters-21578 had divided into training set and testing set based on Mod Apte Split, processed by stopwords removal, appended semantic annotations as metadata and splitted into several subsets according to different document sizes. Experiments outlined several issues that will need to be considered when conducting text mining.

According to the experiment results, the researcher found that stopwords removal, semantic annotation, different classification algorithms and different document sizes could improve the classification accuracy. First, stopwords removal avoids common words from becoming noises that will do harm to classification result. Second, semantic annotation as the extra information could improve the result. Third, complementary naive bayes algorithm could solve the decision boundary problem which naive bayesian cannot handle. Fourth, long documents could dominate the classification results. Sixth, the class imbalance problem could cause a drop of classification accuracy. Text mining result could be improved by adjusting the parameters found above.

Keywords: Cloud Computing, Document Classification, Semantic Annotation, Naive Bayesian Algorithm

目　錄

誌 謝 I
摘 要 II
目 錄 IV
圖索引 VI
表索引 VII
第一章　緒論 1
第一節 研究背景 1
第二節 研究動機 2
第三節 研究目的 3
第四節 研究架構 4
第二章　文獻探討 5
第一節 資料探勘與文字探勘 5
2.1.1 資料探勘 5
2.1.2 文字探勘 7
2.1.3 資料探勘與文字探勘之差異 7
2.1.4 簡單貝氏分類器 9
第二節 雲端運算 10
2.2.1 雲端運算的定義 10
2.2.2 現有的雲端運算服務 16
第三節 MapReduce 軟體設計模型 16
2.3.1 Google 的分散式資料庫 BigTable 18
2.3.2 MapReduce 軟體設計模型應用於資料探勘之研究 20
第四節 實作 MapReduce 架構的框架 20
2.4.1 Hadoop 21
2.4.2 DisCo 22
2.4.3 使用雲端運算平台的其他方案 23
第五節 現今語意網的相關發展 24
2.5.1 鍊結資料(Linked Data) 24
2.5.2 Facebook 25
2.5.3 Wolfram Alpha 27
第六節 語意註解(Semantic Annotation) 28
2.6.1 語意註解的分類 30
2.6.2 語意註解的發展 31
2.6.3 語意註解於資料探勘上的應用 31
第三章　研究方法 32
第一節 研究設計 33
第二節 文本資料集 Reuters 21578 35

3.2.1 路透社資料集文件格式 35
3.2.2 路透社資料集的歪斜資料特性 37
第三節 資料前置處理 39
3.3.1 斷詞切字(Tokenization) 39
3.3.2 去除停用字(Stop Words Removal) 39
3.3.3 詞幹還原(Stemming) 40
第四節 語意註解 41
第五節 分類器(Classifier) 43
3.4.1 簡單貝氏分類器 43
3.4.2 餘集簡單貝氏分類器 45
3.4.3 簡單貝氏分類器與餘集簡單貝氏分類器的實作 45
第六節 評估器(Evaluator) 46
第七節 平台建置 46
第四章 研究成果 48
第一節 去除停用字對文件分類正確率的影響 48
第二節 語意註解對文件分類正確率的影響 50
第三節 比較簡單貝氏分類器與餘集簡單貝氏分類器之分類正確率 51
第四節 餘集簡單貝氏分類器消除偏斜資料產生的決策邊界問題之效果 52
第五節 文本詞彙長度對文件分類正確率的影響 53
第六節 避免模型過適問題 54
第七節 類別分佈均勻程度對於分類結果之影響 56
第八節 各階段實驗結論整理 57
第五章 結論與建議 60
第一節 結論 60
第二節 未來研究方向 61
參考文獻 62

圖索引

圖 2-1 KDD 步驟圖 5
圖 2-2 三種雲端運算產業模式示意圖 11
圖 2-3 雲端運算架構層次 12
圖 2-4 雲端運算服務比較表 16
圖 2-5 MapReduce 軟體設計模型示意圖 17
圖 2-6 MapReduce 軟體設計模型運作過程示意圖 18
圖 2-7 Bigtable 資料儲存之示意圖 19
圖 2-8 Bigtable 資料實際存放的基本單位為 Column Family 19
圖 2-9 鍊結資料目前加入網站示意圖 24
圖 2-10 詮釋資料的種類 28
圖 2-11 語意註解示意圖 29
圖 3-1 本研究之系統架構圖 33
圖 3-2 本研究之文字探勘流程圖 34
圖 3-3 路透社資料集之訓練資料的類別分佈狀態 38
圖 3-4 本研究之系統運作圖 38
圖 4-1 去除停用字對於文件分類正確率的影響 49
圖 4-2 語意註解對於文件分類正確率的影響 50
圖 4-3 簡單貝氏分類器及餘集簡單貝氏分類器之分類正確率的影響 51
圖 4-4 文本詞彙長度對文件分類正確率的影響 52
圖 4-5 簡單貝氏分類器的訓練集錯誤率及測試集錯誤率 55
圖 4-6 餘集簡單貝氏分類器的訓練集錯誤率及測試集錯誤率 55
圖 4-7 資料集中類別分佈對分類結果的影響 56
圖 4-8 路透社資料集中各詞彙長度的文件之分佈情形 59

表索引

表 2-1 MapReduce 之開放程式碼框架及其實作程式語言 21
表 3-1 Reuters 21578 資料集的標籤 35
表 3-2 Reuters 21578 文件的範例 36
表 3-3 路透社資料集的資料分佈 37
表 3-4 斷詞切字的輸入及輸出示例 39
表 3-5 語意註解輸入資料範例 37
表 3-6 語意註解輸出資料範例 39
表 4-1 簡單貝氏分類器及餘集簡單貝氏分類器之權重值為前 2500 名中各類別的分佈情形 52
表 4-2 各階段實驗最佳的參數組合及分類正確率 57

第一章 緒論

第一節 研究背景

在如今企業資訊應用的環境下，所需分析的資料都是相當龐大的，其中資料探勘及文字探勘都是運算密集的工作，若是企業採用單一伺服器執行十分吃力、費時。對於分秒必爭的企業環境來說，如何使用雲端運算環境分散運算的沉重負擔，以迅速取得分析結果成為一門重要的課題。

雲端運算(Cloud Computing)一詞，由 Google 於2007年下半年所提出，由於它有足夠的能力去提供彈性化的動態 IT 基礎設施、保證服務品質的運算環境和可配置的軟體服務，持續不斷地在全球掀起雲端運算的風潮。在學術界與企業界中已有許多的計畫和產品，如 Amazon 使用 EC2(Elastic Compute Cloud)和簡單儲存服務(Simple Storage Service, S3)為企業提供計算和儲存服務；IBM 在2007年11月推出了“Blue Cloud”為客戶提供雲端運算的平台；2008年 IBM 和17個歐洲研究組織以「無障礙的資源和服務虛擬化」為口號，共同發展名為“RESERVOIR”的雲端運算計畫；Intel、Yahoo!、HP 共同成立的雲端運算測試平台(Cloud Computing TestBed)是一個全球性的開放原始碼計畫。此測試平台由許多的資料中心所組成，推展軟體、資料中心管理與大規模網際網路運算硬體等各方面的研究。這三家公司將成立6個卓越中心，擁有以 HP 在系統管理的專門技術，加上 Yahoo!在平行運算的成果與 Intel 處理器為基礎的雲算運算基礎架構，其擁有一千至四千個處理器核心，由來自世界各地的研究者維護運作(謝良奇，2008)。

語意網的出現，文件得以使用機器友好的格式表示，提供電腦理解網頁內容的可能性，使得電腦能夠針對其內容進行處理及自動化。但以語意網文件格式發展出的網站數量仍然鮮少，因而近年來發展出語意註解技術，幫助傳統網頁製作語意註解，使得語意搜尋引擎的搜尋範圍得以不侷限在少量的網站，提

供更為精確的搜尋結果，距離實現自然語言查詢的目標已經不遠。

第二節　研究動機

隨著網路的快速成長，資料探勘所須分析的資料集越來越龐大。透過單一機器執行資料探勘分析受限於記憶體大小及其計算能力，不僅運算時間大幅增加，分析資料集的檔案大小也因而受到限制。

電腦的計算能力與時俱進，演變為多核心處理器，提昇 CPU 的計算處理能力。MapReduce 軟體開發模型可運用在資料探勘，將其運用在雲端運算的環境上，妥善利用運算叢集的運算能力，並且突破資料集大小的限制。

語意網技術的發展，使得電腦得以解讀所處理文件的內容，但目前網際網路所見仍少有相關應用出現。語意註解作為直接將網頁文件分析後產生語意化文件 (RDF 檔案)的語意網技術，受到關注但未見其廣泛的應用；語意註解是一種資訊擷取的技術，萃取出文件的重要內容，將其應用在文字探勘上，得以減少資訊處理量，降低文字探勘演算法的運算時間。同時透過雲端環境進行文字探勘的資料存放及演算法運算，能夠達到負載平衡同時擴大文字探勘範圍的能力。

資料探勘、文字探勘和語意註解背後都牽涉到大規模的資料處理，透過雲端運算的技術使負載平衡，將運算工作分散至運算叢集中的每一台電腦，不僅加快運算和儲存的速度，更可降低整體的風險。

本研究所提出之雲端文字探勘平台，將大量的運算需求透過雲端運算環境分散計算負擔，並結合語意註解技術對資料加註更為詳盡的詮釋資料，比較簡單貝氏分類器與餘集簡單貝氏分類器運作於雲端運算環境的效果。

第三節　研究目的

本研究主要目的分述如下：

一、提出以雲端運算環境作為文字探勘演算法運行的平台

本研究將以 Hadoop 作為雲端運算系統資料儲存及文字探勘演算法的運算平台。Hadoop 為目前最為人使用的 MapReduce 軟體，它提供負載平衡、儲存大量資料的功能，作為分散式文字探勘的系統架構，由 Hadoop 管理電腦之間的溝通與協調，可專注於實現演算法的內容。

以簡單貝氏分類器及餘集簡單貝氏分類器作為資料分類演算法，同時結合 N-gram Word Model 及語意註解技術，處理輸入的測試資料集後產生模型，將該模型應用於未來資料的分類預測，獲得較佳的分類正確率。

二、以路透社資料集在平台上驗證文字探勘演算法運行之分類正確率

本研究之實驗使用路透社資料集(Reuters 21578)進行驗證，用以針對本研究建立的雛型系統進行實驗，驗證本研究提出的雲端文字探勘平台透過大型資料集評估系統的分類正確率，並與 Kibriya et al.的簡單貝氏分類器研究成果比較。

三、透過語意註解做為詮釋資料的來源改善文件分類之結果

以語意註解作為詮釋資料的來源，透過以本體論為基礎的語意註解平台，擷取文件中的重要內容形成本體論實體，提供該文件額外的詮釋資訊。對文本做語意註解，使得文件分類演算法可分析的資料增加，得到更為正確的文件分類結果，並且使這些文件具備語意化的條件，未來可由語意搜尋引擎查詢檢索。

第四節　研究架構

本節將本論文之研究架構做以下說明：

第一章　　緒論

本章對本研究論文進行簡短說明，描述本研究之背景、動機與目的，最後介紹本論文之研究架構。

第二章　　文獻探討

本章以三部分進行文獻整理。詳述簡單貝氏分類器作為資料探勘分類演算法的方法、雲端運算及語意註解技術。

第三章　　研究方法

首先定義研究設計，提出雲端文字探勘平台的系統架構，說明路透社資料集(Reuters 21578)的前置處理，描述文字探勘採用的簡單貝氏分類器演算法。

第四章　　研究成果

詳述本研究之雲端文字探勘平台系統，針對去除停用字、語意註解對文件分類正確率的影響、比較簡單貝氏分類器與餘集簡單貝氏分類器在文件分類正確率進行實驗，依序驗證上述之各影響因子對文件分類正確率的作用，探討影響因子產生的作用是如何發生的。

第五章　　結論與未來研究

根據研究之過程及研究結果，歸納本研究的結論，並提出未來可行的研究方向，以作為後續研究的參考。

第二章 文獻探討

第一節　資料探勘與文字探勘

2.1.1　資料探勘

隨著時間的累積，各組織的資訊系統中所存放的資料量也隨之增加，這些大量的資料中潛藏某些特徵和關係。資料探勘可在大量存放的資料中，找出先前並不知道，但最後可有效理解的資訊。

資料探勘的過程包含很多步驟，每個步驟都會相互影響而使結果不同，許多人認為「資料探勘」和「資料庫中的知識發現」是同義的，但其實資料探勘僅是 KDD 的其中一部分程序而已，但研究過程中兩者的關係是密不可分的(Fayyad, 1996)。

簡而言之，資料探勘可以說是從大量的資料中萃取出知識的動作，Fayyad(1996)、Han (2005)皆認為資料探勘是知識發現(Knowledge Discovery from Data, KDD)其中的重要步驟。

Fayyad et al. (1996)提出簡單明瞭的 KDD 步驟，如圖2-1，說明如下。

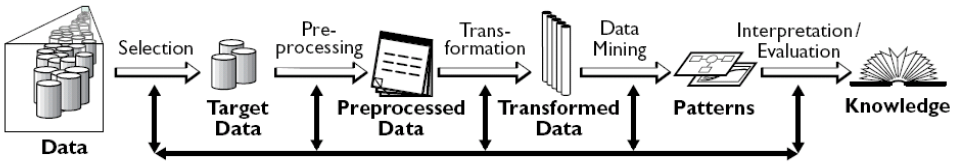


圖2-1 KDD 步驟圖

[資料來源：Fayyed (1996)]

1. 資料選擇 (Selection)

瞭解該領域的知識，挑選與分析工作相關的資料，用以建立目標資料集，在資料探勘的過程中專注於選擇的資料子集合。

2. 前置處理 (Preprocessing)

資料集中的資料會包含錯誤、遺失及不完整的資料內容，必須將其去除，如此一來才能夠排除干擾和不一致資料的影響，並將格式不同的資料進行處理，使其具備一致化。

3. 資料轉換 (Transformation)

進行資料的簡化及轉換工作，從大型資料集中進行分析找出有用的資訊，牽涉龐大的運算量，必須適時減少資料量，例如降維 (Dimension Reduction)、轉換或編碼等方式。

4. 資料探勘 (Data Mining)

KDD 過程中最重要的步驟，透過演算法分析資料找出資料潛藏的特徵及規則，包括了資料分群、資料分類、關聯規則、決策樹、統計回歸等演算法。

5. 解釋或評估 (Interpretation/Evaluation)

經過資料探勘找出的特徵或模式，可用圖形工具轉換為容易理解的圖表，供決策支援之用；另外評估資料探勘產生的模式之正確性也是極重要的，用以判斷產生的模式是否可作為未來商業決策上的應用，例如預測顧客的消費金額。

由於資料庫中的資料存在多種特徵，故資料探勘方法也是相當多樣化。用不同的方法和技術找出不同種類的特徵，基於特徵的分析方式與產生的知識型態，資料探勘的演算方法最常用的為分類分析、群集分析、關聯規則分析。

1. 分類分析 (Classification Analysis)

   分類分析是一種依資料屬性建立類別的過程，通常從資料中產生「若則」法則。

2. 群集分析 (Clustering Analysis)

   群集分析的目的是要把群集與群集間的差異找出來，同時也要將群集內物件的相似性找出來。

3. 關聯規則分析 (Association Analysis)

   每一筆資料庫中的交易資料包含數個交易項目，關聯分析的目的是由這些交易資料中，找出交易項目的關聯法則。此關聯法則提供如下列分析描述「若 X、Y、Z 三種交易項目發生時，會發生交易項目 W 的機率為 p，機率越高表示關連性越高」。

