國立交通大學

資訊管理研究所

碩士論文

以Google 搜尋引擎為基礎之中文剽竊偵測系統

Development of Chinese Plagiarism Detection System Based on Google

研究生：張雅雯

指導教授：柯皓仁、林妙聰

中華民國 九十九 年 六 月

以Google搜尋引擎為基礎之中文剽竊偵測系統

Development of Chinese Plagiarism Detection System Based on Google

研究生：張雅雯　　Student:Ya-Wen Chang
指導教授：柯皓仁、林妙聰　博士　　Advisor: Dr. Hao-Ren Ke、B.M.T. Lin

國 立 交 通 大 學

資 訊 管 理 研 究 所

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 of Science
In
Information Management

June 2010

Hsinchu, Taiwan, the Republic of China

中華民國　九十九 年 六 月

以Google搜尋引擎為基礎之中文剽竊偵測系統

指導教授：柯 皓 仁、林 妙 聰 博士　　研究生：張 雅 雯

國立交通大學資訊管理研究所

摘要

隨著資訊科技與網路的蓬勃發展，搜尋引擎強大的搜尋功能，讓資訊分享變得十分容易，但在使用者缺乏尊重他人智慧財產權觀念的情況下，網路資訊被隨意濫用的情形時有所聞。目前發展出許多不同的剽竊偵測方法，各有其優、缺點，但都針對較具有規則性的英文，而非較無規則性的中文，本研究以Google搜尋引擎為基礎建立中文剽竊系統，利用修正後的最長共同子序列(Longest Common Sequence, LCS)之概念計算搜尋引擎傳回結果與中文文件之間的相似度。實驗證明比未經修正的LCS公式，可大幅降低其假警報(False Positive)機率。期望藉由系統的實際運作，賦有教育意義地教導學生尊重他人智慧財產權。

關鍵字：Google搜尋引擎、剽竊、最長共同子序列

Development of Chinese Plagiarism Detection System Based on Google

Advisor: Dr. Hao-Ren Ke、B.M.T. Lin　　Student: Ya-Wen Chang

Institute of Information Management

National Chiao Tung University

Abstract

With the advancement of information and network technology, powerful search engines facilitate information sharing. However, users who lack the concept of intellectual property rights usually abuse the information on the Internet. As so far, there are many plagiarism detection techniques, most of which focus on regular grammatical patterns in English. Few plagiarism-detection methods were developed for non-regular grammatical patterns like Chinese. This thesis builds a plagiarism detection system for Chinese documents. The proposed system is based on the search results of Google. Considering the concept of the revised longest common sequence (LCS), our system calculates the similarities between the results returned by Google and Chinese documents to be examined. The empirical studies show that the revised longest common sequence can significantly reduce the occurrences of false positives. We expect that the development of this system can teach students to respect intellectual property rights of others.

Keyword：Google search engines、Plagiarism、Longest Common Subsequence (LCS)

誌　　謝

經過兩年來的風風雨雨，什麼稀奇古怪的事情都發生過，但隨著畢業口試的結束，也即將離開這生活五年的環境，一路走來，心中充滿對許多人的感謝。

首先要感謝的當然是指導教授柯皓仁老師一路的辛勤指導，偶爾還要像朋友般聽聽我的小小抱怨，亦師亦友的教導，讓我不斷的成長茁壯。還有林妙聰老師的支持，讓我們一切都很順利。未來我將帶著老師心中的期待與鼓勵，走向人生的另一個階段，心中充滿無限的喜悅。感謝口試委員謝吉隆老師，在口試中給予的建議，使得本研究能夠更加完整。

同時更要感謝資訊檢索研究室這個大家庭，溫暖又溫馨，不管是老師還是學長姐－怡祥、建穎、姿婷，還是同伴筑婷、所辦的淑惠、APC研究室的各位，都是陪伴我一起玩樂、成長的好夥伴，與大家相處的每一個過程，都將成為我人生中最美好的回憶。在這裡要在特別的感謝粘怡祥學長對我的照顧以及祝福實驗室的同伴池筑婷走入幸福的殿堂。

最後則是感謝家人、好友宜庭、育賢、雅紋、家真、培學與男友竑廷的包容與愛護，讓我能順順利利的一路往目標前進，有你們的支持讓我更有勇氣！

張雅雯　謹誌

2010年6月

目錄

表目錄

圖目錄

第一章　緒論

1.1 研究背景與動機

隨著各式各樣的電腦、智慧型手機銷售量的成長，網際網路與人類的生活越來越密不可分。據資策會FIND [25]調查指出，至2009/12/31日止，台灣經常上網人口為1,067萬人，約比前一年成長2%，其上網普及率為46%，可知應用網際網路，已經成為日常生活的一部分。隨著Google搜尋引擎的崛起，其強大的搜尋功能，讓資訊分享變得十分容易，加上「數位化」觀念的普及，大眾已經改變其閱讀與搜尋資訊的習慣，許多創作已透過網路來發佈。但在如此迅速的變化下，其相關的配套措施卻沒有隨之改變，導致網路資訊被任意的使用，即使未獲得著作權人授權，網路使用者仍可以輕易地使用複製、貼上，並將他們取得的資訊內容變成自己的作品，若未標示出資料的出處和取得擁有者之同意，那麼此舉可能形成剽竊(Plagiarism)而侵犯他人之智慧財產權。

近幾年不斷發現學生於學校課業，甚至研究人員在學術研究上之剽竊行為，根據 CNN 於 2002 年 4 月[5]的一則報導指出，調查紐澤西州立羅格斯大學管理教育中心(Rutgers' Management Education Center)的 4500 名中學生中，有 75%的學生，曾有嚴重的欺騙行為，一半以上的剽竊來源自網路，顯示剽竊行為在校園內的氾濫，學生嚴重缺乏尊重他人智慧財產權觀念。在上述的情況越來越嚴重的情況下，保護智慧財產權的觀念受到重視，除了在硬體方面防止資訊的快速散佈以及透過數位版權管理（Digital Rights Management, DRM）等被動的方式外，更應主動積極地偵測是否有剽竊行為的產生。

剽竊與否的判斷十分的主觀，各方專家所下的定義不盡相同，但都傳達一個相同的概念—剽竊是未經原始著作者同意而使用或模仿其思想或語言 [24]。目前發展出許多不同的剽竊偵測方法，各有其優、缺點，但都針對較具有規則性的

英文，針對較無規則性的中文之研究則稀少許多，因此中文文件的剽竊偵測研究有其重要性。

1.2 研究目的

本研究之目的為透過Google搜尋引擎建置一中文剽竊偵測系統，自動化偵測使用者上傳之文件是否有抄襲自網路，並以視覺化介面呈現給使用者，利於人工檢視其文件的剽竊狀況。本研究發展中文文件剽竊偵測系統的目的，不僅僅在於偵測是否有剽竊行為的發生，更期望有教育意義地進而提醒使用者，預防剽竊行為的發生。

由於中文字的意義多變，且字詞組合多變，剽竊與否的判斷，最後還是要依人工的方式去判斷，本研究根據ROUGE之N-gram co-occurrence statistic [10]之其中一種方法--最長共同子序列(Longest Common Subsequence, LCS)之演算法，計算使用者上傳之文件與Google搜尋傳回之搜尋結果之相似度，若高於某一門檻值，代表有剽竊之可能性，則在系統使用介面上顯示出來，並透過ROUGE-W之概念修正LCS演算法與ROUGE-L，降低系統整體假警報率。整體的研究目標如下：

1. 透過Google搜尋引擎，讓使用者知曉其上傳之文件是否有剽竊自網路。
2. 讓使用者檢視其偵測結果，做最後剽竊與否之判斷，並透過警語告知使用者其參考自網路之文句多寡，期望有教育意義地給予大眾避免剽竊的方針。

1.3 論文架構

本論文在第二章介紹剽竊定義、目前最常使用的剽竊偵測方法與本研究相關之文獻。第三章則詳細描述本研究之剽竊系統架構，如何進行文章前置處理作

業，進而與Google搜尋引擎回傳的結果做比對之觀念與方法，最後將偵測結果呈現給使用者。第四章藉由雛形系統的實作，針對網路上新聞、網誌、維基百科之文章與全國高級中等學校小論文寫作比賽之小論文 [21]做系統正確性之驗證，並就產生的結果進行比較與分析。第五章總結本研究並提出未來可改善的研究方向。論文整體架構如圖 1所示。

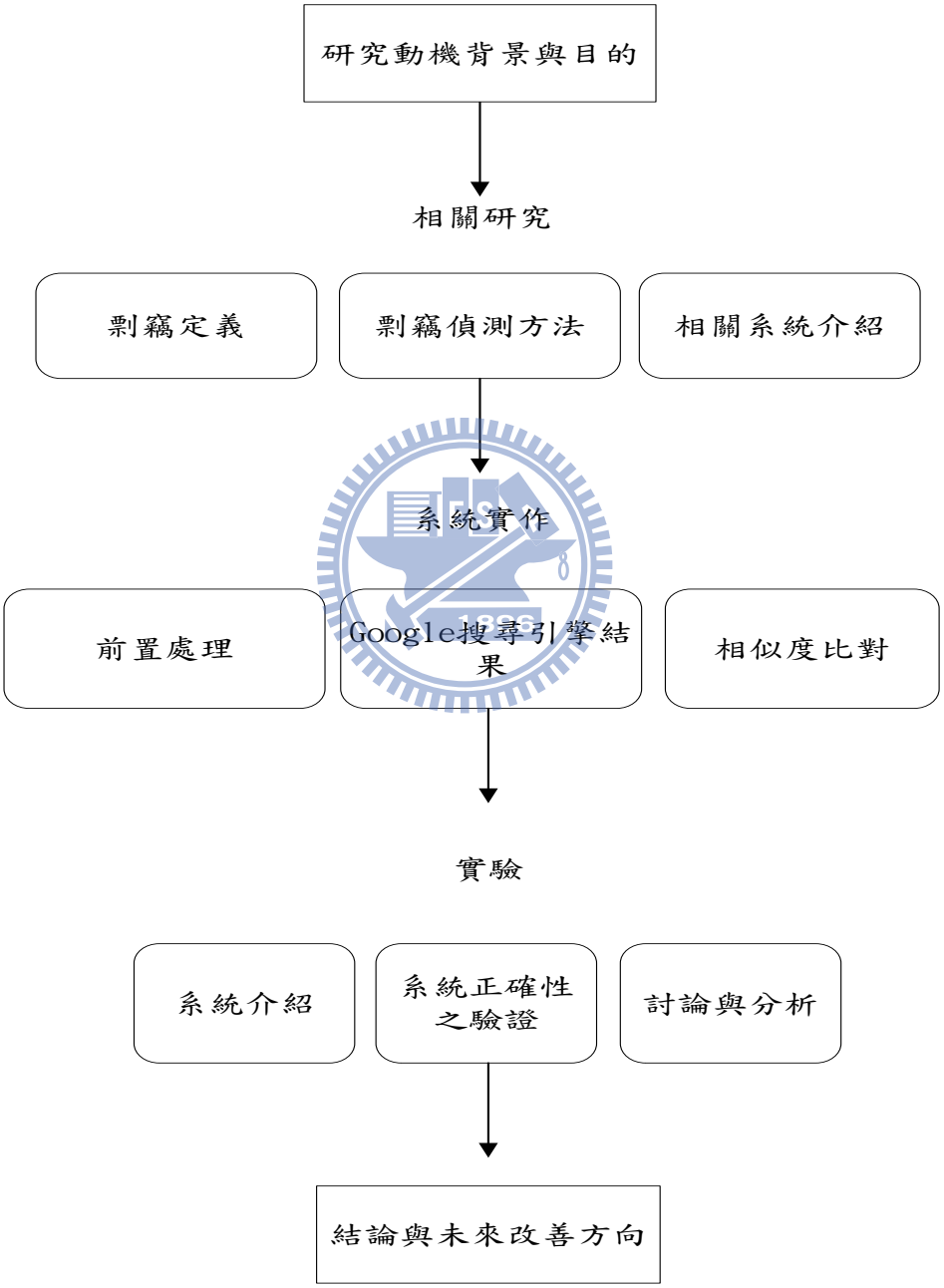

圖 1：論文整體架構

第二章　相關研究工作

本章說明相關的研究工作。分別在2.1節說明剽竊之定義，2.2節闡述常用之剽竊偵測方法，2.3節介紹與本研究相關之文獻。

2.1 剽竊定義

不同的專家或辭典對於剽竊有不同的定義，1995年Random House Compact Unabridged Dictionary [19]定義剽竊為「使用或模仿他人之思想或語言並將其展示成自己的著作」；Merriam-Webster Online Dictionary [14] 定義剽竊為「竊取他人之想法或語言並將其假冒為自己的、不標註引用之來源、將已經存在的想法或產品展示成創新之想法或產品」。概括上述，傳達了一個相同的概念—剽竊乃是未經原始著作者同意而使用或模仿其思想或語言。

2.1.1　剽竊的型式

剽竊的行為有很多種型式，本研究所實作之系統主要專注在偵測中文文件第一種以及第二種剽竊的方式，以下介紹幾種常見的剽竊的方式 [23]：

1. 全部內容相似：兩文件之字詞、文句結構幾乎一模一樣，可能僅有字詞、句子或段落位置有所差異，這種剽竊方式最易被偵測出來。
2. 部份內容相似：兩文件間之內容部份相似，可能僅抄襲一個句子或段落。
3. 替換同義字或改寫：此種剽竊通常會將原文之文字經由添加、刪除或替換，或改變句子結構、詞序；在中文文件上，此種剽竊較難以判斷，傾向主觀的認定。
4. 自我剽竊(Self-Plagiarism)、重覆發表(Dual Publishing) [12]：意指某一作家或研究家將自己同一概念的文章，隔一段時間後，稍做修改後當成新文章發表，並未添加新的想法或是將同一篇文章發表至不

同的期刊。定義上述的剽竊情況，是相當困難的，因為到底多相似才認定為有剽竊嫌疑的界線並不清晰，加上這種狀況有些人並不認為是不道德的行為，更加深定義的困難度。

5. 幽靈作者(Ghost Authorship)：又稱honorary authorship，McCuen [12]認為若某人的名字被掛名於某篇文章上時，但實際上並未對此篇文章有任何有意義的貢獻時，由於可以從中得到好評或名譽，可視為一種未標註引用來源的剽竊行為。

2.2 剽竊偵測方法

迄今，有許多關於剽竊偵測的研究。圖 2 概述了剽竊偵測方法的發展[17]。剽竊偵測方法可大略分成三類，即：指紋識別(fingerprinting)、詞頻統計(term occurrence)、以及風格分析(style analysis)。

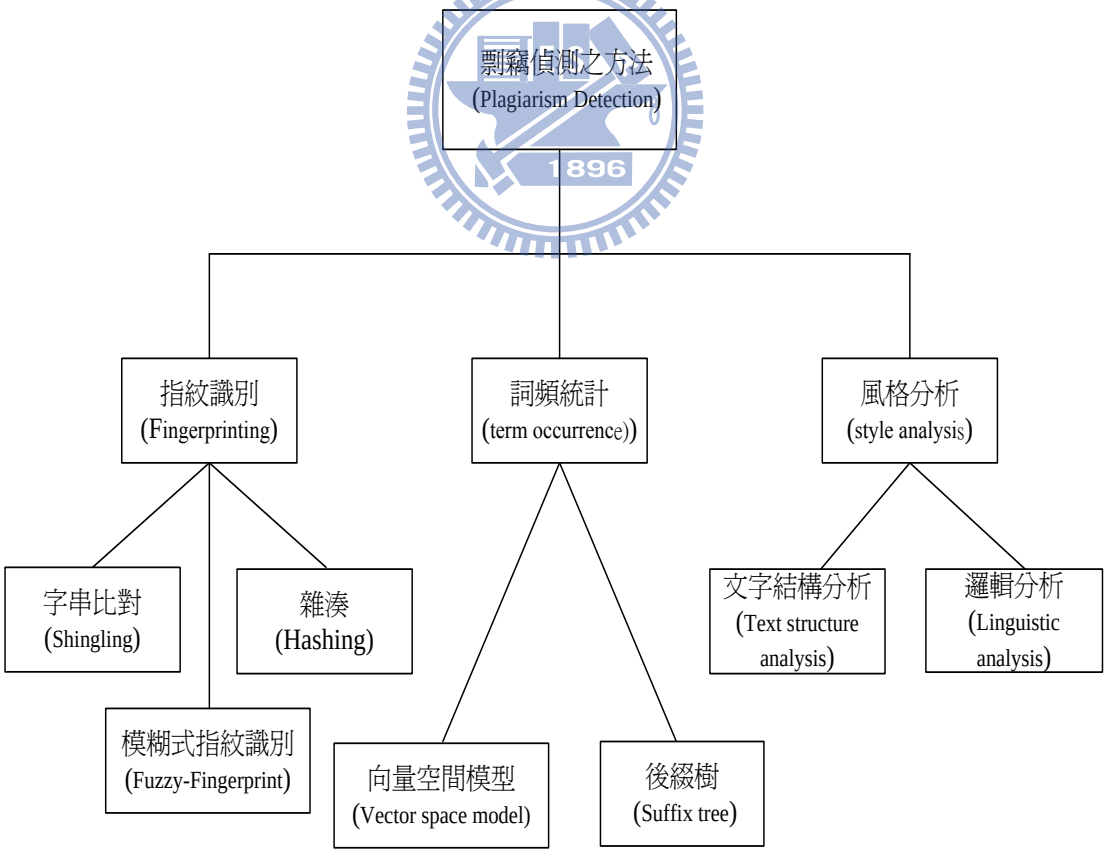

圖 2：剽竊偵測方法之分類與導出方式[17]

2.2.1 指紋識別

指紋識別(fingerprinting)源自Manber [11]，乃廣泛被使用的剽竊偵測技術，針對大範圍的逐字剽竊有良好的剽竊成效。Manber以Rabin Fingerprint scheme [15]為基礎，其每一個文件皆產生各自的獨特識別值(fingerprint)，目的為在資料庫內比對Fingerprint值找出相似之文件。Rabin fingerprint scheme實際上即是透過雜湊函數(hash function)將一串連續的字串轉換為一整數值，而一個良好的基模或函數對於相同子字串應產生相同的整數；另一方面，對於相異之子字串應產生相異之整數，以確保其一致性和避免不必要之Fingerprint碰撞(collisions)。

第一個使用指紋識別的剽竊偵測系統為COPS [3]，以句子(sentence)為最小的單位來做分割並將其儲存在雜湊資料庫中，再經由比對雜湊表檢查兩份文件的相似程度。為了修正指紋識別的方法在某些部份相似的情況成效不佳，延伸出兩種變化：字串比對 [2]和模糊式指紋識別。前者的概念描述如下：

定義文件D是由一連串連續的子字串所組成，其中每一個字(word)稱為Shingle。W為分解文件D中Shingle之單位，且使其為唯一的Shingle。表 1 為字串比對之例子。

表 1 字串比對例子 [2]

文件 D: (a, rose, is, a, rose, is, a, rose)
S(D,4): {(a rose is a)，(rose is a rose)，(is a rose is)}

在給定W的大小後，可由公式(1)、(2)算出文件A、B之間的相似度(resemblance)(方程式(1))與包含度(containment)(方程式(2))。

$$r(A,B)=\frac{|S(A)\cap S(B)|}{|S(A)\cup S(B)|} \quad (1)$$

$$c(A,B)=\frac{|S(A)\cap S(B)|}{|S(A)|} \quad (2)$$

模糊式指紋識別 [4]方法採用IDF(inverse document frequency)，來確定哪些字詞在文章中是有意義的，足以代表該文章。基本上，字串比對與模糊式指紋識別背後的概念是相同的。

2.2.2 詞頻統計

詞頻統計(term occurrence)的概念為最直觀之作法，因為詞彙中包含著關於文字的詳細訊息，可資利用於分析兩文件間之相似度。其中一個假設是當兩文件間擁有愈多之共同詞組時，它們愈相似。因此，詞頻統計可應用於一系列的研究，如自動摘要(text summarization)、資訊檢索(information retrieval, IR)，文件叢集(clustering)和分類(categorization)。

SCAM(Stanford Copy Analysis Mechanism) [16]為應用詞頻統計之代表性系統，透過反轉索引結構(inverted index)來儲存文件的區塊(chunks)，文件區塊可以是段落、句子或字詞，區塊大小的選擇對搜尋成本與儲存成本相當重要。一般來說使用句子一定比使用段落容易找到重疊(overlap)的部份，換句話說，就是chunks區塊越小越能找到文字的重疊，但其搜尋成本與儲存成本就相對於較高。SCAM設定的區塊，以字詞(word)為主，其皆有索引，而索引後面連接著一串串鍊(set of postings)，每個串鍊都有兩個欄位，第一個欄位表示出現在第幾篇文件，第二個欄位表示出現幾次。當有新進文件要註冊時，新進文件會先與註冊在資料庫內的文件做比對，利用向量空間(vector space)的概念，以餘弦(cosine)公式做相似度計算，來確認是否有剽竊的行為。實驗證明在某些情況下會比COPS [3]來的簡易與有效。

餘弦公式如方程式(3)所示：

$$sim(R,Q)=\frac{\sum_{i=1}^{N}\partial_i^2 F_i(R)\cdot F_i(Q)}{\sqrt{\sum_{i=1}^{N}\partial_i^2 F_i^2(R)\cdot\sum_{i=1}^{N}\partial_i^2 F_i^2(Q)}} \quad (3)$$

其中∂為第i個chunk之權重向量，$F_i(R)$為chunk i在文件R出現的次數；$F_i(Q)$為chunk i在文件Q出現的次數，$sim(R,Q)$代表文件R與文件Q之相似度。表 2 說明VSM(Vector Space Model)之例子。

表 2 VSM 例子 [16]

假設 R=<a b c>，Q_1=<a b>，Q_2=<a b c>，Q_3=<a b c d e f g h>，$\partial=1$

$$sim(R,Q1)=\frac{1*1+1*1}{\sqrt{3*2}}=0.82$$

$$sim(R,Q2)=\frac{1*1+1*1+1*1}{\sqrt{3*3}}=1$$

$$sim(R,Q3)=\frac{1*1+1*1+1*1}{\sqrt{8*3}}=0.61$$

由上述例子的$sim(R,Q3)$，可以發現，VSM對於子集偵測方面有缺陷，SCAM提出RFM(Relative Frequency Model)修正這個缺點。

RFM公式如方程式(4)所示：

$$subset(R,Q)=\frac{\sum_{w_i\in c(R,Q)}\partial_i^2 F_i(R)\cdot F_i(Q)}{\sum_{i=1}^{N}\partial_i^2 F_i^2(R)} \quad (4)$$

其中$w\in c(R,Q)$代表c字集內共同出現在文件R與Q的字詞。最終相似度計算為$sim(R,Q)=\max\{subset(R,Q),subset(Q,R)\}$。表 3 說明VSM(Vector Space Model)之例子經RFM修正之後的結果。

表 3 RFM 例子 [16]

假設 R=<a b c>，S_1=<a b>，S_2=<a b c>，S_3=<a b c d e f g h>，$\partial=1$

$$sim(R,S1)=1$$

$$sim(R,S2)=1$$

$$sim(R,S3)=1$$

2.2.3 風格分析

風格分析(style analysis)的目的為找出潛藏於文字中的作家風格資訊。風格分析的基本想法為每一位作家擁有自己的寫作風格，而這些寫作的特性難以模仿或操縱，因此可以藉由不同作家間擁有相異的寫作風格來做剽竊偵測 [8]。雖然風格分析無需參考文集(reference corpus)，但它需要訓練跟學習不同作家的寫作規則。人工神經網路(artificial neural networks, ANNs)和基因演算法(genetic algorithms, GAs)可用於分析文章的文字結構與著作者的邏輯習慣 [8]，訓練過後的ANNs和GAs能夠識別出特定作者之風格。

2.2.4 ROUGE

ROUGE之模型是來自BLEU [13]，BLEU聚焦於機器翻譯效果之評估，而ROUGE專注於摘要評估(Gisting Evaluation)。在評估機器翻譯與摘要的成效時，都具備以下的概念，即若候選文件(candidate document)與參考文件(reference document)愈相似時，則表示翻譯或摘要的成效愈好；相同的概念應用於剽竊偵測上，若候選文件與參考文件愈相似時，其剽竊的可能性愈大。ROUGE提出四種衡量方式，分別說明如下：

I. ROUGE-N:N-gram Co-Occurrence Statistics：

其計算公式如方程式(5)所示：

$$ROUGE-N=\frac{\sum_{S\in\{reference\}}\sum_{gram_n\in S}Count_{match}(gram_n)}{\sum_{S\in\{reference\}}\sum_{gram_n\in S}Count(gram_n)} \quad (5)$$

句子中的每一個字(token)為一個gram，而N代表其gram的長度，$Count_{match}$ 為候選文件與參考文件間共同出現的最大gram數。ROUGE-N在比較兩句子間之相似度時，會記錄句子內每一個字與出現之頻率。從參考文件的第一個句子依序開始比較，每一gram與候選文件內的每一句子內的gram做相似度計算，記錄兩文

件間相同的gram數再除以整體的gram數，來計算ROUGE-N的分數。整體而言，若R為參考文件內句子之總數；Q為候選文件內句子之總數，則共比較$R \times Q$次。

II. ROUGE-L: Longest Common Subsequence

首先介紹共同序列的觀念 [6]：假設給定一個序列$X=[x_1, x_2, ..., x_n]$，序列$Z=[z_1, z_2, ..., z_m]$為X的子序列，如果存在X序列的一個嚴格遞增索引序列$[i_1, i_2, ..., i_k]$,其$j=1,2,...,k$表在X序列中的索引值，則$x_{ij}=z_j$。例如$Z=[A,B,C,D]$是$X=[G,A,B,E,C,D,F]$的子序列，其索引序列為[2,3,5,6]。

LCS為兩序列X和Y中最長共同之子字串序列Z， 表 4 為解釋LCS概念之例子。

表 4 LCS之例子

例1
序列1: ABCDEFG
序列2: ABCESSSFG
LCS: ABCEFG
例2
序列1: ABCDEFG
序列2: SSSFGABCE
LCS: ABCE

由例1可知，ABC為序列1和序列2的共同序列，但非最長共同子序列，ABCEFG才是其最長共同子序列；而且找出來的子序列，可容許其它字元的存在；但由例2可知，若兩序列間其字元不是連續出現，就不計入為LCS。

ROUGE-L運算公式如方程式(6)、方程式(7)、方程式(8)所示：

$$R_{lcs}=\frac{LCS(X,Y)}{m} \tag{6}$$

$$P_{lcs} = \frac{LCS(X,Y)}{n} \tag{7}$$

$$F_{lcs} = \frac{(1+\beta^2)R_{lcs} * P_{lcs}}{R_{lcs} + \beta^2 P_{lcs}} \tag{8}$$