2.1.2 文字探勘

文字探勘所分析的文字為自然語言所撰寫出來的文件，裡頭包含一些具備意義的內容，像是報紙新聞、雜誌專欄、文學作品、使用者手冊、部落格、email 或者線上討論區文章都屬於文字探勘所要分析的資料。在現今資訊爆炸的時代中，可用來分析的文章已經相當具備規模且持續地成長中。

文字探勘的技術雖發展十分繁多複雜，但其目的卻是相當簡單，旨在從這些資料的分析中，發掘新的、有用的的資訊。而文字探勘的技術結合數學、統計、機率、人工智慧、資料檢索及資料庫等相關知識，典型的文字探勘類型為文件分類 (Categorization)、文件分群 (Clustering) 和資訊擷取 (Concept/Entity Extraction)。

2.1.3 資料探勘與文字探勘之差異

資料探勘的目的在於發掘隱藏於「資料」之中的模式(Patterns)，而文字探勘正如其字面上的意義，目的為找尋潛藏於「文字」中的模式。資料與文字之間的差別，就在於結構化與否，資料探勘針對資料庫中所儲存的各種資料進行分析，資料庫中的內容皆是結構化資料，例如購物交易資料庫中的真實記錄，裡頭可能包含交易日期、會員等級、產品類別、促銷折扣、交易金額等資料；反觀文字探勘所分析的文字，是沒有組織的非結構化資料，相較於結構化資料是較難以演算法處理的。

資料探勘可清楚描述為內隱知識的萃取，將潛藏、有用處的知識挖掘出來(Witten & Frank, 2000)，知識是隱藏於資料之中，未知且必須仰賴資料探勘的自動化技術才得以萃取出的；反觀文字探勘，知識是外顯的並且明確表示在文字當中，不像結構化資料的知識是內隱於其中的，因為大多數的作者在撰寫文章時，都會儘量在文中明確表達想法，正因為如此，使得知識於文字中是外顯且明確的。但因為非結構化資料的關係，較難以演算法萃取其中的知識，儘管知識是明確表示在文件中的。

2.1.4 簡單貝氏分類器

單純貝氏分類器 (Naïve Bayes Classifier)是一種簡單且實用的分類方法，在某些領域的應用上，其分類效果優於類神經網路和決策樹。單純貝氏分類器依據貝氏定理(Bayes' Theorem)為基礎，結合各屬性間彼此獨立性事後機率(Posterior Probability)，在計算分類機率時，都是以種類型屬性為考量，並無法處理數值型的屬性。採用監督式的學習方式，分類前必須事先知道分類型態，透過訓練樣本的訓練學習，有效地處理未來欲分類的資料，用於大型資料庫，可以得出準確高且有效率的分類結果。

簡單貝氏分類器(Naïve Bayesian Classifier)的分類原理是透過某對象的先驗機率(或事前機率)，利用貝氏定理(Bayes' Theorem)計算其後驗機率，即該對象屬於某一類的機率，選擇具有最大後驗機率的類別作為該對象所屬的類別，是透過機率統計的計算，達到最小誤差的一種分類方式。

Clark & Niblett(1989)、Cestnik(1990)、Langley et al.(1992) 發現，簡易貝氏分類器的分類正確率與其它分類器相比是具有競爭力，簡易貝氏分類器亦常被其它分類器用作比較的對象。

第二節　雲端運算

雲端運算(Cloud Computing)為分散式運算技術的一種，核心概念是透過網絡將旁大的運算處理程式自動分拆成無數個較小的子程式，再由網路上多部伺服器所組成的龐大系統叢集經搜尋、運算分析之後將處理結果回傳給用戶，透過這項技術，面對數以千萬計的資訊時，能有效的縮短處理時間。雲端運算除了提供強大的運算能力之外，最終則是希望無須安裝任何的軟體，所有的資源(計算能力、儲存空間、應用服務.....等)皆來自雲端，使用者端只需要一個連上雲端的設備與簡單的介面(例如瀏覽器)即可。

2.2.1　雲端運算的定義

網路技術逐漸興起時，在畫示意圖時經常使用一朵雲代表網際網路。隨著網路的發展，除了個人電腦外，更多的設備都具備了上網的能力，例如手機、家電產品或辦公設備。網際網路的作用也不僅僅是瀏覽網頁、收發電子郵件，能夠為企業提供電子商務、客戶關係管理等服務；為一般使用者提供部落格、討論區、網路相簿等功能；為研究單位提供運算處理的能力。網路的定義變得更廣，除了連結、路由外，還包含了運算、儲存、服務和軟體等元素，雲端運算中的「雲」已經不只是代表網際網路而已，更廣泛地包含了被描繪在那朵雲之外的相關事物。

雲端運算中的雲強調對網際網路的使用，而非著重於其運作細節，包含網路、運算、儲存等基礎建設，及作業系統、應用程式平台、Web 服務等，重點在於資源的調度及運用，不是雲的運作細節。

雲端運算的定義百家爭鳴，內容包羅萬象，足見業界對雲端運算的重視，以下舉出數個普遍為人接受的定義。

維基百科(Wikipedia)定義雲端運算為將能夠動態延展(dynamically scalable)的虛擬化資源，透過網際網路提供服務給用戶的運算模式，像是電力網絡般透過網路將分享式的資源、軟體及資料，依需求提供給使用者。

國際研究暨顧問機構 Gartner 認為，雲端運算是三大趨勢匯流的產物，包含服務導向架構(Service Orientation Architecture)、虛擬化技術(Virtualization)和透過網際網路進行運算處理的標準(Standardization of Computing through the Internet)；雲端運算的興起，使得那些使用及販售資訊科技服務的用戶得以重新形塑，找到自己的市場機會。

依據美國國家標準和技術協會(National Institute of Standards and Technology, NIST)第15版的定義，認為雲端運算是一個能夠依需求取用分享的、可組態的資源的一種模式(如網路、伺服器、儲存設備、應用程式、服務)並以最小的管理成本進行提供(Mell & Grance, 2009)。

根據美國國家標準和技術協會的定義，雲端運算具有三種產業模式及四種部署模式：

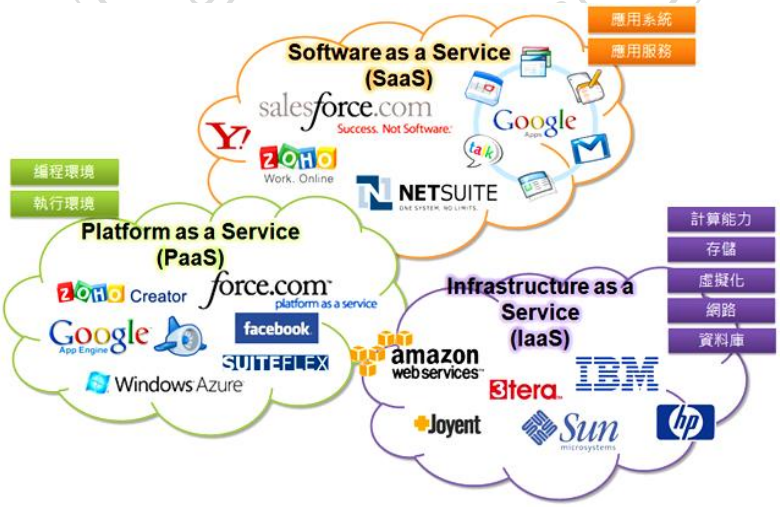


圖2-2 三種雲端運算產業模式示意圖

[資料來源：gipi (2009)]

三種雲端運算的產業模式，按其服務類型可分為以下三類：

圖2-3 雲端運算架構層次

[資料來源：陳瀅 (2010)]

1. 雲端軟體即服務 (Cloud Software as a Service, SaaS)

通過網際網路提供服務，提供用戶使用建構於雲端運算系統上的應用程式，這些應用建構在基礎建設即服務層提供的資源，以及平台即服務層提供的環境之上，讓服務藉由網路交付給用戶。

使用者對於雲端運算系統架構不具備管理及控制權限，僅可調整少部份的應用程式設定，像是 Google 文件，提供使用者線上文書處理，但使用者對於 Google 文件的相關設定自由度較低，以及微軟的線上客戶關係管理和 SharePoint、Adobe 的線上影像處理 Photoshop。

透過 Internet 提供軟體的模式，廠商將應用軟體統一部署在自己的伺服器上，客戶可以根據自己實際的需求，透過網際網路訂購所需的軟體服務，按訂購的服務多寡和使用時間的長短向廠商支付費用，並透過網際網路獲得廠商所提供的服務，亦無須對軟體進行維護，服務提供商會負責全權管理與維護軟體。

2. 雲端平台即服務 (Cloud Platform as a Service, PaaS)

介於基礎建設即服務層(IaaS)和軟體即服務層(SaaS)之間，提供平台給 IT 管理者和開發人員開發、運行、管理和監控的環境，支援特定的程式語言或工具，用以構建、測試及部署應用程式。即透過平台去支援整個產品開發的生命週期，從一開始的系統設計到最終的系統上線，都提供良好的支援，也可滿足雲端運算在擴充性、可用性和安全性的要求。

對於雲端計算系統架構亦不具備管理及控制權限，但部署應用程式時可調整的組態較 SaaS 多，像是 Google App Engine、Amazon S3、Salesforce.com 的應用開發平台和 Microsoft Azure 都屬於這類雲端運算產業模式，提供程式開發者雲端運算的平台，供開發者部署其應用程式，不需要管理機器及網路設定。

3. 雲端架構即服務 (Cloud Infrastructure as a Service, IaaS)

透過虛擬化技術，抽象化基礎建設的實體資源，實現內部流程自動化和資源管理優化。以虛擬化後的硬體和相關管理功能的集合，提供核心計算資源和網路架構的服務，使用者可以部署並執行任何軟體，包含作業系統及應用程式。

對於雲端計算系統架構亦不具備管理及控制權限，但對於作業系統、儲存資料及部署的應用程式有完整的控制權限。部份狀況下可擁有網路元件(像是防火牆)的組態調整權限，Amazon 為提供儲存空間所推出的簡單儲存服務(Simple Storage Service, S3)；為企業提供運算能力的 Amazon EC2 (Elastic Compute Cloud)；SQS(Simple Queue Service)則是針對小型企業和個人消費者提供網路通訊的服務；HP 所提供的 FCS2(Flexible Computing Services)則是為企業提供了運算能力和儲存

空間的基礎設施，以上皆屬於此類的服務。

硬體資源(如儲存空間)和計算能力(CPU 和記憶體)以應用服務的形式提供給使用者，如此一來對於專業伺服器與網絡設備，企業可有別於傳統花錢買設備的方式，改採租用的方法取得資源；使用者可依自身的需求擴展設備，並根據使用雲端資源的多寡進行付費。

雲端運算的部署模式為下列四種：

1. 私有雲 (Private Cloud)

   此雲端基礎建設僅限於公司內使用，因而提供對資料安全性及服務品質的要求最有效的控制。雲端服務由公司自行管理或委託第三方維護，這一種部署模式賦予公司對於雲端資源的使用狀態極高的控制能力，也使得企業具有建立及運作雲端服務環境所需的專業知識。

2. 社群雲 (Community Cloud)

   由數個組織組成之社群共同擁有的雲端基礎建設，雲端環境的建立及運作費用彼此分擔。雖然比起公有雲端費用較高，但享有較高的資料安全性，此雲端環境由組織自行管理或委託第三方維護。

3. 公共雲 (Public Cloud)

   提供大眾使用雲端基礎建設、與使用者相關的雲端服務，企業透過雲端服務提供商(Cloud Provider)可以帶給企業 IT 的效果包含降低成本、彈性、動態的儲存空間，並使得企業不必花費心思於伺服器的更新及管理上，專注於核心業務創造更多的獲利。

4. 混合雲 (Hybrid Cloud)

由上述兩種或兩種以上的部署模式組成，藉由特定的標準或專有的技術捆綁在一起。

雲端運算被視為 Web 2.0後下一個科技產業的重要商機，美林證券預估未來五年全球雲端運算的市場規模將可達到950億美元，佔全球軟體市場的12%；賈特那(Gartner)公司也預估2014年以前，全球企業運用雲端運算所獲得的營收將突破140億美元，2009年的統計顯示，光是利用雲端服務的營收，就可望達到75億美元，比起2008年的統計資料大幅提昇17.7%。

雲端運算的商機使各種科技大廠紛紛積極投入，不僅 IBM、微軟、Google、Amazon、Oracle、HP 和昇陽等公司展開佈局，台灣公司亦不落人後，例如廣達於2009年10月耗資一千萬美元取得美商 IC 設計公司 Tilera 的特別股，投入雲端運算晶片的開發；更有消息指出，工研院的雲端運算行動應用研究中心、趨勢科技、中華電信和資策會所成立的台灣雲端運算公司，將於2011年起運作，結合中華電信的網路資料中心(IDC)和電信網路、趨勢科技的安全產品、工研院的貨櫃資料中心和雲端作業系統，由資策會開發雲端服務及相關的應用，搶攻全球雲端運算的商機。

2.2.2 現有的雲端運算服務

針對現有的雲端運算服務，即 Amazon EC2、Google App Engine、Microsoft Azure 與 Yahoo Hadoop，彼此功能的差別(王耀聰、陳威宇, 2008)。

服務 屬性	Amazon EC2	Google App Engine	Microsoft Azure	Yahoo Hadoop
架構	Iaas/Paas	Paas	Paas	Software
服務型態	Compute/ Storage	Web application	Web and non-web	Software
管理技術	OS on Xen hypervisor	Application container	OS through Fabric controller	Map / Reduce Architecture
使用者介面	EC2 Command-line tools	Web-based Administration console	Windows Azure portal	Command line and web
APIs	yes	yes	yes	yes
收費	yes	maybe	yes	no
程式語言	AMI (Amazon Machine Image)	Python Java	.NET framework	Java,

圖2-4 雲端運算服務比較表

[資料來源：王耀聰、陳威宇 (2008)]

第三節 MapReduce 軟體設計模型

MapReduce 是一種用於簡化大量分散式計算概念的軟體設計模型，一個 MapReduce 系統負責分散式運算環境下機器之間的溝通與協調，而程式開發人員只需要撰寫 Map 及 Reduce 程式。複雜的程式可以撰寫為多個 Map 及 Reduce 程式的串接，不受限於單一機器的記憶體容量，適用於大量資料的分析，利用這個模型所設計的軟體，會自然呈現平行運算的形式，可以運算叢集自動分散運算。分散運算的機制可以透過執行期系統(Run-time)來決定，例如資料輸入、任務執行排程、錯誤控制與必須的叢集節點間的溝通等，換言之，開發人員可以不需要考量任何分散式程式開發工作，讓平行運算變得更容易開發。

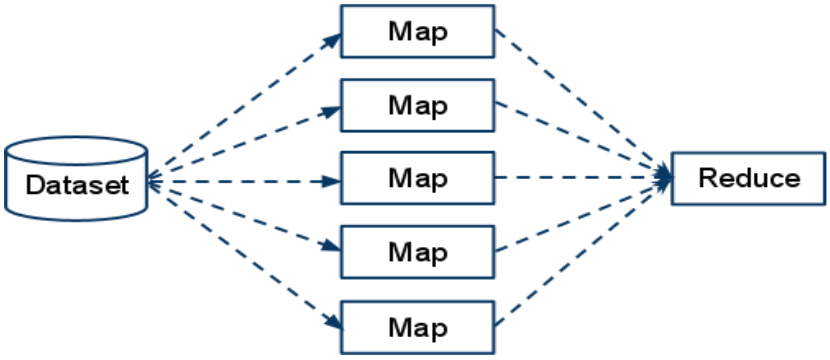


圖2-5 MapReduce 軟體設計模型示意圖

在這個架構之下，因為分散式架構的特性，MapReduce 可以用來處理幾兆位元組(TeraByte)的資料，並且可以輕易的將運算任務交由運算叢集來運算，這個運算叢集將是可以任何擴增與縮減的。Google 所推出的雲端服務中，許多是透過這個軟體設計模型進行開發，可以輕易的達到雲端運算的要求與優點，可以說 MapReduce 是一種符合雲端運算架構要求的演算法，只要軟體架構能夠以這種軟體設計模式進行開發，將可以降低平行運算軟體開發上的困難(Dean & Ghemawat, 2004)。