其中m、n分別為參考句子X與候選句子Y的長度，將LCS分別除以m、n之長度，可計算出R_{lcs}與P_{lcs}。再透過 F_{lcs}的方式將其正規化，利用 β 調整R_{lcs}與P_{lcs}的重要程度來衡量兩文件間的相似度。

III. ROUGE-W: Weighted Longest Common Subsequence

透過紀錄連續字串的長度(k)當成權重的觀念去修正LCS應用於評估一個文件與多個文件相似度的問題，如表 5 可知，雖然兩個候選序列的LCS分數是一樣的，但事實上選擇候選序列1優於選擇候選序列2，透過ROUGE-W可修正此問題。

表 5 WLCS 例子

參考序列1: ABCDEFG
候選序列1: **ABCD**SSS
候選序列2: **A**E**B**E**C**E**D**

IV. ROUGE-S:Skip-Bigram Co-Occurrence Statistics。

Skip-bigram為ROUGE-N=2的一種變形。skip-bigram定義一跳躍距離(skip distance，d) ，讓d個字(tokens)結合成一個skip-bigram，當d決定了，我們即可找出所有在句子中的skip-bigrams去計算其分數。

2.3 剽竊系統相關研究

Sun[18] 等人提出一中文字轉換成數學表示式的方法，可應用於排版、數位圖書館、網路通信等領域。基於中文字本身之結構，將其中文字的位置關係轉換成運算子，包含六種位置關係左右(left-right)、上下(up-down)、左下(left-down)、左上(left-upper)、右下(right-upper)、包含(whole-enclosed)；而字的結構轉換成運算元的數學表示式， 例如，中文字“阿”轉換成422lr148。

劉奕廷 [26]應用搜尋引擎建立一英文剽竊系統，以文件片段(segment)為系統的偵測單位。主要分三個步驟：

I. 文件片段的擷取(segment extraction)

II. 文件片段的排序(segment ranking)－以TF-IDF計算每一個文件片段的重要性順序，在依照其順序上傳至搜尋引擎，若沒有搜尋到結果，表此文件片段並無剽竊嫌疑；相反的，則表示有剽竊嫌疑並接續後面的處理。

III. 定義剽竊的來源－透過精確比對(exact matching)的演算法計算文件片段與搜尋引擎的摘要資訊的相似度。並為了增進整體系統效率，劉氏在此提出一選擇性的程序－coverage expansion，利用通常一篇文章，其中有多個文件片段有剽竊嫌疑時，其來自於同一個網站的可能性很大的特性，當搜尋引擎尋找到某一個文件片段的結果時，將其網站的內容下載下來，與其他優先度較低的文件片段做精確比對的相似度計算，減少系統運算的時間。

王偉全 [23]採用文件基因序列的概念，針對數位文件設計一中文剽竊系統，比對出可能有剽竊情形之文件。首先至網路上蒐集文件當成比對的文集，將其透過Maximum Matching Algorithm進行詞庫斷詞，取出文件中的動、名詞形成文件基因序列。詞庫斷詞方法說明如下：$W_1W_2...W_n$ W代表一個個中文字，先到詞庫內尋找_W1_是否為一存在的單字詞，之後再檢查_W1W2_是否為一存在的雙字詞，以此類推，直到找出詞庫內最長的多字詞為止。之後採用最長相同連續

子序列(longest common consecutive subsequence，簡稱LCCS)，做兩文件基因序列的相似度比對演算法，並將比對結果儲存至資料庫，表 6為一LCCS之例子。

表 6 LCCS例子

序列1: ABCDEFG
序列2: ABCSSDE
LCCS=ABC

陳建穎 [24]提供一個英文剽竊偵測系統之架構。基於ROUGE[10]的系統雛型，將ROUGE的六種方法unigram(N=1-4)、最長共同子序列(longest common subsequence，LCS)、和 skip-bigram應用於剽竊偵測上。當N=1時，N-gram co-occurrence statistics可偵測逐字的剽竊，且不受句子順序改變而影響剽竊偵測成效，但卻有對於刻意針對原文做增減文字而導致剽竊偵測成效不佳之弱點，藉由最長共同子序列和skip-bigram之演算法克服。

第三章　中文剽竊偵測系統實作

在本章中將闡述本研究所建置之中文剽竊系統的架構。圖 3 為系統流程示意圖，本研究透過Google搜尋引擎的應用程式介面(Application Programming Interface, API)，將經過前置處理步驟後的中文文件傳入Google Ajax Search API，分析Google傳回的JSON編碼結果，將其儲存的文句與候選文句做相似度比較，若相似度高於某一門檻值，便認定為可能剽竊之文句。

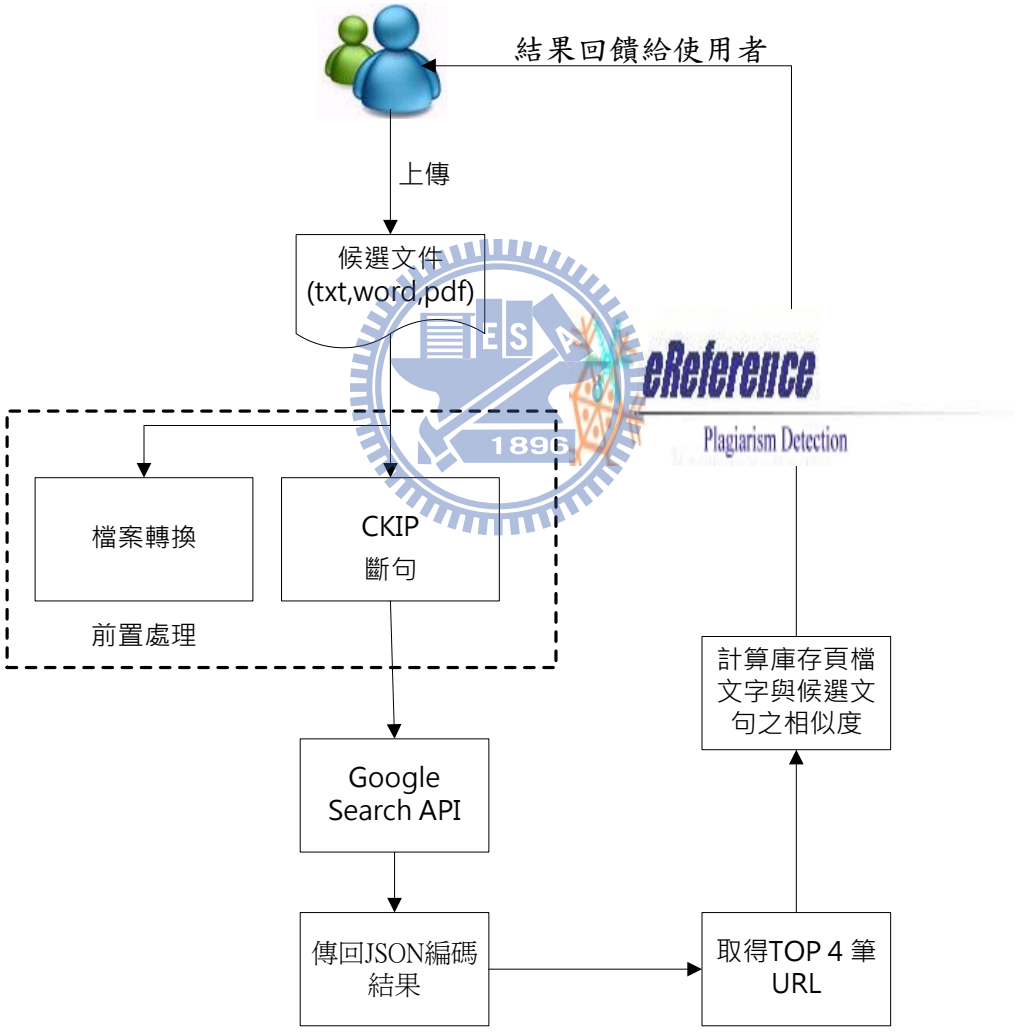

圖 3：系統流程示意圖

3.1 前置處理

前置處理之主要目的為將長篇的中文文件斷成句子。本研究的前置處理步驟包含：檔案轉換、中文斷句處理。

3.1.1 檔案轉換

鑒於Microsoft office 與 Adobe Acrobat 軟體易於編輯、瀏覽便利以及可使排版美觀等特性，使得doc、pdf成為常用的檔案格式。為了讓使用者使用便利，本系統透過Apache POI [1]API將Microsoft Office檔案轉換成字串的形式；以及利用PDFBOX此一JAVA語言所寫就的開放原始碼函式庫，來擷取PDF檔案中的文字內容，以利做後續之處理，使用者不需自己轉換檔案後再上傳至系統介面。

3.1.2 斷句

斷句的目的在於找出文章段落之間的分界，將長篇的文章，斷成以句子為單位。中文是以「字(character)」為基礎的語言，字詞的組合、意義十分多樣化，相較於英文具有其規則性，中文字詞間界線較難定義。本研究採用中央研究院中文詞知識庫小組(Chinese Knowledge Information Processing Group, CKIP)所研發之中文斷詞系統 [22]進行斷句。方法為利用中研院所提供的API，在用戶端撰寫程式經由 TCP Socket 連線傳送驗證資訊將文章送至中研院中文斷詞系統的伺服器，伺服器將處理結果經由原連線傳回，資料的交換方式採用 XML。

中文斷詞系統傳回之結果以XML格式表示，將每個斷出來的詞、標點符號以括弧標示詞性(part-of-speech)，每個詞之間以全形空白隔開。表 7 為輸入CKIP中文斷詞切字系統的原文；表 8 為經由CKIP處理後的輸出結果。

表 7 CKIP中文斷詞切字系統原文實例

冰島一座沈寂了兩百年的火山，突然再度噴發，引爆了雙重危機，岩漿導致冰河融化，可能引發大洪水，還有大量火山灰如果持續飄到高空，還可能造成全球進入寒冬。沈睡了兩百多年的艾雅法拉冰河火山醒過來了，而且五天以來，噴發頻率越來越密集，地質專家直到週四天氣略微好轉，才得以搭乘直升機冒險逼近火山口就近觀察，發現噴發出來的岩漿，已經把冰河融出一道大約500公尺長的裂縫。

表 8 CKIP中文斷詞切字系統實例-斷詞切字與詞性標記結果

冰島(N) 一(DET) 座(M) 沈寂(Vi) 了(Di) 兩百(DET) 年(M) 的(T) 火山(N) ，(COMMACATEGORY) 突然(ADV) 再度(ADV) 噴發(Vt) ，(COMMACATEGORY) 引爆(Vt) 了(Di) 雙重(A) 危機(N) ，(COMMACATEGORY) 岩漿(N) 導致(Vt) 冰河(N) 融化(Vt) ，(COMMACATEGORY) 可能(ADV) 引發(Vt) 大(Vi) 洪水(N) ，(COMMACATEGORY) 還(ADV) 有(Vt) 大量(DET) 火山灰(N) 如果(C) 持續(Vt) 飄到(Vt) 高空(N) ，(COMMACATEGORY) 還(ADV) 可能(ADV) 造成(Vt) 全球(N) 進入(Vt) 寒冬(N) 。(PERIODCATEGORY) 沈睡(Vi) 了(Di) 兩百多(DET) 年(M) 的(T) 艾雅法拉(N) 冰河(N) 火山(N) 醒過來(Vi) 了(T) ，(COMMACATEGORY) 而且(C) 五(DET) 天(M) 以來(POST) ，(COMMACATEGORY) 噴發(Vt) 頻率(N) 越來越(ADV) 密集(Vi) ，(COMMACATEGORY) 地質(N) 專家(N) 直到(P) 週四(N) 天氣(N) 略微(ADV) 好轉(Vi) ，(COMMACATEGORY) 才(ADV) 得以(ADV) 搭乘(Vt) 直升機(N) 冒險(Vi) 逼近(Vt) 火山口(N) 就近(ADV) 觀察(Vt) ，(COMMACATEGORY) 發現(Vt) 噴(Vt) 發出來(Vt) 的(T) 岩漿(N) ，(COMMACATEGORY) 已

經(ADV)　把(P)　冰河(N)　融出(Vt)　一(DET)　道(M)　大約(ADV) 500(DET)　公尺(M)　長(Vi)　的(T)　裂縫(N)　。(PERIODCATEGORY)

經過詞性標記後，以標點符號(COMMACATEGORY、PERIODCATEGORY)為斷句依據。在將斷句後的句子傳入Google AJAX Search API之前，系統會先將小於8個字的句子省略，因為字數少的句子可能是一些發語詞，例如：雖然、然而、但是或者是不具代表性的句子，類似於英文的停用字(Stop Word)，省略該等句子可減少 False Positive的機率；相反的當句子超過56個字以上，Google AJAX Search API 無法處理，系統會自動尋找句子前一個除逗點、句號之後的標點符號來進行斷句。

3.2 運用Google搜尋引擎檢索相關字句

本研究所實作的中文剽竊偵測系統，其運用Google搜尋引擎的流程如下：

1. 將使用者上傳至系統的文章傳入Google AJAX Search API。
2. 將Google傳回的JSON資料格式做分析，取得URL，將其與候選文句做剽竊偵測的比對，超過門檻值的文句會被紀錄下來。
3. 以人工檢視結果。

3.2.1 Google Ajax Search API

本研究利用 Google AJAX Search API [9]進行第一階段相似度篩選，Google AJAX Search API 使用 JavaScript 將「Google 搜尋」放入自身的網頁。嵌入一個簡單的動態搜尋框，並在自身的網頁上顯示搜尋結果，圖 4 為搜尋引擎傳回結果之簡單範例。