MapReduce 適合用來實做平行處理大量資料，顧名思義 MapReduce 是由 Map 程式與 Reduce 程式所組成，輸入一組 Key/Value 組合當作輸入資料，Map 程式產生許多組 Intermediate Key/Value，然後再由 Reduce 程式作相同 Key 的資料合併，產生最後結果。

Map 程式：

map(inKey, inValue) → list(outKey, intermediateValue)

Reduce 程式：

reduce(outKey, list(intermediateValue)) → list(outValue)

MapReduce 架構的示意圖如圖2-6所示：

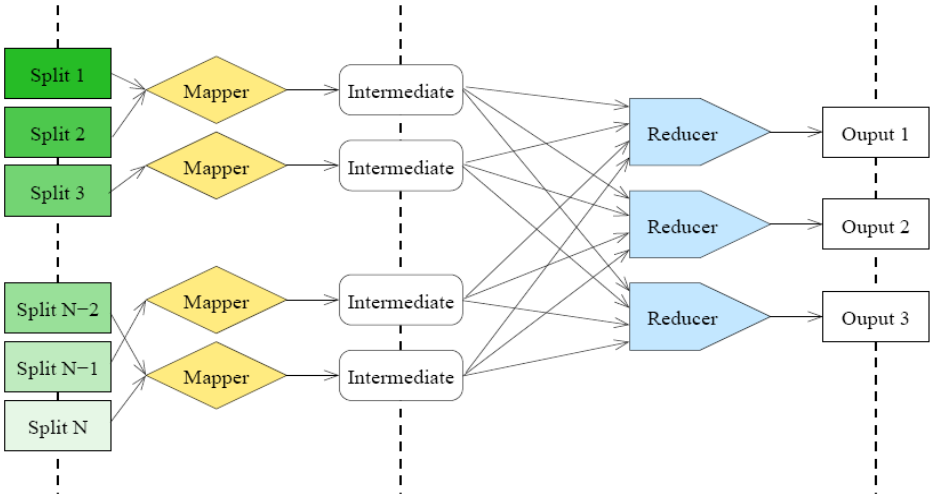


圖2-6 MapReduce 軟體設計模型運作過程示意圖

[資料來源：Papadimitriou & Sun (2008)]

2.3.1 Google 的分散式資料庫 Bigtable

Bigtable 為 Google 為了存放大量資料所設計的分散式儲存系統，為 Slave/Master 架構，由 Namenode 存放目錄架構，Datanode 儲存資料區塊，每筆資料皆備份三份置放於不同的 Datanode 中，能夠達到容錯的功能。目前網路上已有 Cassandra、HBASE、Hypertable、CouchDB 等實做 Bigtable 觀念的 Column-based 資料庫。

儲存方式類似 Excel，但每個資料格可以時間戳記區隔存放多筆資料，取出資料的方式為(row: string, column: string, time:int64) → string，且具備 Column Family 的特性，Column Family 為儲存的單位，一個 Column Family 能夠包含多個 Column，實際存放方式如圖2-7及圖2-8所示。

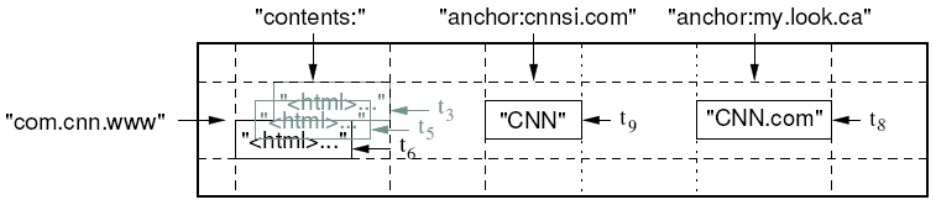


圖2-7 Bigtable 資料儲存之示意圖

[資料來源：Chang et al. (2006)]

Row Key	Time Stamp	Column *"contents:"*
"com.cnn.www"	t6	"<html>..."
	t5	"<html>..."
	t3	"<html>..."

Row Key	Time Stamp	Column *"anchor:"*	
"com.cnn.www"	t9	"anchor:cnnsi.com"	"CNN"
	t8	"anchor:my.look.ca"	"CNN.com"

圖2-8 Bigtable 資料實際存放的基本單位為 Column Family

[資料來源：Chang et al. (2006)]

另有和 Column-based 資料庫架構上相似、和 Column-based 資料庫獲得相當程度的關注的 Key-Value 資料庫，為 Column-based 資料庫的簡化版，沒有提供 Column Family 的功能，其高效率、易用性及彈性使 Key-Value 資料庫的發展逐漸蓬勃。

Key-Value 資料庫與傳統的關聯式資料庫不同之處主要有以下三點：

1. Column-based 資料庫的屬性可以靈活增刪，增加一個新的屬性可以直接在資料寫入的程式中多加該屬性即可，而關聯式資料庫在新增屬性時必須要鎖定住該資料表，資料量大的資料表將會離線幾十分鐘甚至一天。

2. 允許文件之間擁有的屬性不一致，有些資料有 A 屬性而有一些沒有，關聯式資料庫只能將其設定為 Null 值或0，影響儲存和資料查詢的效率。

3. 資料都是以附加(Append)方式寫入的，以版本號碼或時間戳記作為最新資料的辨認，特點是可以追溯資料各版本間的內容變化。因為其附加寫入方式的複雜度較低(logN，觸發1次寫入資料庫及 logN 寫入 B+ Tree 節點)，因此效率較佳。

2.3.2 MapReduce 軟體設計模型應用於資料探勘之研究

Chu et al. (2006)將 Google 所提出的 MapReduce 軟體設計模型應用於資料探勘演算法的設計上，用以分散式處理提高機器學習的效率，並透過實驗證明 MapReduce 軟體設計模型應用在資料探勘演算法的實作上，增加處理器核心的數目基本上可以讓執行速度呈現線性提昇。

Wegener et al. (2009)提到目前尚未出現可在運算叢集上運作的資料探勘工具，因此提出一系統架構將 Weka 這套資料探勘工具整合於 MapReduce 的運算叢集上，內容包含運算模型及資料儲存模型，藉此突破資料探勘於單一機器上執行的實體記憶體限制，並透過實驗指出該系統架構可使大型資料集的資料探勘執行效率提高。

第四節 實作 MapReduce 架構的框架

目前實作 MapReduce 架構的開放原始碼框架非常多，各自使用不同的程式語言撰寫核心程式，表2-1為目前網路上流通的 MapReduce 開放程式碼框架及其核心程式語言，並且針對主要的 MapReduce 框架 Hadoop、DisCo 作簡要介紹：

表2-1 MapReduce 之開放程式碼框架及其實作程式語言

框架名稱 Framework	**核心程式語言** Programming Language
Hadoop	Java
DisCo	Erlang
Octopy	Python
Starfish	Ruby
Skynet	Ruby
Phoenix	C

[資料來源：本研究整理]

2.4.1 Hadoop

Hadoop 是一項由 Apache 軟體基金會所發起的軟體專案，其為一個開放原始碼分散式運算系統軟體平台，以 Java 實做，讓開發人員可以輕易的在這個架構上運算大量的資料，協助大資料集的分析，並具備有可擴充性(Scalable)、經濟(Economical)、有效率(Efficient)與可信賴(Reliable)等優點。這個軟體平台基於 MapReduce 演算法與 HDFS(Hadoop Distributed File System)檔案系統之上，HDFS 建立可信賴的儲存叢集，而 MapReduce 演算法將應用程式分割成小塊，並利用這個龐大的資料叢集進行運算。Hadoop 已經證實可以支援多達25000台電腦所構築的叢集，目前也有多個學術單位與 Hadoop 進行合作，利用這個平台架構應用在不同的領域上。

Hadoop 是目前最常被使用到的 MapReduce 軟體，其中最主要的元件為 MapReduce 軟體設計模型跟分散式檔案系統 (HDFS)，Hadoop 基於資料片段實際存放於 HDFS 的位置，分配工作給各節點進行分散式運算。

Hadoop 主要的內容有 HDFS (Hadoop Distributed File System)及 MapReduce 軟體設計模型的實作，程式開發者繼承其 MapReduce Class 開發自己欲應用的

程式，因大部分程式語言在 Java Platform 上都有實做版本，使用 Java 當作程式語言意即等同支援大多數語言，目前是最多開發者使用的 MapReduce 軟體設計模型。趨勢科技的2009騰雲駕霧程式競賽即是採用 Hadoop 當作 MapReduce 軟體設計模型的雲端運算開發環境。

Nick Jenkin(2009)認為資料探勘演算法實做於 MapReduce 軟體設計模型上存在著許多困難，並非所有的資料探勘演算法都適合移植到 MapReduce 軟體設計模型上。因此他的研究提出了使用一般性分散式運算的方式取代 MapReduce 軟體設計模型上執行移植過去的演算法之見解，針對無法適用於 MapReduce 軟體設計模型上的資料探勘演算法，是個相當良好的方向，得以透過一般性分散式運算的技術將演算法作分散式的處理，避免單一機器硬體規格的種種限制。

目前 Apache 軟體基金會另外成立了 Mahout 專案，用於建立一個具備擴充性的機器學習函式庫，將演算法實作於 Hadoop 雲端運算環境上，以 MapReduce 軟體設計模型實作，使得演算法具備處理大量資料及分散式運算的能力。這是一個開放原始碼的專案，可以依照需求針對其中演算法的程式碼做調整，減少演算法移植過程的大量開發時間。

2.4.2 DisCo

DisCo 為 Nokia 研究中心所發展出來的 MapReduce 系統，核心程式採用 Erlang 撰寫，值得注意的是開發者所撰寫的 Mapper 及 Reduce 函式皆限定為 Python 撰寫，因為 Erlang 本身即具有平行處理的語言特性，適合處理 MapReduce 軟體設計模型的分散式運算工作，操作上較為靈活。

2.4.3 使用雲端運算平台的其他方案

Amazon Web Services in Education 為 Amazon 所提供的 AWS 之教育版，提供研究人員一定的免費額度，讓他們得以使用 AWS 的服務進行研究環境的建置及運算的平台，唯一可惜的地方是針對學生所提供的版本功能非常有限，且沒有包含 Amazon Elastic MapReduce 雲端運算功能。

RIGHTSCALE 為一家提供使用者免費使用 Amazon Web Services 的公司，但免費期間僅為10 CPU 小時，超出免費使用額度的部份則必須另外付費，因此需要輸入信用卡號作為扣款帳戶資訊。

Google App Engine 為 Google 所推出的服務，使用者能夠使用 Google App Engine 所提供的 API 進行程式的開發，將程式及資料都儲存於 Google 的雲端平台上，開發者於 Google App Engine 開發程式時將受限於 API 提供的功能及 CPU、資料空間的免費額度。

第五節　現今語意網的相關發展

語意網相關技術的發展已持續多年，逐漸從學術界熱烈探討的議題轉換為實務上的應用，許多網站皆想要透過實現語意網的技術提供與眾不同的服務。本研究整理幾個現今實現語意網相關技術的案例，分別為鏈結資料(Linked Data)、Facebook 和 Wolfram Alpha 語意搜尋引擎。

2.5.1　鏈結資料 (Linked Data)

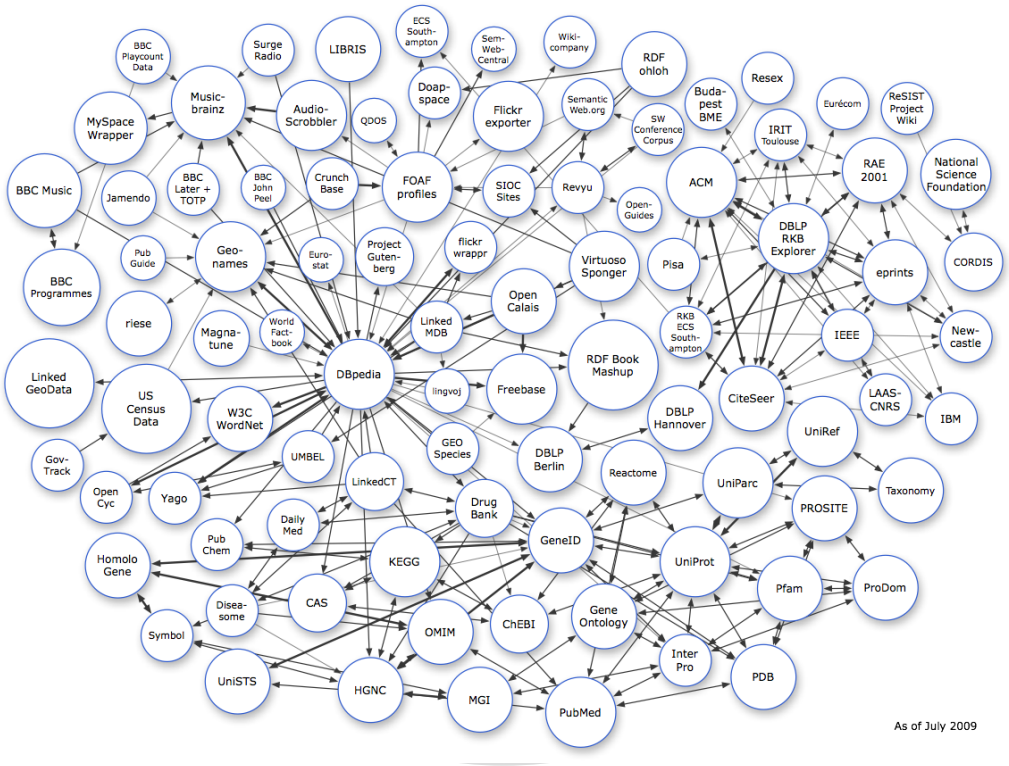


圖2-9 鏈結資料目前加入網站示意圖

［資料來源：Linked Data］

由網路之父 Tim Berners-Lee 所提出的概念，將資料發佈在網際網路上提供其他網站瀏覽搜尋，將傳統的單一資料來源透過鏈結資料的概念成為聯合的資料來源，鏈結資料作為一種新的資料傳輸中介，連結各網站的結構化資料，並使其能為機器所閱讀，其他網站定義過的資料綱目及其內容可透過 URI 及 RDF 連結使用，存取相關網路資源的內容。

鏈結資料所提倡的是語意化網站出版(Semantic Web Publishing)，將網站內容透過語意化的標籤發佈在網際網路上，提供語意化的脈絡，使得電腦可以禮節這些非結構化資料所包含的結構化資訊，使得資料的搜尋以及整合更為有效。

傳統的關聯式資料庫若欲轉換為鏈結資料的資料格式，可透過 D2R 函式庫轉換，資料格式不必重新定義，使得關聯式資料庫的內容得以迅速與鏈結資料中的資料來源結合，擴大資料的內容。

2.5.2 Facebook

Facebook 在 F8開發者大會中，提出開放圖譜協議(Open Graph Protocol)概念，使得其他網站得以與 Facebook 連結，將使用者的資料集中在一起，使其精準分析使用者的偏好，例如使用者在網路電影資料庫(IMDb)網站中瀏覽喜歡電影的條目，點選「讚(Like)」鈕即可將這筆喜好記錄傳送至 Facebook 作為電影的偏好記錄。

資料量日漸增多後即可作為更進一步的分析應用，對於廣告商可提供精準行銷的相關服務，Facebook 亦可挾此優勢踏入搜尋領域，搜尋開放圖譜協議上所有節點的相關資料，搜尋結果的排序亦可加入使用者點選「讚」鈕的數量，應能提供更為精準的搜尋結果。

Facebook 提供的 Open Graph Protocol 是透過 RDFa 的資料格式去實現的，透過內嵌於 HTML 的內容描述資料的相關屬性。

關於 Facebook 所提出的開放圖譜協議，有幾點尚為人所批評：

1. 開放圖譜協議尚未具備消除描述事物歧義的能力，例如在 IMDb 網站上，有許多電影名稱相同的電影，但在描述內容時無法表示兩者的不同之處，造成原本的電影及翻拍的電影，將會辨識為同一部電影，對於精確的使用者喜好之蒐集造成不好的影響，以及當使用者點選「讚」

鈕時，必須要撰寫特別的處理程式才能分辨電影和演員之間的差別。

但這些問題都已經隨著開放圖譜協議逐漸成熟而解決，關於消除事物歧義能力的部分，IMDb 網站目前在電影名稱後面附加了拍攝年份，儘管電影名稱相同，拍攝年份不同即表示不同的電影，如此一來原創的電影和翻拍的電影之間便獲得了區隔；IMDb 網站先前配合開放圖譜協議所製做的詮釋資料僅包含 Facebook 應用程式編號(fb_app_id)、標題(og:title)和網站名稱(og:site_name)，無法做到電影、演員以及其他電影幕後人員的區分，但隨後加上了類型(og:type)的詮釋資料欄位，因此當使用者點選「讚」鈕時，Facebook 可以知道使用者是喜愛該電影或是該演員，可更為精準蒐集使用者偏好資料。

IMDb 網站中有關電影的描述：

```
<meta property="og:type" content="movie" />
<meta name="fb_app_id" content="115109575169727" />
<meta name="og:title" content="Star Trek (2009)" />
<meta name="og:site_name" content="IMDb" />
```

IMDb 網站中有關導演的描述：

```
<meta property="og:type" content="actor director" />
<meta name="fb_app_id" content="115109575169727" />
<meta name="og:title" content="J.J. Abrams" />
<meta name="og:site_name" content="IMDb" />
```

2. 開放圖譜協議不支援在同一網頁中描述多重物件，僅可表示整個網頁是描述一個人物、新聞事件、音樂家或者是一部電影，無法標註網頁內容中的個體。

3. 目前配合開放圖譜協議的網站，並未依照開放圖譜協議的規定內容實作詮釋資料，例如 IMDb 在描述具備多重身份的人時，會在類型欄位

(og:type)將兩個身份連結起來，像是上述所提到的導演兼演員(actor_director)，這是目前 Facebook 開放圖譜協議沒有支援的物件類型，使得 Facebook 得針對 IMDb 等網站所傳送過來的資料重新解析，才能獲得正確的資訊。

4. Facebook 自身的網頁並沒有依照開放圖譜協議的內容撰寫詮釋資料，反而要求其他的網站依照開放圖譜協議製作詮釋資料。

5. 目前 Facebook 上許多使用者資料是混亂不清且有許多是重複的，若依照開放圖譜協議製作詮釋資料，務必要先排除這些重複的物件，使得資料的完整性具備。

2.5.3 Wolfram Alpha

Wolfram Alpha 為一答案引擎(Answer Engine)，與我們習以為常的搜尋引擎(Search Engine)不同，它並不提供包含關鍵字的相關網頁列表，而是提供一個整合式的結果，即 Wolfram Alpha 針對問題所認定的答案。

Wolfram 的實作原理為使用者遞交查詢內容，可以為一般的自然語言問句(例如英文，目前尚不支援中文)，與傳統的語意搜尋引擎不同，Wolfram Alpha 將大量的問題建立索引，分析使用者的查詢需求後，將查詢需求與答案配對，找出最為合適的答案作為查詢的結果。

第六節　語意註解 (Semantic Annotation)

語意註解為資料得以被理解且以更新的存取方式流通，為語法(Syntax)及資料格式(Structure)增加意義 (Kaarthik Sivashanmugam et al., 2003)，詮釋資料(Metadata)的種類如圖2-10所示。

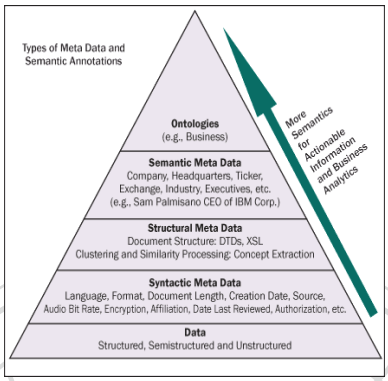


圖2-10 詮釋資料的種類

[資料來源：Kaarthik Sivashanmugm et al. (2003)]

詮釋資料分為語法詮釋資料(Syntactic Metadata)、結構詮釋資料(Structural Metadata)及語意詮釋資料(Semantic Metadata)。

1. **語法詮釋資料 (Syntactic Metadata)**

   資料來源的詳細描述資料，例如資料使用的語言、資料產生的時間、資料標題、資料大小、資料格式等等。

2. **結構詮釋資料 (Structural Metadata)**

   用以描述資料的結構，利於資料的儲存、處理、呈現及資料檢索，例如 XML 綱目(XML Schema)。

3. **語意詮釋資料 (Semantic Metadata)**

   描述和資料內容相關的訊息，以特定領域的本體論產生，透過語意詮釋資料可提供機器解讀資料內容的可能性，並提供更進一步的應用，比起語法詮釋資料及結構詮釋資料更能提供有意義的描述資料。

語意註解為原始資料產生語意詮釋資料，使得資料得以更新的存取方式流通，註解綱目是以資料檢索系統所萃取出的資訊建構而成。透過語意註解使文件之間得以相互連結，並具備語意網的特性，簡而言之，語意註解是在分析的文章中針對特定文字指派實體(Entity)和關係(Relation)，使其成為一個本體論具備描述自身的能力(Davies et al., 2006)。

語意註解是為了辨認出文件中有意義的詮釋資料 (像是個體、關聯等)，達到網路文件語意化的目的，使得電腦能夠透過理解文件內容，針對使用者搜尋找出有用而正確的資料，這種文件註解方式已經逐漸普遍。

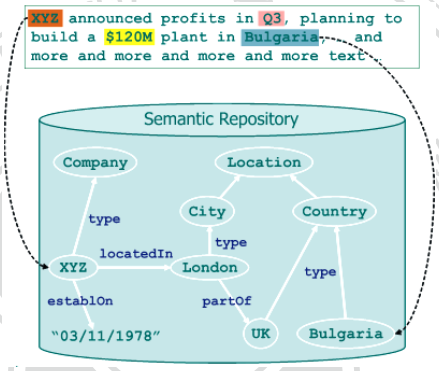


圖2-11 語意註解示意圖

[資料來源：John Davies, et al. (2006)]

語意註解的另一個用途為豐富文件的內容，為純文字的檔案加入相關的連結，使用者可以透過這些連結獲取更多相關的知識，目前已有 Inform Engine 將此功能實現。

透過語意註解的技術可幫助語意網的普及化，實現語意網的願景，使得網路上的文件可以相互連結，文件以機器可以閱讀的格式；使機器得以理解文件的內容，未來可以用作自然語言查詢，提供更為精準的查詢結果。

2.6.1 語意註解的分類

語意註解系統分為 Web-based 與 Ontology-based 兩種，前者透過網際網路使用語意註解工具，結合網頁內容及語意註解資訊後再顯示給使用者，後者則是利用本體論的優點達到語意註解自動化，將網頁內容與語意註解系統的本體論作映射，達到語意註解的功能。本研究所使用的語意註解系統即為 Ontology-base 語意註解系統。

另有一種語意註解的類型，稱作「協同註解」，又被稱為「社會標籤(Social Tagging)」和「分散式分類(Distributed Classification)」，集合眾多個人對網路資源進行標記、以詞分類的新興資訊組織方式。這種自由標記與傳統以專家為基礎的圖書資訊組織方式相當不同，目前當紅的 Delicious 網站即是以協同註解的方式，提供使用者創造、分享及結合彼此的註解，達到更高的搜尋精準度，同樣類型的網站還有 Flickr、CiteULike, Youtube 和 Last.fm 等。

語意註解的使用有助於提昇資訊檢索(Information Retrieval)的效率及精準度，舉例來說，文件中各個個體之間的關係可用來實做以個體為基礎的資訊檢索系統(又稱作垂直搜尋系統, Vertical Search Engines)。

2.6.2 語意註解的發展

紐約時報目前使用了大量的表頭詮釋資料(Metadata)去描述新聞；路透社也發布了 Open Calais API，透過自動化語意註解 HTML 文件，目的在於改良資料處理的效能及資料搜尋的精準度。

2.6.3 語意註解於資料探勘上的應用

Berendt et al. (2002)提出語意網路資料探勘概念(Semantic Web Mining)，結合語意註解與資料探勘，將語意註解產生的語意詮釋資料用於建立語意網，使得語意網的應用得以存取這些原先非結構化的網頁文件；使得這些網頁文件的知識得以自動化管理，並用於強化資料探勘的結果，提昇分析結果的正確度。

Laclavik et al. (2008)將 MapReduce 軟體設計模型應用於語意註解技術，將語意註解運算處理分散至運算叢集中的機器中，藉由實驗證明語意註解可以運行在使用 Hadoop 建立的運算叢集上。將 MapReduce 軟體設計模型應用於語意註解演算法的實作，並較單機進行語意註解的效率提高許多，資料集的檔案大小越大效率提昇越明顯。

第三章 研究方法

根據文獻探討，在對雲端運算、文件分類、簡單貝氏分類器及語意註解做完大略的介紹後，本研究將提出將簡單貝氏分類器演算法移植到 Hadoop 這套 MapReduce 軟體上的雛型架構，使得資料探勘得以在雲端運算的環境下運作。

本研究將以簡單貝氏分類器及餘集簡單貝氏分類器進行文件分類，以公開的資料集 Reuters 21578作為實驗上建模的測試資料。將路透社資料集依照 Mod Apte 切分法將資料分為訓練資料集與測試資料集，驗證文件的分類正確率，以測試資料集最佳的分類正確率作為該實驗的實證結果；測試本研究提出的雲端文字探勘平台的分類結果，是否將運算需求分散給各節點平行運算達到負載平衡(Load Balance)，提昇文字探勘的分析效率，用以針對本研究建立的雛型系統進行驗證以示本研究提出的雲端文字探勘平台之可行性，透過大型資料集評估雲端文字探勘平台之正確率。

實驗中探討語意註解技術應用於文字探勘對於分類結果的影響，將資料集進行語意註解，擷取文件中的重要內容產生語意化資料，作為詮釋資料附加於文件內容中，使文字探勘演算法在運算過程中獲得更多的資訊，改善文件分類效果。

第一節　研究設計

本研究以 Hadoop 作為雲端運算系統平台，用於分散式文字探勘及結果分析，使用 Mahout 的演算法作為文件分類演算法。文字探勘分為兩個部份，分別為分類器(Classifier)與評估器(Evaluator)，兩者以模型(Model)連接，模型是由分類器以訓練資料建模而成，過程中產生的資料均存放於 HDFS 中，以供存取。

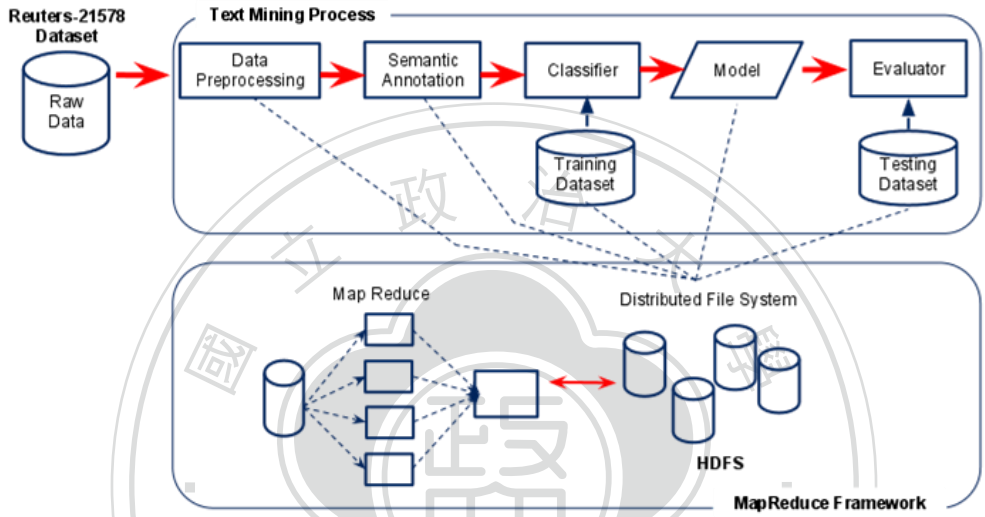


圖3-1 本研究之系統架構圖

[資料來源：本研究整理]

本研究之系統的資料輸入為路透社的 Reuters 21578資料集，透過文字探勘的資料前置處理步驟(Data Preprocessing)，資料集切分為訓練資料(Training Set)及測試資料(Testing Set)。簡單貝氏分類器根據訓練資料建模產生模型，接著將測試資料用於驗證模型的準確度。分類的結果將透過語意註解使其具備語意網的自我描述能力，作進一步的語意網應用。

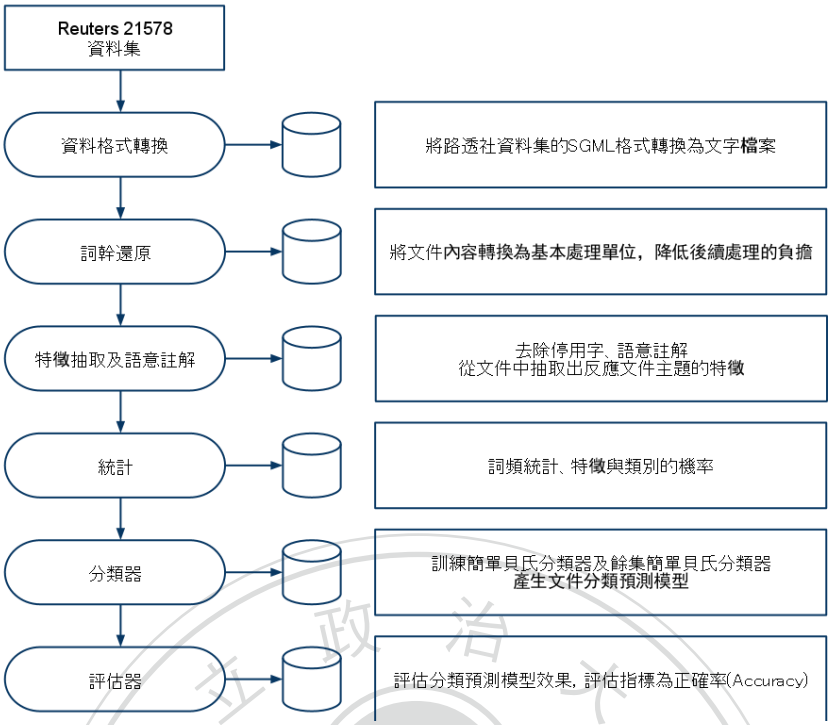


圖3-2 本研究之文字探勘流程圖

[資料來源：本研究整理]

文件分類的流程詳述如圖3-2，文件分類為分析待分類文件的特徵，並與已知類別中文件所具備的共同特徵進行比較，然後將待分類文件歸類為特徵最接近的類別。本研究之文字探勘可細分為資料格式轉換、詞幹還原、特徵抽取及語意註解等步驟，其中各模組為：