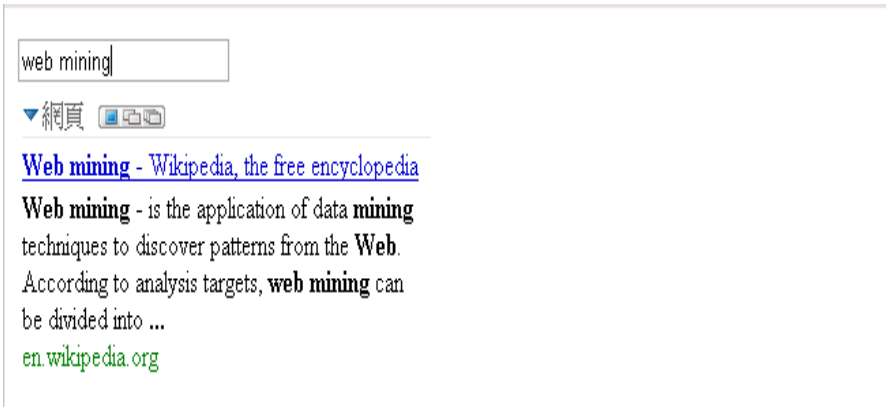

圖 4：Google Search API傳回結果之簡單範例

但針對於其他非JavaScript環境，Google AJAX Search API提供原始RESTful介面，可以傳回大部分語言和執行程式皆能夠處理的JSON編碼結果。以下是一個範例查詢：

ajax.googleapis.com/ajax/services/search/web?v=1.0&lr=lang_zh-TW&q=

可透過調整此 URL 格式，抓取不同格式如影音、圖片等搜索結果，並可設定其所搜尋的語言。

透過Google AJAX 搜尋API所提供的RESTful介面，可傳回JSON編碼結果。表 9 是一個JSON編碼結果範例：

表 9 JSON編碼結果範例

查詢：

http://ajax.googleapis.com/ajax/services/search/web?v=1.0&lr=lang_zh-TW&q=冰島一座沈寂了兩百年的火山

結果：

responseData{"results":[{"GsearchResultClass":"**GwebSearch**","**unescapedUrl**":"http://atlantischiu.blog.ithome.com.tw/post/3058/68926","**url**":"http://atlantischiu.blog.ithome.com.tw/post/3058/68926","**visibleUrl**":"atlantischiu.blog.ithome.com.tw","**cacheUrl**":"http://www.google.com/search?q=cache:baU6FuTAiMsJ:atlantischiu.blog.ithome.com.tw","**title**":"亞特蘭提斯| <b>冰島火山<\/b>爆發，沉睡了兩百多年的<b>火山<\/b>醒來了，全球皮皮剉！

","**titleNoFormatting**":"亞特蘭提斯| 冰島火山爆發，沉睡了兩百多年的火山醒來了，全球皮皮剉！","**content**":"<b>冰島一座沈寂了兩百年的火山<\/b>，突然再度噴發，引爆了雙重危機，岩漿導致冰河融化，可能　引發大洪水，還有大量火山灰如果持續飄到高空，還可能造成全球進入寒冬。 <b>...<\/b>"}

{"GsearchResultClass":"**GwebSearch**","**unescapedUrl**":"http://news.cts.com.tw/cts/international/201003/201003250436566.html","**url**":"http://news.cts.com.tw/cts/international/201003/201003250436566.html","**visibleUrl**":"news.cts.com.tw","**cacheUrl**":"http://www.google.com/search?q=cache:IcoKWaGF0KMJ:news.cts.com.tw","**title**":"<b>冰島火山<\/b>爆發全球皮皮剉!-華視新聞網","**titleNoFormatting**":"冰島火山爆發全球皮皮剉!-華視新聞網","**content**":"2010年3月25日 <b>...<\/b> <b>冰島一座沈寂了兩百年的火山<\/b>，突然再度噴發，引爆了雙重危機，岩漿導致冰河融化，可能　引發大洪水，還有大量火山灰如果持續飄到高空，還可能造成全球 <b>...<\/b>"}

{"GsearchResultClass":"GwebSearch","unescapedUrl":"http://weihsinchiu.pixnet.net/blog/post/5668226","url":"http://weihsinchiu.pixnet.net/blog/post/5668226","visibleUrl":"weihsinchiu.pixnet.net","cacheUrl":"http://www.google.com/search?q=cache:U2zYgoHipDUJ:weihsinchiu.pixnet.net","title":"<b>冰島火山<\/b>爆發，沉睡了兩百多年的<b>火山<\/b>醒來了，全球皮皮剉！@ 維新生活 <b>...<\/b>","titleNoFormatting":"冰島火山爆發，沉睡了兩百多年的火山醒來了，全球皮皮剉！ @ 維新生活 ...","content":"<b>冰島一座沈寂了兩百年的火山<\/b>，突然再度噴發，引爆了雙重危機，岩漿導致冰河融化，可能　引發大洪水，還有大量火山灰如果持續飄到高空，還可能造成全球進入寒冬。 <b>...<\/b>"}

{"GsearchResultClass":"GwebSearch","unescapedUrl":"http://tw.myblog.yahoo.com/ihsien-2012/article?mid=23698&prev=23699&next=-1","url":"http://tw.myblog.yahoo.com/ihsien-2012/article%3Fmid%3D23698%26prev%3D23699%26next%3D-1","visibleUrl":"tw.myblog.yahoo.com",

"cacheUrl":"http://www.google.com/search?q=cache:_cDT5FSFanIJ:tw.myblog.yahoo.com","title":"<b>冰島火山<\/b>爆發全球皮皮剝! - 您好,這是一個分享的部落格!!! - Yahoo <b>...<\/b>","titleNoFormatting":"冰島火山爆發全球皮皮剝! - 您好,這是一個分享的部落格!!! - Yahoo ...","content":"<b>冰島一座沈寂了兩百年的火山<\/b>，突然再度噴發，引爆了雙重危機，岩漿導致冰河融化，可能　引發大洪水，還有大量火山灰如果持續飄到高空，還可能造成全球進入寒冬。 <b>...<\/b>"}]

"cursor":{"pages":[{"start":"0","label":1},{"start":"4","label":2},{"start":"8","label":3},{"start":"12","label":4},{"start":"16","label":5},{"start":"20","label":6},{"start":"24","label":7},{"start":"28","label":8}],"estimatedResultCount":"75","currentPageIndex":0,"moreResultsUrl":"http://www.google.com/search?oe=utf8&ie=utf8&source=uds&start=0&lr=lang_zh-TW&hl=en&q=%E5%86%B0%E5%B3%B6%E4%B8%80%E5%BA%A7%E6%B2%88%E5%AF%82%E4%BA%86%E5%85%A9%E7%99%BE%E5%B9%B4%E7%9A%84%E7%81%AB%E5%B1%B1"}}

透過此JSON編碼結果，可分析出由Google傳回的有關於資料探勘的前4筆資料，以利進行後續相似度計算。表 10 與表 11 為Google AJAX Search API傳回的JSON參數之解釋。

表 10 results參數解釋

參數	說明
GwebSearch	網頁搜尋
unescapedUrl	網頁的原始網址
url	原始網址的escaped version，會對一些特殊的字元進行編碼
visibleUrl	縮短版的網址，省略掉協定與路徑
title	網頁搜尋結果的標題
titleNoFormatting	網頁搜尋結果的標題，省略掉HTML的標記
content	關於搜尋結果網頁的簡短描述
cacheUrl	Google對web網頁的快取版本

表 11 cursor參數解釋

參數	說明
pages	Google AJAX Search API 預設傳回的搜尋結果頁數，網頁搜尋支援8頁。
start	搜尋結果的陣列索引參數
label	搜尋結果的陣列索引參數
estimatedResultCount	預估符合使用者所下之查詢的結果數
currentPageIndex	搜尋結果的陣列索引參數
moreResultsUrl	更多的搜尋結果網址

3.3 相似度計算

透過 Google 搜尋引擎傳回的結果，會有摘要資訊，如圖 5 所示。在上一小節提及的 JSON 編碼結果包括了其摘要資料，透過比對候選文句與此摘要資訊相似度，決定其是否為可能之剽竊句子。

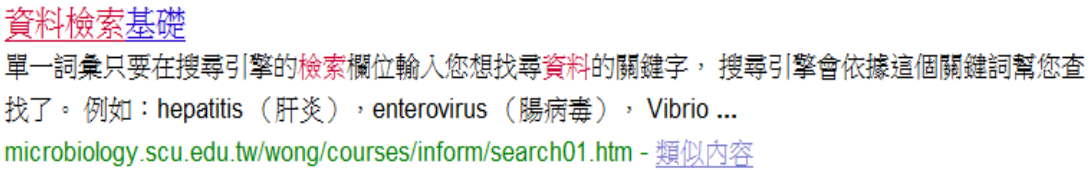

圖 5：Google搜尋引擎傳回結果示意圖

3.3.1 最長共同子序列

本研究利用最長共同子序列(Longest Common Sequence, LCS)的演算法作為相似度計算的核心。

方程式(9)為未經本研究修正的LCS計算方式，S^R表使用者所上傳的文章斷成的候選文句，S^G表Google搜尋引擎傳回的摘要資訊文句，計算其共同最長共同之子字串當成分子，除以S^R_{length}之候選文句之長度。但若僅僅除以候選文句之長度，會造成嚴重假警報(false positive)的問題，後續實驗部份會再詳述。因此本研究提出利用ROUGE-W概念所修正的LCS公式與ROUGE-L公式，來降低系統整體假警報機率。方程式(10)為本研究之修正的LCS公式，將候選文句長度與S^L_{length}—Google搜尋引擎傳回的摘要資訊文句中最長連續出現的關鍵字長度，將連續關鍵字之間的距離考慮進去，取最大值當成分母。方程式(11)~(13)為ROUGE-L公式，其β參數設為一，可將S^R_{length}與S^G_{length}之長度皆納入考慮，計算 F-measure之分數。上述方法說明如表 12 所示，當計算出的LCS分數高於系統所設定的門檻值，就認定為具有剽竊嫌疑的句子。

$$LCS(S^R, S^G) = \frac{LCS(S^R, S^G)}{S^R_{length}} \tag{9}$$

$$LCS_{revised}(S^R, S^G) = \frac{LCS(S^R, S^G)}{\max(S^R_{length}, S^L_{length})} \tag{10}$$

$$\begin{cases} LCS_{rrouge}(S^R, S^G) = \dfrac{LCS(S^R, S^G)}{S^R_{length}} & (11) \\ LCS_{rprec}(S^R, S^G) = \dfrac{LCS(S^R, S^G)}{S^G_{length}} & (12) \\ F_{lcs} = \dfrac{2 * LCS_{rrouge}(S^R, S^G) * LCS_{rprec}(S^R, S^G)}{LCS_{rrouge}(S^R, S^G) + LCS_{rprec}(S^R, S^G)} & (13) \end{cases}$$

表 12 LCS分數計算之例子

S^R ：“針灸、傷科給付一直都是擇一申報，”
S^G ：“**傷科**推拿一事是錯的說法。自勞保、健保開辦以來，**針灸、傷科給付 一直都是擇一申報**，也就是說如果病患就醫經中醫師診斷後，**針灸**不推拿與針灸”
S^R與S^G之LCS為“針灸、傷科付一直都是擇一申報，”，利用方程式(9)計算分數為15/15=1；方程式(10) 分數為15/15=1，在此分母會取S^G最長連續的子字串“針灸、傷科付一直都是擇一申報”之長度，並不會將前面傷科與後面針灸不連續的關鍵字長度計入；ROUGE-L分數為 $\frac{2*1*0.23}{(1+0.23)} = 0.38$

3.3.2 全文相似度計算

經由上一小節的LCS公式計算，將其分數高於系統設定門檻值之句子，代表具有剽竊嫌疑的句子，放入plaset子集中儲存，依據其進行使用者上傳文章的全文剽竊程度的計算。

方程式(14)為全文相似度之計算，將具有剽竊嫌疑的句子之長度總和除以 $Fulltext_{length}$ 文章總長度，計算出全文剽竊程度的分數。

$$Fulltext(sim) = \frac{\sum_{S^R \in plaset} S^R_{length}}{Fulltext_{length}} \tag{14}$$

第四章 系統發展與結果分析

本章描述系統雛型發展，及驗證本系統實作應用的準確度。

4.1 系統簡介

本研究的系統經由分析使用者上傳之內容，透過Google搜尋引擎做第一階段的相似度，傳回相關之結果，再利用LCS演算法萃取出具有剽竊嫌疑的句子；綜合以上兩者之結果，以瀏覽器(browser)將偵測結果呈現給使用者，最後依人工判斷是否有剽竊之嫌疑。實驗環境如表 13 所示。

表 13 實驗環境

	PC
硬體環境	Pentium(R) Dual-Core CPU E6300 2.80GHz with 1.96 GB RAM
作業系統	Windows XP Profession Edition sp3
系統使用工具	Eclipse 3.5.1 for Java jdk1.6.0 Tomcat 5.5 CKIP、Apache POI、PDFBOX、JSON.