1. 資料格式轉換：路透社資料集為 SGML 格式，將其轉換為文字資料，格式化成為　統一格式，便於後續處理。
2. 詞幹還原：還原字根，統一詞性及時態上的變化，避免同義字或相同字詞的變化形擁有不同的編碼而產生困擾。
3. 特徵抽取及語意註解：去除停用字並應用語意註解技術增加與該文件之主題相關的詮釋資料，並從文件中抽取出反應文件主題的特徵。
4. 統計：進行詞頻統計，計算 TF-IDF 值，以及特徵與類別的機率。
5. 分類器：訓練簡單貝氏分類器與餘集簡單貝氏分類器，產生文件分類預測模型。

6. 評估器：針對分類器的預測結果進行分析。

以下各節將針對各主要模組進行更為詳細的說明。

第二節　文本資料集 Reuters 21578

3.2.1　路透社資料集文件格式

Reuters 21578是由路透社(Reuters Newswire)自1987年2月26日到1987年10月9日間收集的新聞文件，該資料集涵蓋21578篇新聞文件，由22個 SGML 格式的檔案所組成，每一個檔案各包含1000篇文件，最後一份則僅有578篇文件。如表3-1所示，該資料集中每一篇文件都以標籤＜REUTERS＞做為一篇文件起始，並以</REUTERS>作為一篇文件的結束。

表3-1 Reuters 21578資料集的標籤

<REUTERS TOPIC=?? LEWISSPLIT=?? CGISPLIT=?? OLDID=?? NEWID=???>

[資料來源：本研究整理]

文件大多已經經過人工區分類別，TOPIC 屬性紀錄該文件是否存在主題；LEWISPLIT 屬性則是為了符合 Modified Apte split 訓練-測試文件集區分法，這個版本的資料集剔除了 Reuters 21578資料集中沒有標註類別的文件，並且選擇至少有一個訓練資料和測試資料的類別而得到的資料集，路透社資料集中的文件皆以專家事先進行分類；OLDID 屬於早期的 Reuters 22173資料集編號，目前已經不使用而僅作為新舊資料集的；NEWID 代表文件編號；CGISPLIT 不在本研究考慮的屬性中，屬於另一種切分資料集的方法。

表3-2 Reuters 21578文件的範例

<REUTERS TOPICS="NO" LEWISSPLIT="TRAIN" CGISPLIT="TRAINING-SET" OLDID="16322" NEWID="1002">
<DATE> 3-MAR-1987 09:19:31.96</DATE>
<TOPICS></TOPICS>
<PLACES><D>usa</D><D>taiwan</D></PLACES>
<PEOPLE></PEOPLE>
<ORGS></ORGS>
<EXCHANGES></EXCHANGES>
<COMPANIES></COMPANIES>
<UNKNOWN>
&#5;&#5;&#5;G
&#22;&#22;&#1;f0295&#31;reute
d f BC-TAIWAN-REJECTS-TEXTIL 03-03 0137</UNKNOWN>
<TEXT>&#2;
<TITLE>TAIWAN REJECTS TEXTILE MAKERS EXCHANGE RATE PLEA</TITLE>
<DATELINE> TAIPEI, March 3 - </DATELINE><BODY>Central bank governor Chang Chi-cheng
rejected a request by textile makers to halt the rise of the
Taiwan dollar against the U.S. Dollar to stop them losing
orders to South Korea, Hong Kong and Singapore, a spokesman for
the Taiwan Textile Federation said.
He quoted Chang as telling representatives of 19 textile
associations last Saturday the government could not fix the
Taiwan dollar exchange rate at 35 to one U.S. Dollar due to
U.S. Pressure for an appreciation of the local currency.
The Federation asked the government on February 19 to hold
the exchange rate at that level.
The federation said in its request that many local textile
exporters were operating without profit and would go out of
business if the rate continued to fall.
Reuter
&#3;</BODY></TEXT>
</REUTERS>

[資料來源：本研究整理]

本研究的實驗，採用 Modified Apte Split 區分法，將資料集做出訓練、測試以及不使用的三個主要集合，分別有 9603 篇、3299 篇及 8676 篇。每個類別中的數量不一，有 90 個類別中的文件。

3.2.2 路透社資料集的歪斜資料特性

路透社資料集依據 Mod Apte Split 資料集切分法，將資料切分為訓練資料集、測試資料集及不使用資料集三個部分，分別具有9603筆、3299筆及8676筆資料，總共具有90個類別，本研究採用52個類別作為實驗所用的資料集，藉此避免資料集過大，無法在單一機器上執行文件分類演算法的情況發生。

表3-3 路透社資料集的資料分佈

序號 #	類別 Class	訓練資料數目 Train Set #	測試資料數目 Test Set #	資料總數 Total #
1	earn	2840	1083	3923
2	acq	1596	696	2292
3	crude	253	121	374
4	trade	251	75	326
5	money-fx	206	87	293
6	interest	190	81	271
7	money-supply	123	28	151
8	ship	108	36	144
9	sugar	97	25	122
10	coffee	90	22	112
11	gold	70	20	90
12	gnp	58	15	73
13	cpi	54	17	71
14	cocoa	46	15	61
15	grain	41	10	51
16	alum	31	19	50
17	jobs	37	12	49
18	reserves	37	12	49
19	copper	31	13	44
20	ipi	33	11	44
21	rubber	31	9	40
22	iron-steel	26	12	38
23	nat-gas	24	12	36
24	bop	22	9	31
25	veg-oil	19	11	30
26	tin	17	10	27
27	cotton	15	9	24
28	wpi	14	9	23
29	orange	13	9	22
30	retail	19	1	20
31	pet-chem	13	6	19
32	gas	10	8	18
33	livestock	13	5	18
34	housing	15	2	17

35	strategic-metal	9	6	15
36	lei	11	3	14
37	zinc	8	5	13
38	carcass	6	5	11
39	fuel	4	7	11
40	income	7	4	11
41	lumber	7	4	11
42	heat	6	4	10
43	lead	4	4	8
44	meal-feed	6	1	7
45	dlr	3	3	6
46	instal-debt	5	1	6
47	potato	2	3	5
48	tea	2	3	5
49	cpu	3	1	4
50	nickel	3	1	4
51	jet	2	1	3
52	platinum	1	2	3
	Total	**6532**	**2568**	**9100**

[資料來源：本研究整理]

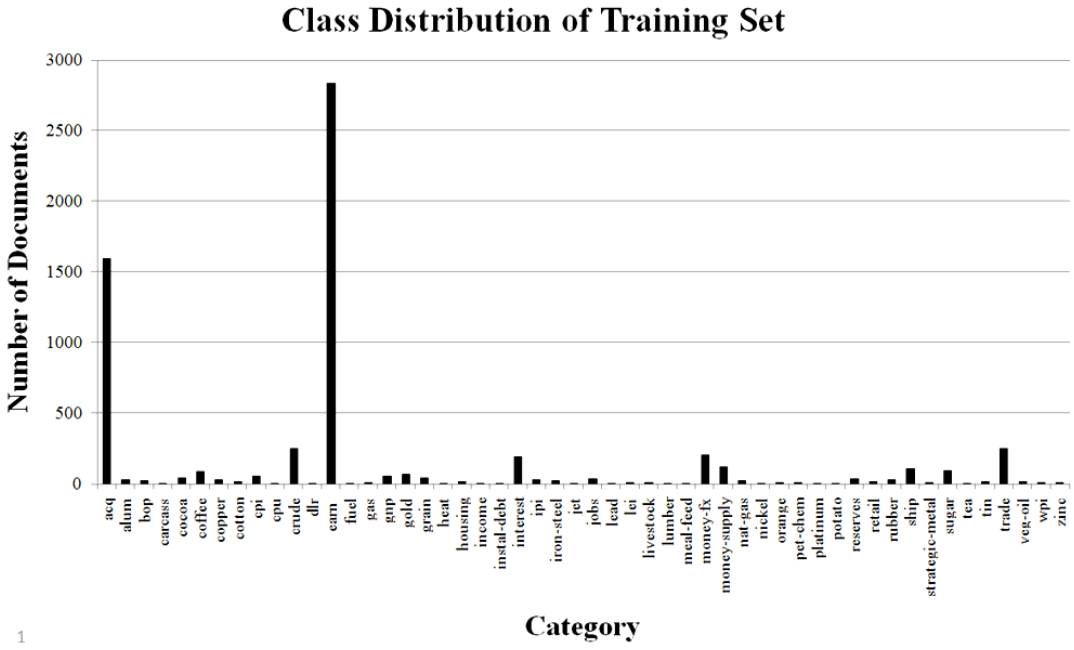


圖3-3 路透社資料集之訓練資料的類別分佈狀態

[資料來源：本研究整理]

圖3-3為路透社資料集中 52 個類別資料分佈的狀況，可以發現大多數的訓練資料皆分佈在 acq 及 earn 這兩個類別，其餘類別的資料筆數相較之下少了很多，是極不平均的類別分佈，依據歪斜資料的定義，訓練資料之類別的機率分

佈不對稱，亦即某些類別的資料筆數較其他類別多時，我們即可稱此資料集具備歪斜特性，而訓練資料的歪斜特性，將對簡單貝氏分類器的分類正確度造成負面影響，第四章中的試驗將會驗證這一點。

第三節　資料前置處理 (Data Preprocessing)

3.3.1　斷詞切字 (Tokenization)

將語句拆解為單字，包含空白及標點符號等等，表3-4為斷詞切字的示例。

表3-4 斷詞切字的輸入及輸出示例

輸入: Jerseys, dresses, wigs, and inflatable crowns.
輸出: < Jerseys, dresses, wigs, and, inflatable, crowns>

[資料來源：本研究整理]

3.3.2　去除停用字 (Stop Words Removal)

字詞分為兩種：功能字詞(function)與內容字詞(content)。前者常用於表示文法結構，像是 the, to 和 and 等常出現文件中的字詞；後者給予文件的內容含意，像是「The prancing blue cat is on a snowboard」中即包含了四個內容字詞(prancing, blue, cat, snowboard)與四個功能字詞(the, is, on, a)，分別表示內容及文法結構。

功能字詞通常皆包含於停用字列表(Stoplist)內，停用字列表中的字詞在文字探勘中是被忽略的字詞，像是「the」常出現在文件中，但僅以功能字詞的角色存在，對於文件內容的意義貢獻不高，不具備鑑別力。

去除常用詞 (如 a, an, and, as, ..., was, were, with 等字)，因為這些詞出現在文件中的頻率相當高，故不具鑑別力，反而會成為分析過程中雜訊(Noise)的來源，故需將停用字去除，除此之外，去除停用字亦可減少特徵空間的維度以提昇分類器的速度。

3.3.3 詞幹還原 (Stemming)

還原字根，統一詞性及時態上的變化，若字詞沒有還原其字根，往往會使得同義字或相同字詞的變化形擁有不同的編碼而產生困擾，譬如 automate, automatic, automation 屬於同義字，必須表示為同一字。

為了達到這樣的目的，因而產生還原字根演算法，除了能夠解決時態問題外，也可以解決大小寫問題及將同義字的不同詞性歸類為同義字，1980年由 Martin Porter 提出波特詞幹還原演算法(Porter Stemming Algorithm)，是一個相當有效率的還原字根演算法，因此本研究使用 Porter 所提出的波特詞幹還原演算法進行還原字根的處理。

第四節　語意註解

本研究採用路透社所支持的開放應用程式介面 OpenCalais，藉由路透社所研發的語意技術與自然語言處理器，以它的語意化標籤能力執行語意註解，以統一的 RDF 格式作為語意註解資料的格式。

將路透社資料集的文件分析後產生語意註解資料，再將語意註解結果作為該文件的詮釋資料，用以在文件分類演算法執行時提供額外的資訊，提高分類正確率。

表 3-5 語意註解輸入資料範例

ireland put colorado beetle alert irish agriculture department issued colorado beetle alert today beetles found box parsley imported france officials colony black and amber coloured beetles can destroy potato field day females lay eggs each boxes parsley consignment had distributed markets and department called shopkeepers and catering trade alert

[資料來源：本研究整理]

表3-6 語意註解輸出資料範例

entities [{

'_typeReference': 'http://s.opencalais.com/1/type/em/e/Organization',

'_type': 'Organization',

'name': 'irish agriculture department',

'__reference': 'http://d.opencalais.com/genericHasher-1/52c9469d-6d45-3b43-8074-653058d11d9d',

'instances': [{

'suffix': ' issued colorado beetle alert today beetles found',

'prefix': 'put colorado beetle alert ',

'detection': '[put colorado beetle alert]irish agriculture department[issued colorado beetle alert today beetles found]',

'length': 28, 'offset': 34,

'exact': 'irish agriculture department'}],

'relevance': 0.85699999999999998,

'nationality': 'irish',

'organizationtype': 'governmental civilian'}]

topics [{

'category': 'http://d.opencalais.com/cat/Calais/Environment',

'categoryName': 'Environment',

'__reference': 'http://d.opencalais.com/dochash-1/081ea5d6-a8f6- 3d34-9b56-cc8da5b7664e/cat/1',

'classifierName': 'Calais',

'score': 0.98699999999999999}]

Extractions:

1 entities, 1 topics

Topic: Environment

Organization: irish agriculture department (0.86)

[資料來源：本研究整理]

表 3-5 為語意註解前的原始資料，藉由 OpenCalais 產生語意註解資料內容，見表 3-6，取得輸入資料的實體(Entity，例如人名、地點、組織)、事實(某人為現任美國總統)與事件(某時某地發生了某件事情)。以本語意註解範例來說，語意註解過程分析出愛爾蘭農業署(Irish agriculture department)這個實體，並且預測這篇這篇文章的主旨為環境(Environment)相關議題。

將路透社資料集中的每一篇新聞執行語意註解，如同表3-6中所表示的語意註解內容，用以作為該新聞的註解資料，提供額外的資訊，產生分類效果更好的預測模型，使得文件分類正確率獲得提升，第四章中的試驗將驗證語意註解對文件分類正確率的影響效果。

第五節　分類器 (Classifier)

分類器執行特定演算法，將訓練資料處理運算後產生模型，用於預測資料的分類，因為本研究目的為將文字探勘演算法應用於 MapReduce 軟體開發模型上，因此選定的分類器演算法必須能夠以多個 MapReduce 任務進行運算，再整合運算結果得到最終的答案。

因此原有的文字探勘演算法並非皆適用於 MapReduce 軟體開發模型上運作，本研究採用簡單貝氏分類器，不僅適用於 MapReduce 軟體開發模型，實作上亦較為不複雜。

3.5.1　簡單貝氏分類器

單純貝氏分類器，主要的運作原理是透過訓練樣本，學習與記憶分類根據所使用屬性的關係，產生這些訓練樣本的中心概念，再用學習後的中心概念對未歸類的資料進行類別預測，以得到受測試資料物件的目標值；每筆訓練樣本，一般含有分類相關連屬性的值，及分類結果 (又稱為目標值)；一般而言，屬性可能出現兩種以上不同的值，而目標值則多半為兩元的相對狀態，例如 「是/

否」,「好/壞」,「對/錯」,「上/下」,若目標的分類多於兩種，則逐一比較此訓練樣本落在各類別的機率，將其指派給機率最大的類別作為預測結果。

單純貝氏分類器主要是根據貝氏定理，交換事前及事後機率，配合決定分類特性的各屬性彼此間是互相獨立的假設，來預測分類的結果。

$$BayesianTheorem: \quad h_{MAP} = \arg\max_{h \in V} P(h \mid D) = \arg\max_{h \in V} \frac{P(D \mid h)P(h)}{P(D)} = \arg\max_{h \in V} P(D \mid h)P(h)$$

hMAP: 最大可能的假說 (Maximum A Posteriori)

D: 訓練樣本

V: 假說空間 (hypotheses space)

P(D): 訓練樣本的事前機率，對於假說 h 而言，為一常數

P(h): 假說 h 事前機率 (尚未觀察訓練樣本時的機率)

P(h|D): 在訓練樣本 D 集合下，假說 h 出現的條件機率

單純貝氏分類器根據訓練樣本，假設一共有 n 個學習概念的屬性 $A_1, A_2, \ldots, A_n$，a_1為 A_1相對應的屬性值，對於所給予測試物件 X 的屬性值 $x_1, x_2, x_3, \ldots, x_n$ 指派具有最高機率值的類別 (C 表示類別的集合) 為目標結果。

亦即假設共有 m 種類別$(C_1, C_2, \ldots, C_m)$，給定一筆測試物件 $X=(x_1, x_2, x_3, \ldots, x_n)$，單純貝氏分類器將預測 X 屬於具有最高機率值的類別，若 X 預測屬於類別 C_i，則：

$$P(C_i|X) > P(C_j|X) \text{ for } 1 \le j \le m, j \ne i.$$

只要單純貝氏分類器所涉及學習概念的屬性，彼此間互相獨立的條件被滿足時，單純貝氏分類器所得到的最大可能分類結果，與貝氏定理的最大可能假說 h_{MAP} 具有相同的功效。

3.5.2 餘集簡單貝氏分類器

簡單貝氏分類器處理偏斜資料時會產生決策邊界(Decision Boundary)問題，亦即當訓練資料中某些類別的訓練資料較多，簡單貝氏分類器的分類預測模型將會偏向預測這些訓練資料較多的類別，導致分類正確率的下降。

$$l_{MNB}(d) = \arg\max_c \left[\log P(\hat{\theta}_c) + \sum_i f_i \log \frac{N_{ci} + \alpha_i}{N_c + \alpha} \right]$$

$$l_{CNB}(d) = \arg\max_c \left[\log P(\overrightarrow{\theta}_c) - \sum_i f_i \log \frac{N_{\tilde{c}i} + \alpha_i}{N_{\tilde{c}} + \alpha} \right]$$

而餘集簡單貝氏分類器可用於改善簡單貝氏分類器的決策邊界問題，比較簡單貝氏分類器與餘集簡單貝氏分類器在選擇預測類別的分類器模型，權重項的組成改為 $N_{\tilde{c}}$ 與 $N_{\tilde{c}i}$，$N_{\tilde{c}}$ 為不屬於類別 C 的文件中所有字詞的總數，$N_{\tilde{c}i}$ 為 i 這個詞出現在不屬於類別 C 的文件中的次數，權重項的負號意義為欲將待分類文件指派至最不適合餘集參數預測的類別 C 中。

3.5.3 簡單貝氏分類器與餘集簡單貝氏分類器的實作

簡單貝氏分類器與餘集簡單貝氏分類器在實作上皆做了數值的轉換，權重的計算使用 TF-IDF 值避免常用字(Common terms)影響分類器的預測結果，並且透過正規化的方式，排除待分類文件篇幅長度對於分類結果的影響，使得篇幅長的文件在分類器的訓練過程中佔有過多優勢。

第六節　評估器 (Evaluator)

評估器中使用正確率(Accuracy)，用以評估分類器建模產生的預測模型之正確率，將資料集中切分出來的測試資料用來評估分類的正確程度，在本研究中分類正確率的定義為：

$$\text{Accuracy} = \frac{\#\text{ of Correct Prediction s}}{\#\text{ of Examples}}$$

在本研究中，評估器亦運作於 Hadoop 雲端運算系統平台，因此得以適用於大型的測試資料集，評估器將測試資料分配給運算叢集中的每個運算節點，每個運算節點根據分類器產生的模型預測測試資料的分類，最後再整合預測結果用以評估分類模型的正確率。

第七節　平台建置

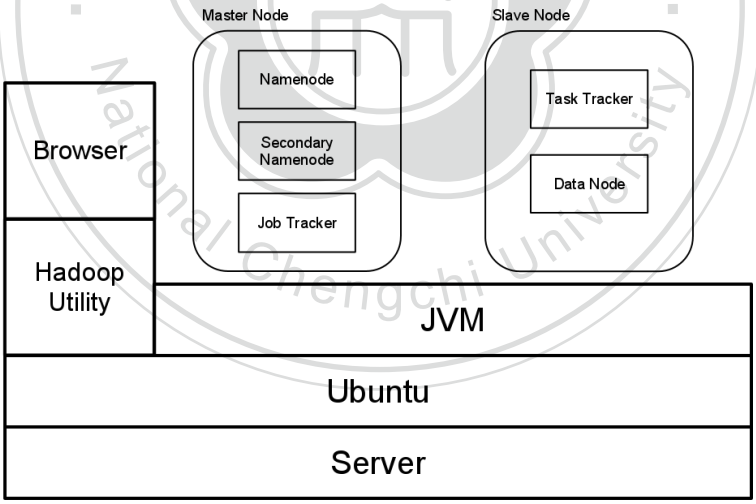


圖3-4 本研究之系統運作圖

[資料來源：本研究整理]

為了使文件探勘演算法得以在雲端運算環境上運作，本研究使用 Hadoop 作為雲端運算軟體平台，執行單一節點的虛擬分散式運算模式(Pseudo-distributed Mode)，在此模式下每個 Hadoop Daemon 執行於各自獨立的 Java 虛

擬機器上(JVM)，用於模擬真實的分散式環境，如圖3-4所示，將文件探勘演算法的工作、設定、Map 函式和 Reduce 函式交給 JobClient 執行，用於在雲端運算環境下進行分散式文字探勘及結果分析，JobClient 遞交工作給 Job Tracker，Job Tracker 將簡單貝氏分類器的工作切分為數個工作單元，分派各工作單元給各個 Task Tracker 執行工作。

文字探勘分為數個部份，分別為資料前置處理、語意註解、分類器與評估器，路透社資料集經過去除停用字、附加語意註解資料及文本詞彙長度統計分類，再進行分類器的訓練，分類器與評估器以模型(Model)連接，模型是由分類器以訓練資料建模而成，過程中產生的資料均存放於 HDFS 中，以供存取，在本研究中為了資料檢索的便利性，將簡單貝氏分類器及餘集簡單貝氏分類器訓練過程中所產生的權重計算檔案匯入關聯式資料庫(MySQL)，方便進一步的分析。

Hadoop 雲端文字探勘平台將 MapReduce 軟體設計模型應用於簡單貝氏分類演算法及餘集簡單貝氏分類演算法的實作上，作為分類器的演算法，在第四節的實驗中將會驗證文件分類演算法於分類正確率上的差異。

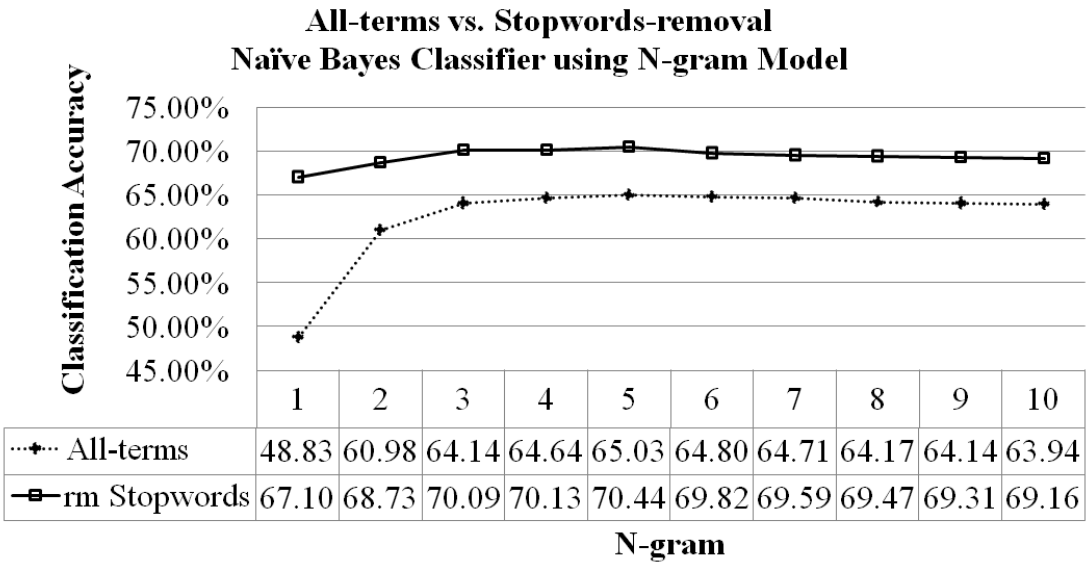
第四章 研究結果

為發展本研究所提出之雲端文字探勘平台，主要分為(1)將 MapReduce 軟體設計模型應用於簡單貝氏分類演算法的實作上，(2)執行路透社資料集(Reuters 21578)的資料前置處理，包含斷詞切字、去除停用字及詞幹還原，(3)以路透社資料集(Reuters 21578)驗證本研究提出的雲端文字探勘平台之分類正確率，(4)藉由實驗比較去除停用字、語意註解、文件分類演算法及文本詞彙長度對於文件分類正確率的影響。

以下各實驗中所使用的文件分類演算法，不論是簡單貝氏分類器演算法，或是餘集簡單貝氏分類器演算法，皆套用 N-gram 模型做為文件分類演算法的輔助，避免特徵與目標詞以及詞義的相關聯性表現不出來。

第一節　去除停用字對文件分類正確率的影響

第一階段實驗主要目的為驗證去除停用字對文件分類正確率的影響。使用原始資料以及去除停用字的文本作為訓練資料及測試資料，在文字探勘的資料前置處理中，去除停用字被視為很重要的一個步驟，因為停用字在文章中出現的頻率相對於其他字詞較高，將會某種程度地干擾文件分類模型，造成文件分類正確率的正確率下降。



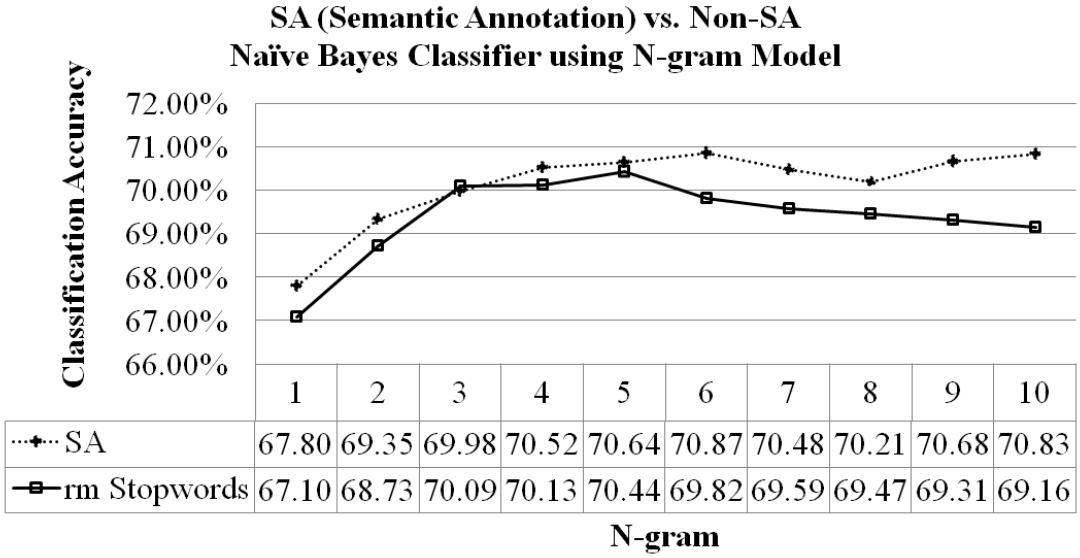
	1	2	3	4	5	6	7	8	9	10
All-terms	48.83	60.98	64.14	64.64	65.03	64.80	64.71	64.17	64.14	63.94
rm Stopwords	67.10	68.73	70.09	70.13	70.44	69.82	69.59	69.47	69.31	69.16

圖4-1 去除停用字對於文件分類正確率的影響

[資料來源：本研究整理]

本實驗所使用的簡單貝氏分類器演算法，其中計算權重的部分為透過 TF-IDF 值計算，比起傳統的簡單貝氏分類器演算法，較可避免停用字或各文件中出現頻率較高的字詞干擾文件分類模型的情況發生，但本實驗證實停用字是否去除依然會對文件分類模型的分類正確率產生影響，如圖4-1所示，因此接下來的實驗皆使用去除停用字的文本作為訓練資料及測試資料。

第二節　語意註解對文件分類正確率的影響



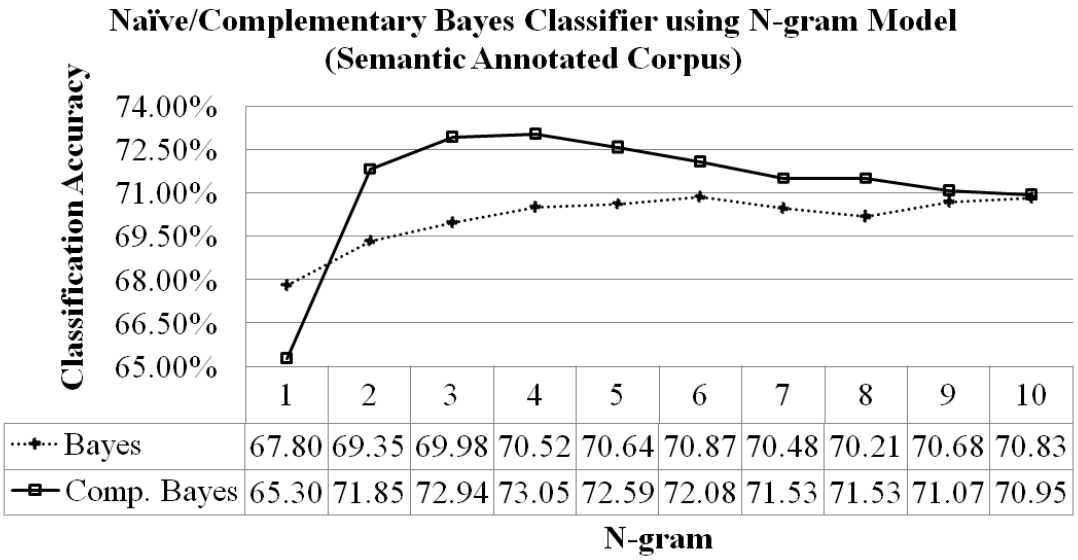
	1	2	3	4	5	6	7	8	9	10
SA	67.80	69.35	69.98	70.52	70.64	70.87	70.48	70.21	70.68	70.83
rm Stopwords	67.10	68.73	70.09	70.13	70.44	69.82	69.59	69.47	69.31	69.16

圖4-2 語意註解對於文件分類正確率的影響

[資料來源：本研究整理]

第二階段實驗為評估應用語意註解前後是否對文件分類結果產生影響，如圖4-2所示，基於同一個文件分類演算法(簡單貝氏分類器)，使用語意註解作為詮釋資料的來源，可提高文件分類的正確率。目前可獲得的助益雖不大顯著，語意註解的技術仍在持續發展中，所能取得的詮釋資料將會隨著時間與日俱增，未來將能提供文件分類更為充足的語意註解內容。

第三節　比較簡單貝氏分類器與餘集簡單貝氏分類器之分類正確率



	1	2	3	4	5	6	7	8	9	10
Bayes	67.80	69.35	69.98	70.52	70.64	70.87	70.48	70.21	70.68	70.83
Comp. Bayes	65.30	71.85	72.94	73.05	72.59	72.08	71.53	71.53	71.07	70.95