4.1.1 系統介面介紹

系統分成兩種方式讓使用者輸入欲偵測剽竊的中文文件，如圖 6 所示，第一種(圖 6上方)是透過上傳檔案的方式;第二種(圖 6下方)則是讓使用者輸入文字的方式進行剽竊偵測，但限制字數必須少於500個字。

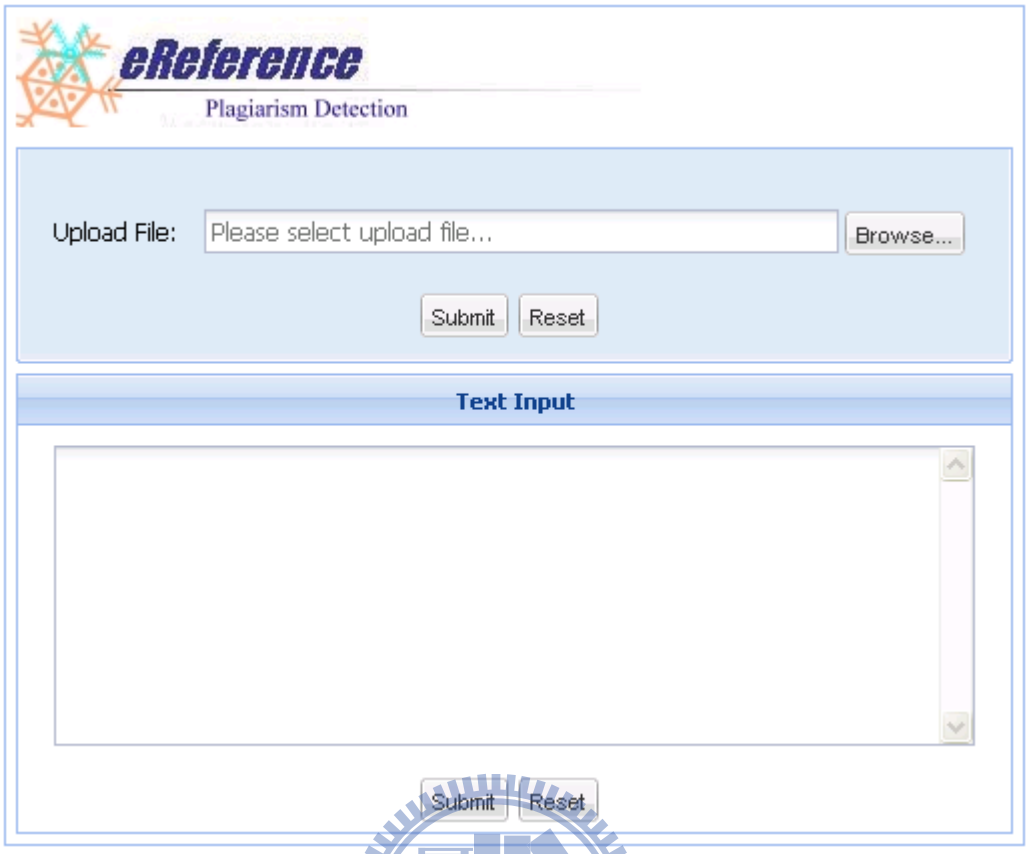

圖 6：系統上傳檔案介面

傳送完使用者上傳之檔案後，系統會將該檔案與Google搜尋引擎回傳的資料加以比較，並將比較結果顯示於系統上，如圖 7 所示，右方為偵測結果分析與使用者上傳之文章原文內容，使用者可透過系統所提供資訊進行是否有剽竊行為的判斷。左方為被偵測出有參考自網路的句子，會對照到右方的原文，顯示成超連結型式。其左、右方點閱後可查閱Google搜尋引擎回傳資料之網址標題與摘要資訊，藍色字為網址標題、紅色字部份表示摘要資訊中找到之目標文字，如圖 8 所示；

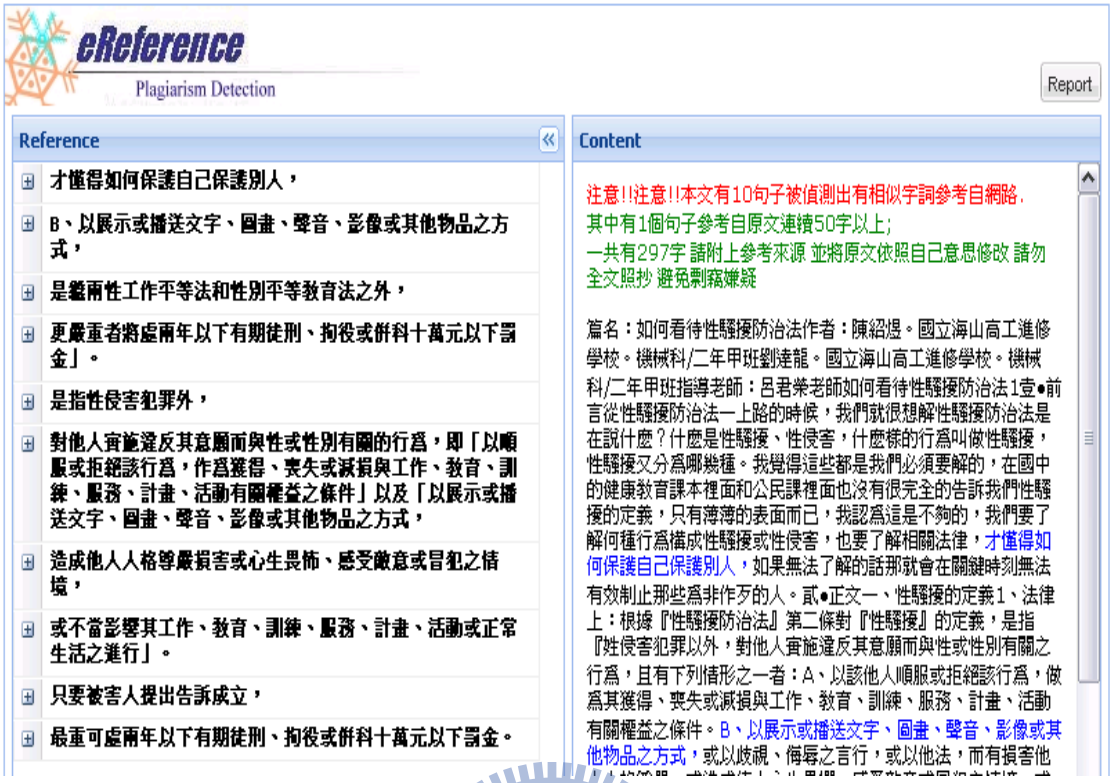

圖 7：系統偵測結果

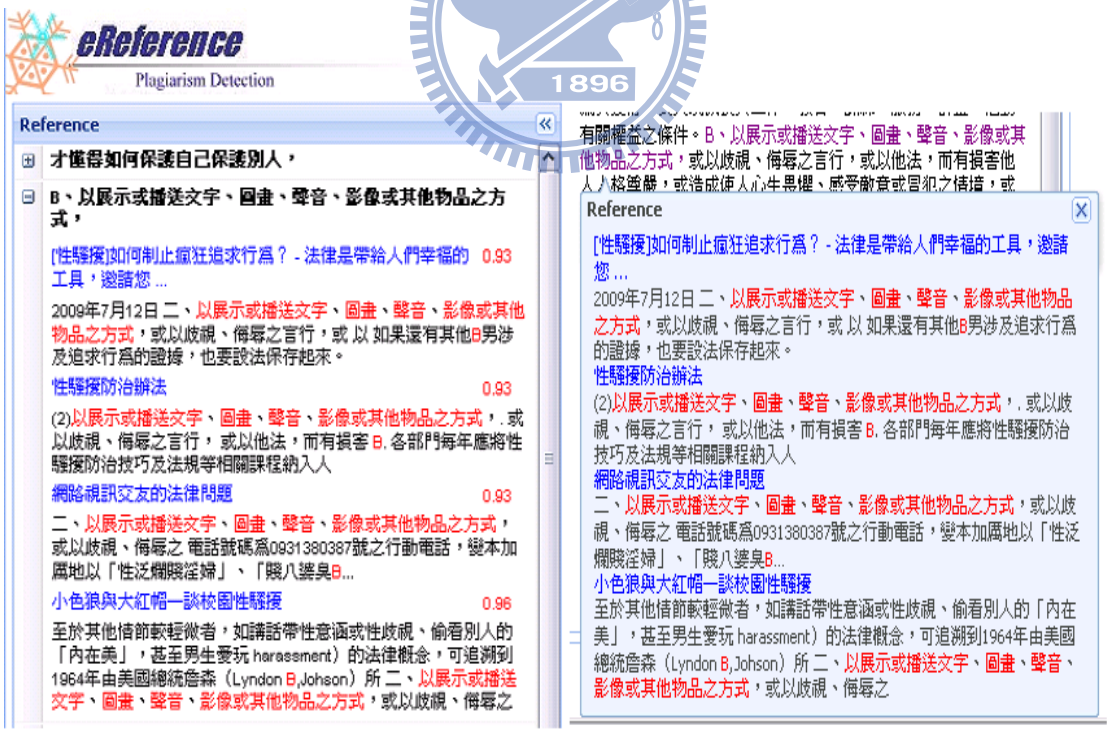

圖 8：目標網址與摘要資訊

使用者經人工判斷完是否有剽竊嫌疑之後，可透過右上方之Report按紐，將

原文與剽竊偵側結果以PDF檔輸出，以利支援後續的處理。圖 9 為一PDF檔案輸出範例，Content代表使用者上傳之原文內容，Reference部分是原文內容參考到的網頁網址與摘要資訊，以參考文獻之型式呈現。

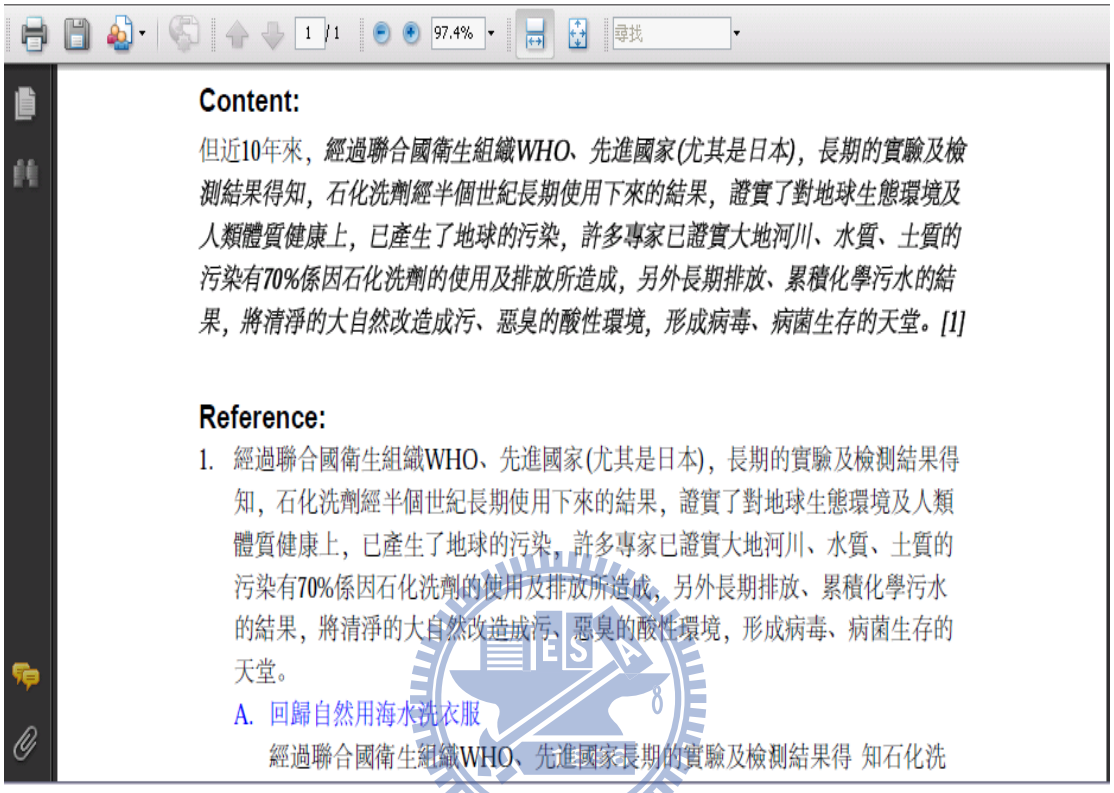

圖 9：PDF輸出範例

4.2 實驗

中文剽竊偵測的領域中，並無標準的公開性剽竊文集。所以本研究利用人力來組成剽竊文集與非剽竊文集來驗證其成效。剽竊文集收集網路上新聞、網誌、維基百科之文章，每一個類別25篇，共75篇，由於系統是利用Googel搜尋引擎做第一階段的相似度篩選，所以複製網路上的某一篇文章，必然可在網路上搜尋到相同的文章；另外透過人工檢視篩選全國高級中等學校小論文寫作比賽之小論文[21]共25篇當成未剽竊之文集，進行未經修正的LCS公式、修正LCS公式、ROUGE-L公式之系統正確性驗證。本實驗的目的為二：

1. 透過比較實驗結果，檢視三種LCS公式在不同門檻值之資料集上，其正

確率與假警報機率為何？

2. 推薦一適宜之門檻值。

4.2.1 未經修正的 LCS 公式實驗

透過上傳剽竊文集的75篇文件與未剽竊之文集的25篇文件至系統比對後，在門檻值0.5、0.6、0.7、0.8，其正確率趨勢圖如圖 10 所示：

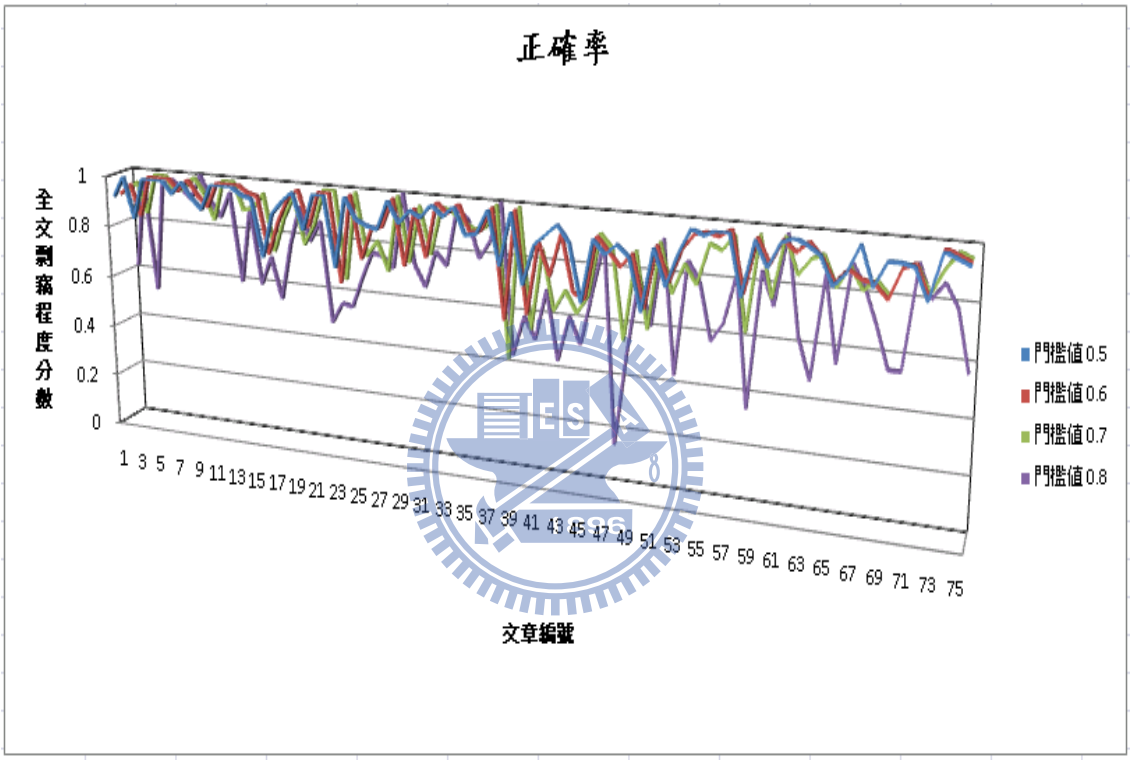

圖 10：未經修正的正確率之趨勢圖

上圖橫軸為每一篇文章的文章編號、縱軸為透過方程式(9)篩選出具有剽竊嫌疑的句子，再利用方程式(11)算出的全文剽竊分數，全部內容相似的文章其分數為1.0，相反的分數為0。上傳之75篇文章皆為複製於網路的剽竊文章，其全文剽竊分數應為1.0，所以計算出的全文剽竊分數可代表系統之正確性比率，以此類推，非剽竊文章的分數應為0，非剽竊文集的25文章計算出的全文剽竊分數可代表假警報機率。由圖 10 可知門檻值0.8之紫色線，其正確率趨勢線震盪幅度很大，甚至有一篇全文剽竊分數低於0.2，其偵測效率不佳；而門檻值0.6、0.7之

紅色、綠色趨勢線十分的接近，雖然震盪幅度不比門檻值0.8大，但在某些文章上面，偵測效率依舊不夠理想。整體而言，正確率最高的是門檻值0.5；圖 11 為各門檻值的假警報機率，可看出其假警報與正確率趨勢相反，其門檻值越高，假警報機率越低，正確率最高的門檻值0.5，其假警報機率反彈最大。

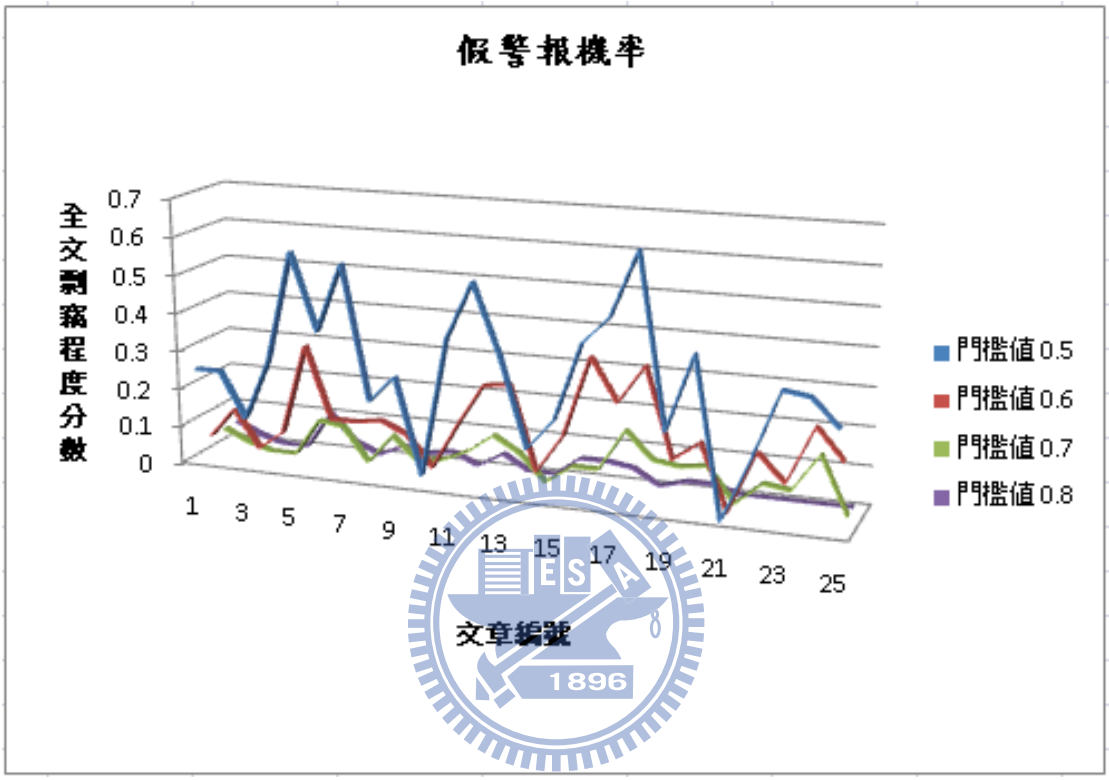

圖 11：未經修正的假警報趨勢圖

綜合上述，其平均正確率與平均假警報機率如表 14 所示，可知在門檻值0.5與0.6時，其假警報機率偏高。

表 14 未經修正的各門檻值之平均正確率與平均假警報機率

	門檻值0.5	門檻值0.6	門檻值0.7	門檻值0.8
平均正確率	0.93	0.90	0.86	0.72
平均假警報機率	0.31	0.15	0.05	0.02

4.2.2 問題分析

本系統利用未經修正的LCS公式的偵測結果中，發現其假警報機率之結果偏高，透過案例的方式來說明其可能因素，並利用提出的修正公式與ROUGE-L公式改善之。

例如：上傳之候選文句為“只有薄薄的表面而已，”，圖 12 為系統傳回之結果，可以得知其最長共同子序列為“只有表面而已”，原本LCS公式將其除以候選文句的長度10，得到0.6的相似度。但經由人工的判斷，這句並非有剽竊嫌疑句子。造成此種情況的可能原因為：候選文句的句子長度本身不長，又加上使用的字是較為普遍的字詞，如我們、常常，只是...等等，容易導致假警報的情況。為此本研究採用ROUGE-L公式，將兩句文句的長度皆納入考慮與利用ROUGE-W概念修正的LCS公式，將關鍵字之間的距離納入考量，以此範例為例，ROUGE-L計算出的相似度分數為 $\frac{2*0.6*0.26}{(0.6+0.26)}=0.36$，而修正後的LCS公式會取“**只有**11.4mm厚，在Windows Mobile希望不會是集嘉做做**表面**工夫**而已**”之長度當成分母，相似度分數為6/23=0.26，降低整體系統假警報的情況。

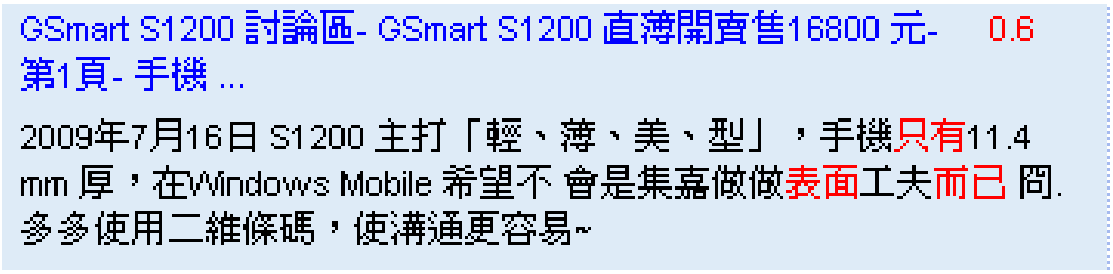

圖 12：假警報個案例子

4.2.3 ROUGE-L 與修正後的 LCS 公式實驗

重新上傳剽竊文集的75篇文件與未剽竊之文集的25篇文件至系統比對後，在門檻值0.5、0.6、0.7、0.8，其ROUGE-L與修正後LCS公式之正確率趨勢圖，如圖 13、圖 14 所示：

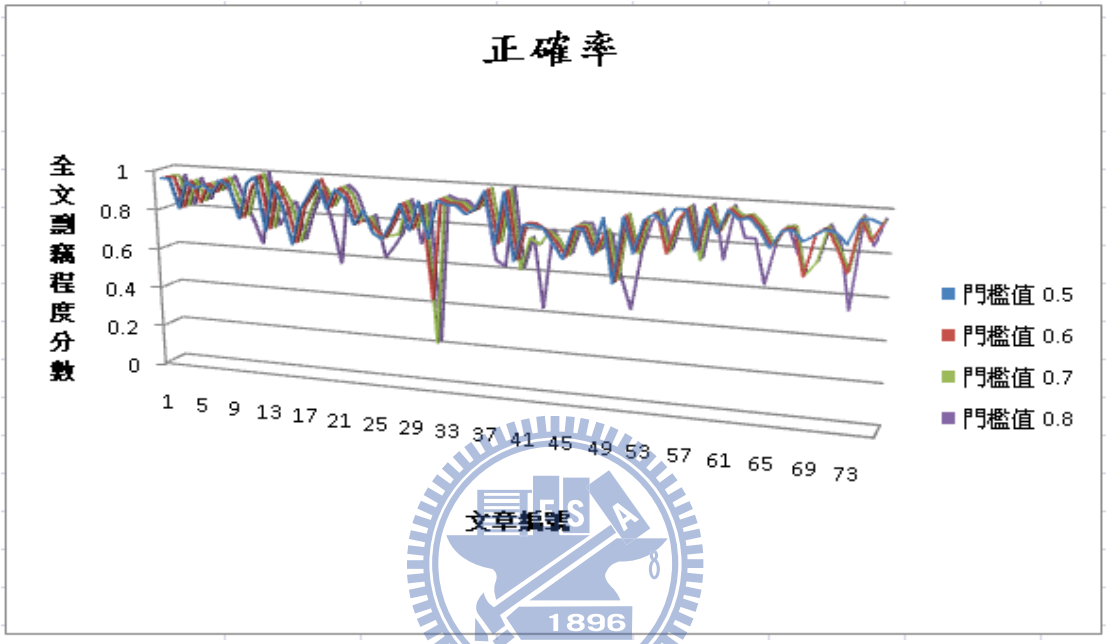

圖 13：ROUGE-L正確率趨勢圖

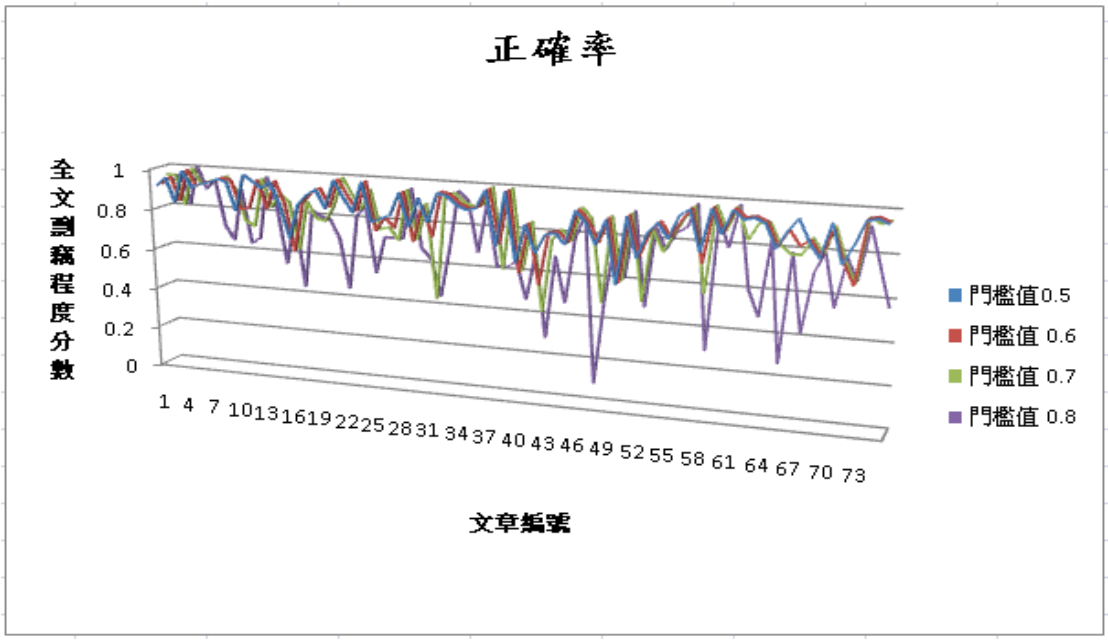

圖 14：修正後LCS的正確率趨勢圖

將圖 13、圖 14 比較，趨勢線相當相似，但在門檻值0.8之紫色線，修正後LCS正確率之趨勢線震盪幅度依然很大，其偵測效率不如於ROUGE-L佳，這是由於ROUGE-L在部份相似的句子計算的相似度分數會高於修正後LCS公式；但在某些情況下，某些關鍵字的出現沒有意義(字數短又不連續)，爾後又出現一串較長的連續關鍵字，且關鍵字之間的距離差異大時，修正後LCS會優於ROUGE-L，上述兩種情況舉例於表 15。門檻值0.5、0.6、0.7之藍色、紅色、綠色趨勢線並無太大的差異，整體而言，正確率最高的還是門檻值0.5；圖 15、圖 16 為ROUGE-L及修正後LCS各門檻值的假警報機率，可看出其假警報與正確率趨勢依然是相反的，其門檻值越高，假警報機率越低，但相對於圖 11 的未經修正的LCS公式的假警報機率實驗結果，明顯看出ROUGE-L與修正過LCS公式的假警報趨勢圖平坦許多，整體的假警報機率降到0.2以下，大幅改善假警報的情形。