圖4-3 簡單貝氏分類器及餘集簡單貝氏分類器之分類正確率的影響

[資料來源：本研究整理]

第三階段為評估套用 N-gram Model 的簡單貝氏分類器及補集簡單貝氏分類器，應用在語意註解後的資料集。其正確率是否會出現差異，如圖4-3所示，補集簡單貝氏分類器的確可以改善簡單貝氏分類器應用在偏斜資料時會偏好訓練資料較多的類別的問題，Reuters 21578資料集如前面所示，正如偏斜資料所定義的，許多訓練資料都集中於少數幾個類別。

N-gram 模型的使用，也使得文件分類正確率獲得可觀的提升，尤其是當 N-gram 數值為 2 及 3 時，分類正確率可獲得較多的提升，其原因為某些詞由兩個或三個詞所組成，因此使用 N-gram Model 可有效將這些辭彙挖掘出來，表現出特徵與目標詞以及詞義的相關聯性，使得文件分類的判斷更為準確，進而提高正確率。但當 N-gram 數值持續增加時，新增的特徵可能以干擾資料的形式出現在文件分類模型中，造成文件分類的正確率下降，從補集簡單貝氏分類器的正確率曲線可得知此點。

第四節　餘集簡單貝氏分類器消除偏斜資料產生的決策邊界問題之效果

簡單貝氏分類器處理偏斜資料時產生的決策邊界問題，亦即當訓練資料中某些類別的訓練資料較多，簡單貝氏分類器的分類預測模型將會偏向預測這些訓練資料較多的類別，導致分類正確率的下降。

從第三節實驗中，比較了簡單貝氏分類器和餘集簡單貝氏分類器，對具備偏斜特性的路透社資料集做文件分類的正確率差異，從圖4-3可以發現餘集簡單貝氏分類器在大多數的情況中，分類正確率皆高於簡單貝氏分類器的分類結果。

以簡單貝氏分類器和餘集簡單貝氏分類器其中的權重計算結果做比較，兩者的權重愈大皆代表該詞彙重要性愈高，分析數值大小為前2500名的權重。

表4-1 簡單貝氏分類器及餘集簡單貝氏分類器之權重值為前2500名中各類別的分佈情形

(類別依照訓練資料由多至少排列)

演算法 / 類別	**簡單貝氏分類器 (Naive Bayes) N=6, Annotated Dataset**	**餘集簡單貝氏分類器 (Complementary Bayes) N=4, Annotated Dataset**
earn	1932	1762
acq	559	709
crude	2	7
trade	0	5
money-fx	4	10
interest	3	5
money-supply	0	2
共計	**2500**	**2500**

[資料來源：本研究整理]

從表4-1可發現簡單貝氏分類器的權重皆為訓練資料較多的類別(earn, acq, crude, money-fx 和 interest 等類別)，其餘類別的權重皆未出現在前2500名的權重中，使得文件分類預測的結果將會偏好這些訓練資料較多的類別，造成誤判、分類正確率的下降。

而餘集簡單貝氏分類器的權重分佈較簡單貝氏分類器分散，權重值高的詞彙沒有集中在特定幾個大類別，也出現在訓練資料較少的類別中(earn, acq, crude, trade, money-fx, interest 和 money-supply 等類別)，因此餘集簡單貝氏分類器的預測結果將較不會偏好訓練資料較多的大類別，減少誤判造成正確率下降的情形發生。

第五節　文本詞彙長度對文件分類正確率的影響

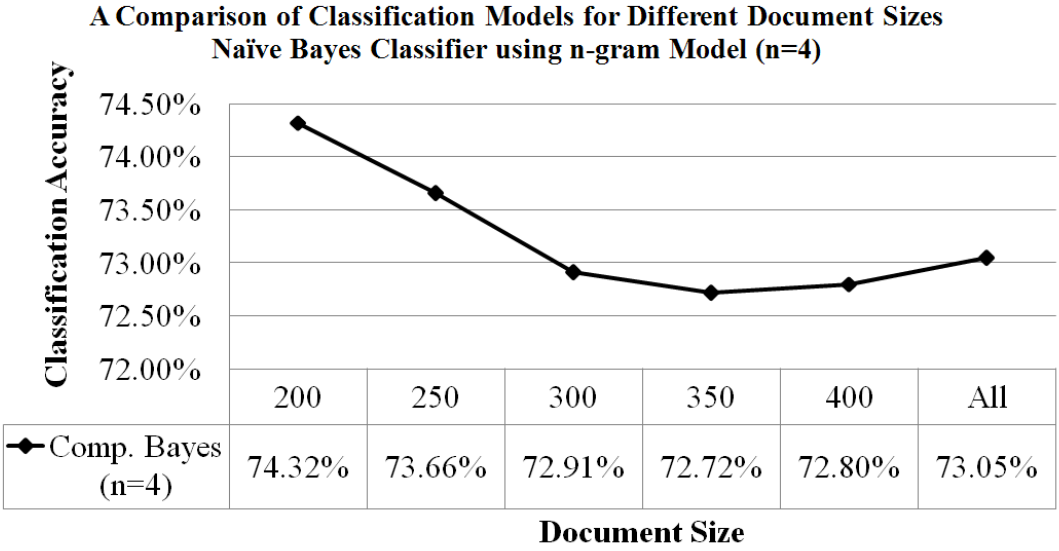


圖4-4文本詞彙長度對文件分類正確率的影響

[資料來源：本研究整理]

本節為評估文本詞彙長度(Document)對於文件分類正確率的影響，使用第三階段試驗的最佳組合作為測試，即採用餘集簡單貝氏分類器、N-gram 模型的 N 值為4，使用套用語意註解後、詞彙長度分別為200字以內、250字以內、300字以內、350字以內和400字以內的文本執行簡單貝氏分類器的訓練及評估。

如圖4-4所示，選用詞彙長度較小的文本所訓練出的餘集簡單貝氏分類器，可獲得較佳的文件分類正確率，其原因在於篇幅較長的文件用於訓練餘集簡單貝氏分類器的分類預測模型，會主導分類模型的產生，對於篇幅較小的文件將產生不利的影響，造成分類正確率的下降。

詞彙長度為350字以內時，正確率下降的比例減緩，文件分類的正確率未如預期繼續下降，以及當詞彙長度為400字時，出現分類正確率不減反增的情形，原因在於當文件篇幅較長時，語意註解從其中辨認出的個體及關係增加許多，提供文件更多的註解資料，使得文件內容的意義更為明確；從圖4-4中，詞彙長度標示為「All」的正確率來看，透過語意註解可以適度緩解文本詞彙長度對於文件分類正確率的負相關關係。

第六節　避免模型過適問題

在資料探勘及文字探勘的相關研究中，所訓練出來的模型必須要避免模型過適問題(Overfitting)，不對訓練資料作過多的解釋，使模型對於訓練資料的局部特性解釋過多，造成模型過度訓練，對於其他的測試資料預測能力因而降低。

常見的作法為使用測試資料集驗證、交互驗證(N-folds Cross Validation)，使用測試資料集驗證即是完整的資料集切分為兩部分，一部份作為訓練資料集，另一部份則為測試資料集，以測試資料集驗證訓練資料集所產生的模型，作為分類的結果；所謂交互驗證則是把資料集隨機分為 N 個子集合(N 通常為4、5或10)，每一次選擇其中一個子集合作為訓練資料，輪流作為訓練資料集及測試資料集，最後將各次驗證結果的正確率取平均值得到交互驗證的正確率，作為該模型的分類結果。

本研究使用測試資料集驗證方式，作為分類結果的效果評估，實驗中採用路透社資料集 Mod Apte 切分版本，使用它的訓練、測試文件集區分法，依據其資料標籤將資料分為訓練資料集與測試資料集，分別有6532筆及2568筆資料。

訓練資料集的錯誤率會隨著 N-gram 值的成長逐漸下降，但若僅考慮訓練資料集錯誤率就會造成模型過適的問題，因此使用了測試資料集(不包含於訓練資料集中)作為評估因子，一旦測試資料集錯誤率上升，即代表應該要停止，因為已經產生模型過適問題。

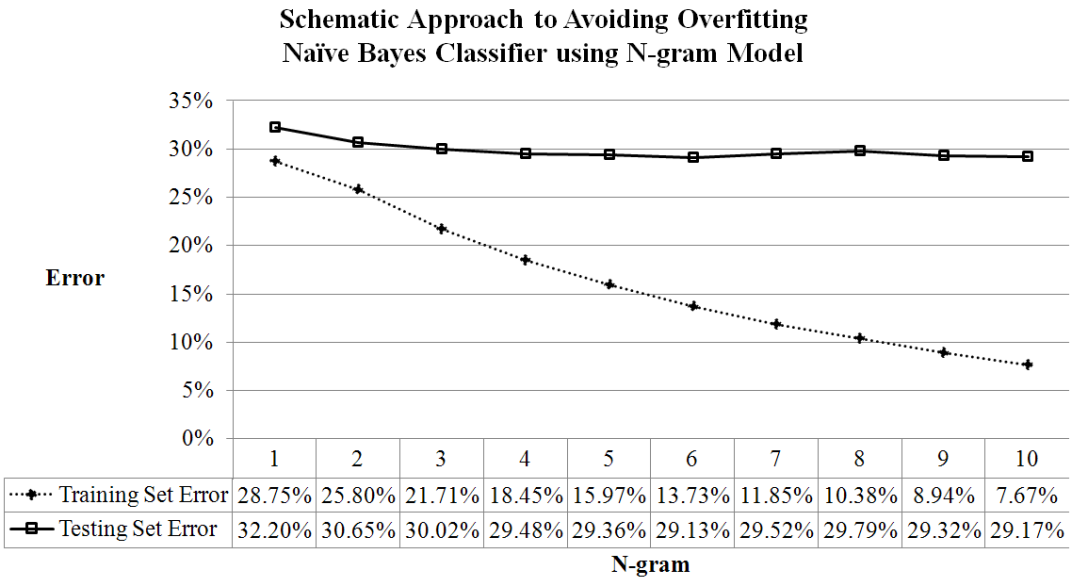


圖4-5 簡單貝氏分類器的訓練集錯誤率及測試集錯誤率

[資料來源：本研究整理]

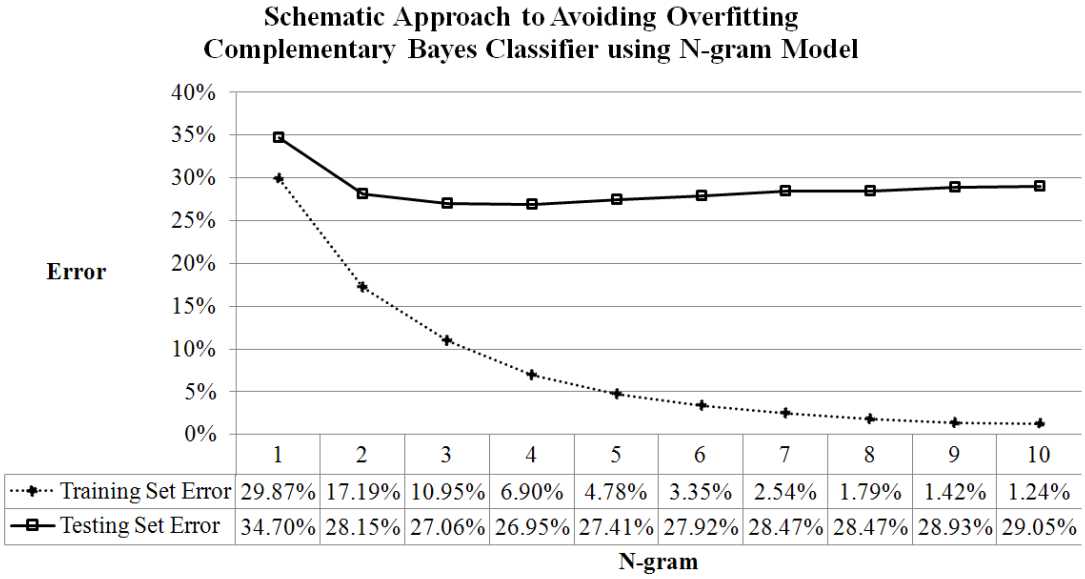


圖4-6 餘集簡單貝氏分類器的訓練集錯誤率及測試集錯誤率

[資料來源：本研究整理]

如圖4-5及圖4-6所示，不論是簡單貝氏分類器或餘集簡單貝氏分類器的訓練，訓練資料集的錯誤率皆逐次下降，但測試資料集的錯誤率在某些參數組合下持續下降後隨即上升，圖4-5及圖4-6中可見簡單貝氏分類器及餘集簡單貝氏分類器分別在 N-gram 為6及4時，其測試集錯誤率最低，透過這樣的方式可以避免模型過適問題的產生，因而知道停止點應設定在哪一點上。

第七節　類別分佈均勻程度對於分類結果之影響

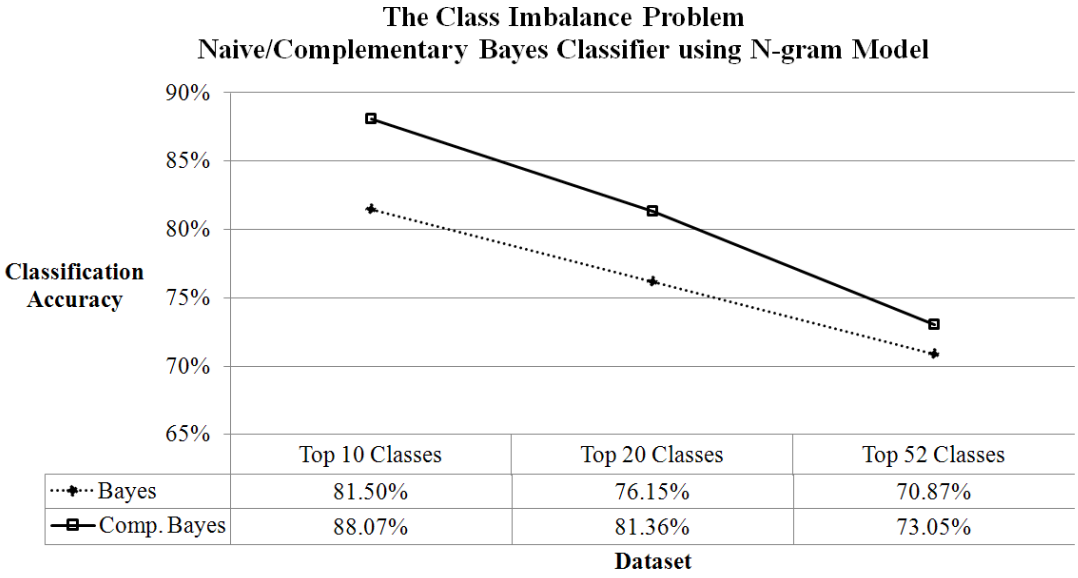


圖4-7 資料集中類別分佈對分類結果的影響

[資料來源：本研究整理]

由於路透社資料集具備資料偏斜特性，各類別的訓練資料數量差異相當大，將會使得分類器偏向預測訓練資料較多的類別，造成分類正確率的下降。本節探討類別分佈的偏斜程度對於分類正確率的影響，分別選用表3-3中資料總數為前10名及前20名的類別組成訓練資料集和測試資料集，採用簡單貝氏分類演算法及餘集簡單貝氏分類演算法，依照第六節中避免模型過適的方法，挑選出最佳的組合及分類正確率做為文件分類結果，並與原先52個類別資料集之分類結果做比較。