表 15 ROUGE-L與修正後LCS計算相似度分數之例子

部份相似的情況：
上傳之候選句子：“公文只是請當事人到局來說明。”
Google傳回之參考句子：“公文只是請當事人到”
其LCS長度為9，ROUGE-L分數為0.78；修正後LCS分數為0.64。
關鍵字的出現沒有意義的情況：
上傳之候選句子：“不過國稅局所發出公文受文者寫的竟然是「飯糰」，”
Google傳回之參考句子：“**飯糰**的老夫婦，由於未辦理營業登記， 被國稅局開單告發，**不過國稅局所發出公文受文者寫的竟然是**”
其LCS長度為15，此例中，**飯糰**的出現因為為非連續的關鍵字所以沒有意義。ROUGE-L分數為0.54；修正後LCS分數為0.78。

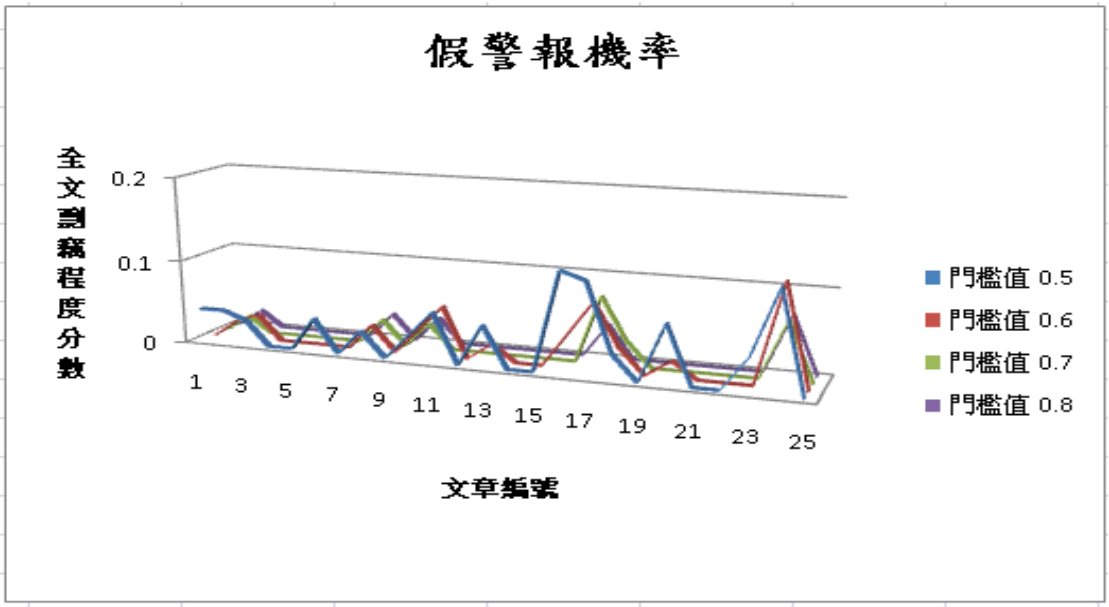

圖 15：ROUGE-L的假警報趨勢圖

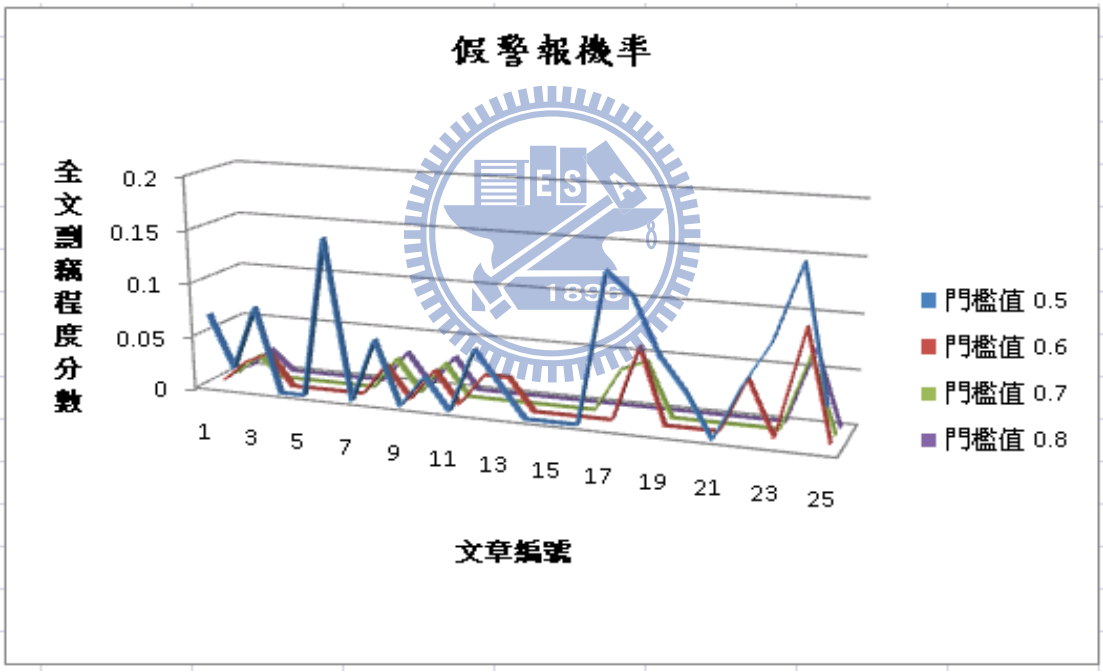

圖 16：修正後LCS的假警報趨勢圖

概觀其兩種公式的實驗結果，其平均正確率與平均假警報機率如表 16 與表 17 所示，其假警報的機率大幅的降低至0.05以下；但由於修正了整體的假警報機率，其整體正確率也向下修正，但皆有在八成以上，其適宜的門檻值為0.6，可達到系統的最佳偵測效率。

表 16 兩種公式的各門檻值之平均正確率

	未經修正的LCS	ROUGE-L	修正後LCS
門檻值0.5-平均正確率	0.93	0.88	0.89
門檻值0.6-平均正確率	0.90	0.86	0.87
門檻值0.7-平均正確率	0.86	0.85	0.84
門檻值0.8-平均正確率	0.72	0.80	0.71

表 17 兩種公式的各門檻值之平均假警報機率

	未經修正的LCS	ROUGE-L	修正後LCS
門檻值0.5-平均假警報機率	0.31	0.03	0.05
門檻值0.6-平均假警報機率	0.15	0.02	0.02
門檻值0.7-平均假警報機率	0.05	0.01	0.01
門檻值0.8-平均假警報機率	0.02	0.01	0.01

4.3 討論與分析

在本系統的驗證中，有些偵測結果十分匪夷所思，探究其可能因素涵蓋許多面向，以下就幾個案例說明之。

I. 案例一：圖 17 為本研究實驗剽竊文集中的一篇新聞文章，發現最後一句－”歐巴馬在紐約市長彭博（Michael Bloomberg）和警察局長凱利（Ray Kelly）的陪伴下說：「國家、市長、局長和總統都以你們為榮。」”，無法被偵測出來，其可能因素為：句子夾雜著英文字詞，文章在經由CKIP斷句取回後，其英文字之間的空白會消失，如“Michael Bloomberg”變成“MichaelBloomberg”，使得Google搜尋引擎無法搜尋到相關之結果。並且由於英文是以字詞

(word)為單位，但系統的運作卻是透過字元(char)的方式進行處理，若句子中含有太多的英文字，也會影響系統相似度的計算。

此句在本系統中會被斷成兩句，”歐巴馬在紐約市長彭博（MichaelBloomberg）和警察局長凱利（RayKelly）的陪伴下說：「國家、”與” 市長、局長和總統都以你們為榮。」”，將其實際傳入一般搜尋引擎介面，結果如圖 18 所示，可知Google搜尋引擎無法傳回相關之結果，使得系統無法比對出來。

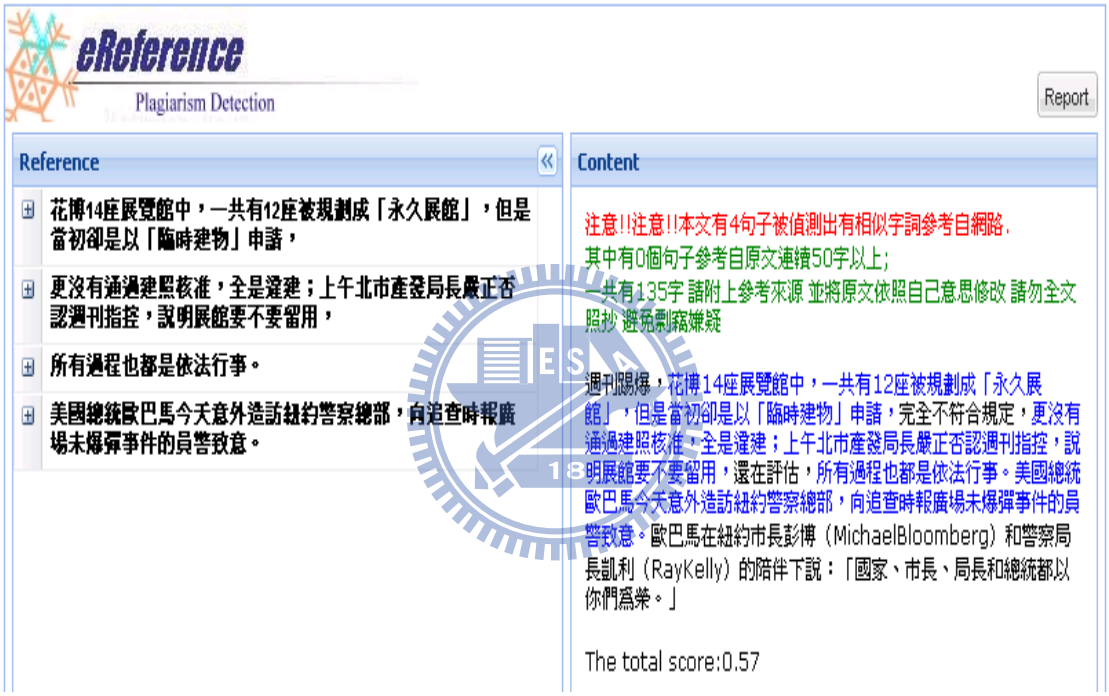

圖 17：案例範例一

圖 18：Google一般介面搜尋結果(範例一)

II. 案例二：中文編碼問題

有些文章偵測結果，明明在Google的一般介面搜尋得到，可是系統卻無法比對出來，如圖 19 的最後一句－”我問她：「今天好玩嗎？」她露出笑容對我點點頭。”，檢視一般介面搜尋結果如圖 20 所示，並無案例一的情形，可搜尋到相關結果。

實際到系統中，查看Google傳回的JSON編碼結果，我們可以發現傳回摘要資訊內容為「2010年5月12日　就在這個時間，我發現人群外的一個小女生，靜靜的站在那裡，用木然的表情看著我，我停住了與這群孩子的對話，走近那孩子。　：「？」　」，後面的句子變成了問號，造成系統無法做後續的相似度比對。