圖4-7為其實驗結果，加入資料總數較少的類別，會使得分類結果變差，此情形同時出現在簡單貝氏分類演算法及餘集簡單貝氏分類演算法的實證結果中，路透社資料集中前兩大的類別(earn, acq)即分別佔了43.11%、25.19%，使得資料集的歪斜特性變得相當明顯，在訓練資料集及測試資料集中加入小類別將會使誤判增加，造成分類正確率的下降，不論簡單貝氏分類演算法或者是餘集簡單貝氏演算法皆會受到大類別的影響而偏向預測大類別，但透過實驗證實餘集簡單貝氏分類器確實可以在分類具備偏斜特性之資料集的情況下，得到比簡單貝氏分類演算法較佳的結果。

第八節　各階段實驗結論整理

前述各階段的實驗結果，各自取出最佳的參數組合整理於表4-2中：

表4-2 各階段實驗最佳的參數組合及分類正確率

資料集處理方式 (Data Preprocessing)	文件分類演算法 (Algorithm)	N-gram	文本詞彙長度 (Document Length)	正確區別率 (Training Set Accuracy)	分類正確率 (Accuracy)
原始檔案	簡單貝氏分類器	5	全部	72.92%	65.03%
去除停用字	簡單貝氏分類器	5	全部	81.06%	70.44%
去除停用字、語意註解	簡單貝氏分類器	6	全部	86.27%	70.87%
去除停用字、語意註解	餘集簡單貝氏分類器	4	全部	93.10%	73.05%
去除停用字、語意註解	餘集簡單貝氏分類器	4	200字以內	94.22%	74.32%

[資料來源：本研究整理]

將各階段實驗的資料集處理方式、使用的文件分類演算法及相關參數(N-gram 的 N 值與資料集是否僅包含特定詞彙長度的文章)，而正確區別率指的是當模型的訓練出現模型過適問題時，訓練資料集用於模型產生的分類正確率，也就是簡單貝氏分類器或餘集簡單貝氏分類器應該要停止訓練的點(Cutting

Point)，如表4-2中所示，可以發現分類正確率逐次上升。

對本研究所使用的路透社資料集(Reuters 21578)而言，去除停用字，可避免頻率高且不具意義的停用字主導文件分類模型的預測，使分類正確率獲得巨幅的成長(65.03% → 70.44%)。

而加入語意註解後，為資料集提供許多額外的資訊，在文件分類的過程中提供一定程度的助益，在實驗結果中也可證實此點(70.44% → 70.87%)。

採用餘集簡單貝氏分類器，用以避免具備歪斜特性的資料集對於分類正確率的負面影響，實驗結果驗證了使用餘集簡單貝氏分類器，可以減少誤判的情況，使得文件分類的預測較不會偏向訓練資料較多的類別，因此提高了文件分類正確率(70.87% →73.05%)。

最後是探討資料集中詞彙長度的影響，以餘集簡單貝氏分類器做文件分類，使用相同的 N 值，從去除停用字及語意註解的資料集中篩選出文本詞彙長度為200字以內的文件，如此一來減少了長篇幅文件對文件分類模型預測結果的負面效果(73.05% →74.32%)。

本研究以路透社資料集驗證簡單貝氏分類器及餘集簡單貝氏分類器於雲端運算環境執行的分類正確率。實驗結果顯示，分類正確率的改善結果為74.32%，比較 Kabriya et al.(2004)的研究結果(69.46%)，對於分類正確率的改善效果更佳，去除詞彙長度的篩選對分類正確率的影響，本研究的實驗結果尚有73.05%的分類正確率，經過本研究提出的文件分類流程及其中影響因子的調整，可以得到更好的文件分類效果。

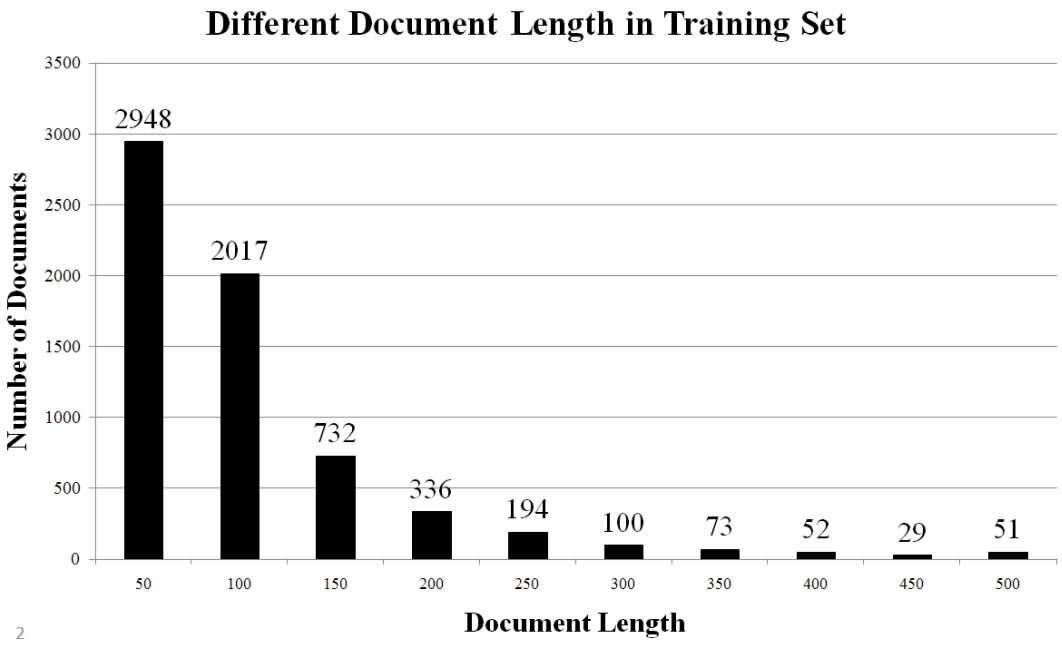


圖4-8 路透社資料集中各詞彙長度的文件之分佈情形

[資料來源：本研究整理]

由圖4-8資料集中各詞彙長度文件的分佈情形及第五節驗證文本詞彙長度對文件分類正確率的影響，雖詞彙長度為200字以上的文件僅佔499篇，佔全部訓練資料的7.6%，但卻顯著地造成文件分類正確率的下降，由此可見篇幅較長的文章可有效主導文件分類模型的預測結果，造成數量可觀的誤判，進而使得分類正確率減少。

第五章 結論與建議

第一節　結論

本研究採用 Hadoop 建構雲端系統平台執行文件分類，使得大型資料集得以更迅速且取得分析結果，不必受限於單一機器的規格。更為完善利用運算叢集的強大運算能力，經確認雲端運算環境下執行文字探勘演算法的正確率和單機執行文字探勘演算法的正確率沒有顯著差異後，便可將大型資料集的文字探勘應用於雲端運算的環境上，對於大型企業及學術單位的研究有偌大的幫助。

本研究經由實驗比較去除停用字、語意註解、文件分類演算法及文本詞彙長度對於文件分類正確率的影響，得到以下實證結果：

1. 資料集去除停用字：65.03%→ 70.44%
2. 加入語意註解做為詮釋資料：70.44% → 70.87%
3. 使用餘集簡單貝氏分類器解決決策邊界問題：70.87% →73.05%
4. 避免篇幅較長的文件主導分類預測結果：73.05% →74.32%

從去除停用字實驗中得知出現頻率高的停用字對於分類預測的負面影響；語意註解作為詮釋資料的取得方式，可增加文件分類的效果，但在本研究之實驗可發現加入語意註解僅輕微改善文件分類的結果，其原因在於語意註解技術目前仍不斷在發展當中，本研究所使用的 Open Calais 語意註解服務所能提供的詮釋資料，未來會因為 Open Calais 語意註解服務越來越成熟而得到更多的詮釋資料作為額外的資訊，可望更進一步改善文件分類結果；而餘集簡單貝氏分類器，可用以減少偏斜資料對於分類預測結果的誤判；文本詞彙長度較長的文章則會某種程度主導分類預測結果，造成誤判的產生，降低分類正確率，各階段的實驗使得文件分類的結果獲得改善，並得以掌握文件分類正確率的影響因子。

比較 Kabriya et al.(2004)的研究結果，以路透社資料集驗證餘集簡單貝氏分類器的分類正確率為69.46%，本研究之文件分類結果改善不少(74.32%)，儘管

去除掉詞彙長度對分類正確率的影響，本研究之分類結果尚有73.05%的分類正確率，透過實證結果證實本研究的文件分類流程及其中影響因子的調整，能夠改善文件分類結果。

語意註解的使用，使得資料得以機器可閱讀的格式流通在網路上，未來可朝向本體論整合和本體論學習之研究，而語意註解結果亦可用作語意搜尋引擎的使用，擴展自然語言查詢的應用範圍。而將語意註解應用於雲端運算環境上，使得語意註解演算法中的大量計算都得以分散至運算叢集中的機器運算處理，幫助語意註解及語意網技術的推展，作為語意搜尋引擎解析索引之用。

第二節　未來研究方向

本研究驗證了去除停用字、語意註解、文件分類演算法及文本詞彙長度對於分類正確率的影響，藉由本論文的實證結果與經驗的累積，提出以下建議：

1. 本研究採用具備公證性的路透社資料集，作為文件分類正確率的比較評估標準，透過各階段的實驗，使其分類正確率獲得改善。因此未來應使用 Twitter 或 Facebook 網站中大量非結構化的資料，執行進一步分析，如此一來亦可提高本研究雲端文字探勘平台的實用性，更可以提供更多更具意義的分析結果。
2. 因為語意註解平台 OpenCalais 目前尚未支援中文，使得本研究之實驗僅使用英文文本作為訓練資料及測試資料。未來應納入處理中文的能力，使得本研究之雲端文字探勘平台得以應用在更多中文資料上，提供分析在地資料的功能。
3. 因為簡單貝氏分類器具有高效率和低複雜度的特性，使得本研究在分類器的實作上採用了簡單貝氏分類器及餘集簡單貝氏分類器，未來可繼續實作分類效果更好的文件分類演算法，例如支持向量機(Support Vector Machine, SVM)或類神經網路(Neural Network)，提升文件分類的效果，使得正確率進一步提高。

參考文獻

[1] Apte, C., Damerau, F., & Weiss, S. M. (1994). *Towards language independent automated learning of text categorization models.* Paper presented at the ACM SIGIR Conference on Research and Development in Information Retrieval.

[2] AWS in Education. (2010). from http://aws.amazon.com/education

[3] Berendt, B., Hotho, A., & Stumme, G. (2002). Towards semantic web mining. *The Semantic Web—ISWC 2002*, 264-278.

[4] Center, N. R. (2008). DisCo. from http://discoproject.org/

[5] Chang, F., Dean, J., Ghemawat, S., Hsieh, W. C., Wallach, D. A., Burrows, M., et al. (2008). Bigtable: A distributed storage system for structured data. *ACM Transactions on Computer Systems (TOCS), 26*(2), 4.

[6] Chu, C. T., Kim, S. K., Lin, Y. A., Yu, Y. Y., Bradski, G., Ng, A. Y., et al. (2007). *Map-reduce for machine learning on multicore.* Paper presented at the NIPS.

[7] Dean, J., & Ghemawat, S. (2008). MapReduce: Simplified data processing on large clusters. *Communications of the ACM, 51*(1), 107-113.

[8] Dlugolinsky, S., Laclavik, M., Seleng, M. (2010). Ontea Semantic Annotation. from http://ontea.sourceforge.net/

[9] Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). The KDD process for extracting useful knowledge from volumes of data. *Communications of the ACM, 39*(11), 27-34.

[10] Foundation, A. S. (2008). Cassandra. from http://incubator.apache.org/cassandra/

[11] Foundation, A. S. (2008). Hadoop. from http://hadoop.apache.org/core/

[12] Foundation, A. S. (2010). Hbase. from http://hadoop.apache.org/hbase/

[13] Gillick, D., Faria, A., & DeNero, J. (2006). Mapreduce: Distributed computing for machine learning.

[14] Google. (2010). Google App Engine. from http://code.google.com/intl/en/appengine

[15] Hypertable. (2010). Hypertable. from http://www.hypertable.org

[16] Jenkin, N. (2009). COMP390-09A Report Distributed Machine Learning with Hadoop.

[17] Kibriya, A., Frank, E., Pfahringer, B., & Holmes, G. (2005). Multinomial naive bayes for text categorization revisited. *AI 2004: Advances in Artificial Intelligence*, 235-252.

[18] Laclavik, M., eleng, M., & Hluchy, L. (2008). Towards large scale semantic annotation built on mapreduce architecture. *Computational Science–ICCS 2008*, 331-338.

[19] Laclavik, M., Seleng, M., Gatial, E., Balogh, Z., & Hluchy, L. (2007). Ontology based Text Annotation–OnTeA. *Information modelling and knowledge bases XVIII*, 311.

[20] Lioma, C., Moens, M. F., & Azzopardi, L. (2008). Collaborative annotation for pseudo relevance feedback. *ESAIR, 11*, 25-35.

[21] Maedche, A. (2001). *Ontology learning for the semantic web*: Intelligent Systems, IEEE.

[22] Mell, P., & Grance, T. (2009). The nist definition of cloud computing. *National Institute of Standards and Technology*.

[23] Ontotext. (2009). KIM Semantic Annotation. from http://www.ontotext.com/kim/introduction.html

[24] Papadimitriou, S., & Sun, J. (2008). *Disco: Distributed co-clustering with Map-Reduce: A case study towards petabyte-scale end-to-end mining.* Paper presented at the ICDM.

[25] ReadWriteWeb. (2010). Does Facebook Really Want a Semantic Web? , from http://www.readwriteweb.com/archives/does_facebook_really_want_a_semantic_web.php

[26] Reeve, L., & Han, H. (2005). *Survey of semantic annotation platforms.* Paper presented at the Proceedings of the 2005 ACM symposium on Applied computing.

[27] RIGHTSCALE. (2010). RIGHTSCALE. from http://www.rightscale.com/index.php

[28] Sivashanmugam, K., Sheth, A., Miller, J., Verma, K., Aggarwal, R., & Rajasekaran, P. (2003). Metadata and semantics for Web services and processes. *Datenbanken und Informationssysteme: Festschrift zum, 60*, 245-271.

[29] Stanford. (2005). TAP. from http://ksl.stanford.edu/projects/TAP/

[30] Stanford. (2007). Phoenix. from http://csl.stanford.edu/~christos/sw/phoenix/

[31] Wegener, D., Mock, M., Adranale, D., & Wrobel, S. (2009). *Toolkit-based high-performance data mining of large data on MapReduce clusters*.

[32] Wikipedia. (2007). Dbpedia. from http://wiki.dbpedia.org/

[33] Wikipedia. (2010). Cloud Computing. from http://en.wikipedia.org/wiki/Cloud_computing

[34] Witten, I. H., & Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques*: Morgan Kaufmann Pub.

[35] gipi 的學習筆記，2009，http://www.dotblogs.com.tw/jimmyyu/。

[36] 劉繼鴻，2009，影音 Web 2.0平台網站上行銷傳播之社會網絡與資料探勘分析研究－以 YouTube-Mac 網絡為例，國立政治大學資訊管理研究所碩士論文。

[37] 劉俊宏，2009，雲端運算環境下學習社群服務導向架構平台之研究，國立政治大學資訊管理研究所碩士論文。

[38] 王耀聰、陳威宇，2008，雲端運算簡介，http://bit.ly/bXsTVT。

[39] 謝良奇，2008，HP、Intel、Yahoo 共組開放源碼雲端運算計畫，http://bit.ly/adLC7D。

[40] 葉慶隆，2009，Semantic Web and Knowledge Management，http://www.deg.byu.edu/ding/research/SemanticAnnotation.html。

[41] 陳瀅，2010，雲端策略：雲端運算與虛擬化技術，天下雜誌。