由於分析JSON傳回的編碼結果過程中，有些中文字或帶有特殊字元、符號的句子，系統無法解析或還原回來其編碼所造成的，這是未來可改善的方向之一。

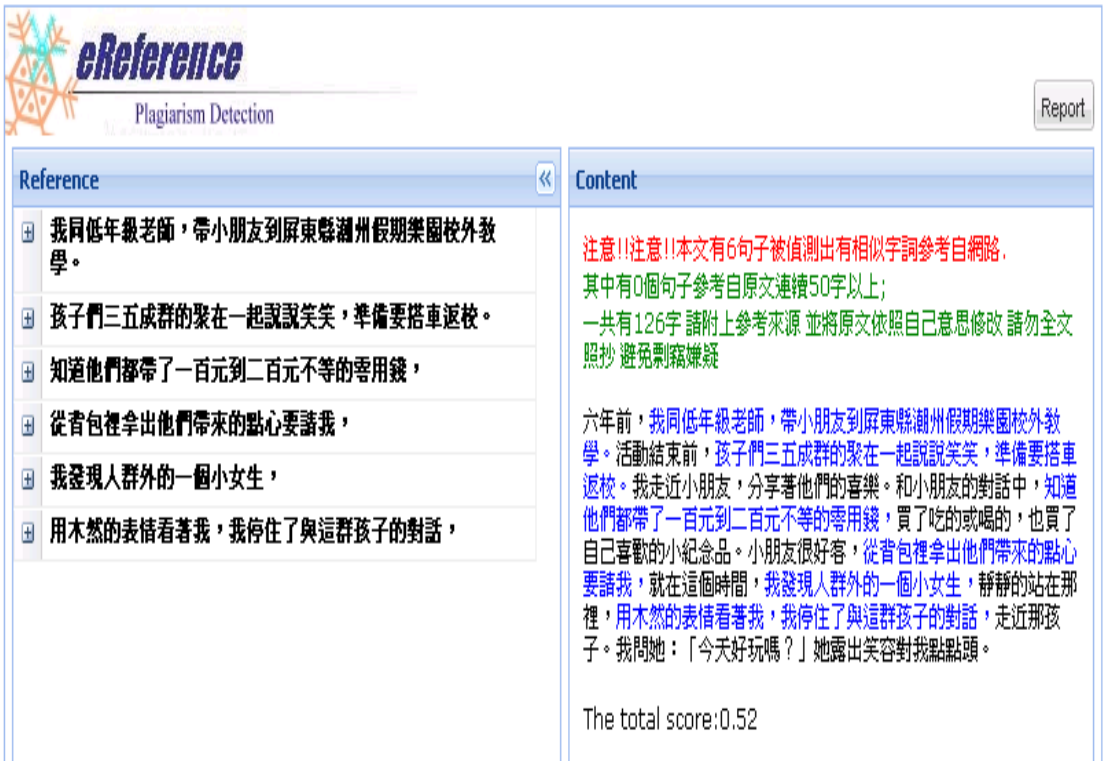

圖 19：案例二範例

圖 20：Google一般介面搜尋結果(範例二)

III. 案例三－Google Page Rank 問題

延續上一個案例二範例圖 19，其中有一句”分享著他們的喜樂。”，將之至一般介面搜尋，檢視其結果圖 21，發現到此句的相關結果，在Google搜尋的Page Rank並非在前四筆，而是在第六筆，導致系統無法正確抓取到相關結果。

其可能因素為，用字太過一般化，是經常會使用字詞或者是特定的主題經常會使用到這類型的字句，如基督教，常用到喜樂、阿門…等等。又加上系統是利用Google做第一階段的相似度篩選，並僅傳回前四筆資料做第二階段的相似度比對，若其相關結果在第一階段的篩選中並未正確傳回，導致系統無法比對出後續的結果。

圖 21：Google一般介面搜尋結果(範例三)

IV. 案例四－修正後假警報情況

本系統採用ROUGE-L及修正後LCS公式的偵測結果中，尚有一些假警報情況，在此以案例的方式說明為何會發生此狀況。

例如：圖 22 為系統上傳”等到送醫後才知道，”之句子傳回的結果，其最長共同子序列為“等到後才知道”，本研究修正LCS公式，會取”等到你生病後才知道”之長度當成分母，得到相似度0.67；而ROUGE-L考慮兩句的長度得到相似度分數也為0.67。但事實上，此句並非剽竊的句子，雖然兩種公式都已可以降低其相似度，但由於關鍵字與關鍵字之間距離不遠，其降低的程度不夠，而導致此種假警報情況，不過其最根本的原因還是由於候選文句的句子長度不夠長，且其使用的字詞是較為常見的。

某一些情形下，修正後LCS公式會出現假警報的狀態，但ROUGE-L不會，如圖 23 所示，系統上傳”經由政府的倡導”之句子傳回的結果，其最長共同子序列為“經由政府倡導”，本研究修正LCS公式，會取”經由台中市政府於倡導”之長度當成分母，得到相似度0.6；而ROUGE-L計算的相似度卻只有0.29，ROUGE-L修正的幅度比修正LCS公式高，解釋了表 16修正LCS公式在門檻值0.5時假警報機率比ROUGE-L高。

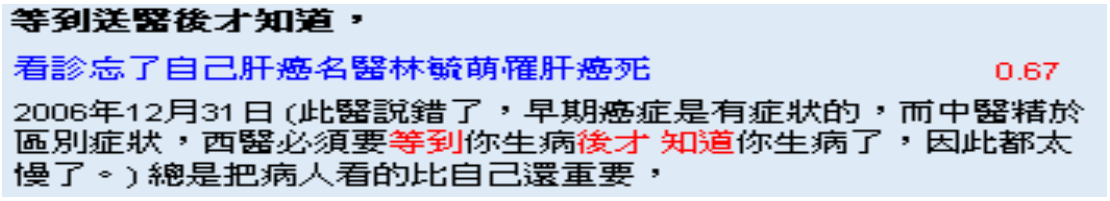

圖 22：案例四範例一

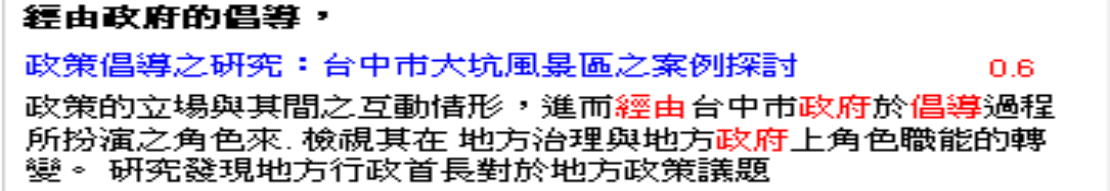

圖 23：案例四範例二

第五章　結論與建議

5.1 結論

本研究運用Google AJAX Search API建置一中文剽竊偵測系統，將Google搜尋引擎機制當成第一階段的相似度篩選，自動化偵測使用者上傳之文件是否有抄襲自網路的嫌疑，並以視覺化系統介面呈現給使用者，利於人工檢視系統偵測結果。

本研究首先會將使用者上傳之文章，經CKIP斷成句子後，一句一句傳入Google AJAX Search API，分析傳回之JSON編碼結果，取其摘要資訊做第二階段之LCS相似度比對，若高於本研究所設定的門檻值，即顯示於系統上，做後續的人工檢視。透過實驗分析比較修正前後的相似度比對公式，本研究的結果整理如下：

一、　本研究利用Google搜尋引擎與修正前後的最長共同子序列公式來偵測文件是否有剽竊之嫌疑，透過實驗結果所推薦出系統最適門檻值0.6或下，其系統正確率達八成以上，而且修正後的最長共同子序列相似度比對公式，可大幅降低系統假警報之機率。

二、　本研究將實驗的結果做討論與分析，發現幾個可能影響系統偵測正確率之因素，包含文章含有英文字，Google Page Rank、編碼問題，這是後續可改善的方向。

剽竊偵測系統的主要目的是去嚇阻進行剽竊行為的惡意人士，使文件或文章作者得到應有的智慧財產權保護；還有讓學生以自我檢視的方式，了解到怎樣的寫作方式可能會觸犯他人的權益，提高自我的警覺心。雖然透過不斷改進的剽竊偵測系統，如商業化 Turnitin[20]、Docol© c[7]，其偵測績效愈來愈佳，甚至在2009 九月繁體中文版本的剽竊系統引進台灣，進行販賣，許多大學也著手準備將其引進大學教育中，但僅有監測並無法完全杜絕剽竊行為的產生，而且其偵測

的意義也值得我們深思，唯有加強下一代的剽竊觀念教育、智慧財產權的宣導，才能達到預防之效，一味的偵測然後處罰並無太大的意義。本研究期望能透過系統的實作，賦有教育意義地讓使用者知曉不得隨意使用網路上的資訊，此舉可能會觸犯他人的智慧財產權，應正確引用他人想法、言詞。

5.2 未來改進方向

本研究藉由剽竊系統的實作發現某些文章的偵測效率不佳，經過案例的討論與分析，深入了解後，若要加強擴充本研究，尚有許多地方可再補強，以下針對後續應改善與發展方向做說明：

I. 資料庫的擴充

經研究結果知道，剽竊偵測必須依賴後端文件集(Corpus)的收集，若能收集越多的文章當成比對的基礎，其系統的偵測效率就會越佳，也更具可信度，但由於本研究的文件集僅限於Google搜尋引擎所收集的文章，並沒有自身的資料庫，其中不包含一些實體書籍、碩博士論文、中文期刊等文集的資料庫，導致從這些文集剽竊而來的字句無法被系統偵測出來，為了讓系統更具實用性與可信度，未來可增加更多Corpus讓系統更具其正確性。不過這些資料的收集，又會牽涉到文章著作權的問題，有些作家並不願意提供其作品，或者是使用著作時要付出多少代價難以商議。

II. 編碼問題

中文字存在中文編碼標準的問題，目前光台灣使用的中文編碼就有數種，包含Big5、ISO 10646、UTF-8、ASCII…等等，每個編碼標準所概括的中文字數不同，編碼方式也不相同， 且沒有統一的國家標準規格，規範資料在各式各樣的程式介面之間應如何被轉換，導致在轉換跟分析間，會造成困擾，使得系統無法正確去運作。未

來可透過Sun[18]等人所提出的方式，基於中文字本身之結構，將其中文字的位置關係轉換成運算子、字的結構轉換成運算元的數學表示式，來減少分析中文字編碼時的錯誤，增加系統的偵測正確率。

III. Google搜尋引擎參數調整

利用Google搜尋引擎查詢的特性，將使用者上傳的文章經CKIP斷成的句子加上雙引號，傳入Google搜尋引擎，傳回精確搜尋的結果，雖無法傳回部份相似的相關結果，但可改善部份相似時所造成的假警報的問題。

本研究僅取Google搜尋引擎傳回的前四筆結果，未來可增加取得的筆數，改善Google Page Rank造成的問題。

參考資料

[1] Apache POI, http://poi.apache.org/.

[2] Broder, A. Z., Glassman, S. C., Manasse, M. S., and Zweig, G. (1997). Syntactic Clustering of the Web. *Computer Networks and ISDN Systems*, vol. 29, no. 8, 1157 – 1166.

[3] Brin, S., Davis, J., and Garcia-Molina, H. (1995). Copy Detection Mechanisms for Digital Documents. *ACM SIGMOD Record*, vol. 24, no. 2, 398 – 409.

[4] Chowdhury, A., Frieder, O., Grossman, D., and McCabe, M. C. (2002). Collection Statistics for Fast Duplicate Document Detection. *ACM Transactions on Information Systems*, vol. 20, no. 2, 171 – 191.

[5] CNN.com, http://edition.cnn.com/.

[6] Cormen, T. H., Leiserson C .E., and Rivest R. L. (1989) Introduction to Algorithms. The MIT Press.

[7] Docol© c,http://www.docoloc.de/.

[8] Dierderich, J. (2006). Computational Methods to Detect Plagiarism in Assessment. *Information Technology Based Higher Education and Training*, pp. 147–154. Sydney, Australia.

[9] Google AJAX Search API, http://code.google.com/intl/en/apis/ajax/.

[10] Lin, C.-Y. (2004) ROUGE: A Package for Automatic Evaluation of Summaries. *In Proceedings of Workshop on Text Summarization Branches Out, Post-Conference Workshop of ACL 2004*, pp. 74-81. Barcelona, Spain.

[11] Manber, U. (1994) Finding Similar Files in a Large File System. *In Proceedings of the USENIX Winter 1994 Technical Conference*, pp. 2-2. San Francisco, California.

[12] McCuen,R.H. (2008)The Plagiarism Decision Process:The Role of Pressure and

Rationalization. *IEEE Transactions on Education*, vol. 51, no. 2, 152–156.

[13] Papineni, K., Roukos, S., Ward, T., and Zhu, W. -J. (2002) BLEU: A Method for Automatic Evaluation of Machine Translation. *In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 311--318. Philadephia, USA.

[14] Plagiarism.org, http://www.plagiarism.org/

[15] Rabin, M. O. (1981) Fingerprinting by Random Polynomials. *Center for Research in Computing Technology*, Harvard University, Report TR-15-81.

[16] Shivakumar, N., and Garcia-Molina, H. (1995) SCAM: A copy detection mechanism for digital documents. *In Proceedings of the Second International Conference in Theory and Practice of Digital Libraries*, Austin, Texas.

[17] Stein, B., and Meyer Zu Eissen, S. (2006) Near Similarity Search and Plagiarism Analysis. *Data and Information Analysis to Knowledge Engineering*, vol. 10, 430 –437.

[18] Sun, X. M., Chen, H. W., Yang, L. H., and Tang, Y. Y. (2002)Mathematical Representation of a Chinese Character and its Applications. *International Journal of Pattern Recognition and Artificial Intelligence*, pp.735--747.

[19] Stepchyshyn, Vera, and Nelson, Robert S. (2007) Library plagiarism policies. *Association of College and Research Libraries*, p. 65.

[20] TurnItIn, http://www.turnitin.com/.

[21] 中學生網站， http://www.shs.edu.tw/essay/。

[22] 中文斷詞系統，http://ckipsvr.iis.sinica.edu.tw/。

[23] 王偉全，“文件抄襲偵測”，元智大學資訊管理研究所，碩士論文，2006年。

[24] 陳建穎，”以ROUGE和WordNet為基礎的N-gram共現於剽竊偵測”，國立交通大學資訊管理研究所，碩士論文，2009年。

[25] 資策會，2009年12月底止台灣上網人口，

http://www.find.org.tw/find/home.aspx?page=many&id=219。

[26] 劉奕廷，“以搜尋引擎進行剽竊模式之評估”，國立成功大學工程科學研究所，碩士論文，2007年。