利用語法結構及語意相似度建立改寫句子抄襲偵測方法

蘇嘉穎
國立成功大學資訊管理研究所
soukaweng@hotmail.com

王惠嘉
國立成功大學資訊管理研究所
hcwang@mail.ncku.edu.tw

摘要

目前學術研究發表愈來愈受到重視，但隨著網際網路的進步，資訊取得越來越容易，導致學者有意或無意地抄襲他人想法或作品。事實上，有許多學者對於抄襲的認知不正確，因此需要提供一個可主動協助或教育學者偵測所著文件是否抄襲的環境。但目前市上的抄襲偵測軟體只能偵測簡單的抄襲類型，且無法引導學者適切地改寫句子。

本研究將提出一能偵測改寫行為的改寫句子抄襲偵測方法，利用語法結構分析並計算句子語意相似度，找出可能有抄襲情形的修改文件，分析並自動判斷該文件中改寫的部分是否可能為抄襲。期望藉由系統告知使用者已使用哪些改寫手法，讓使用者了解其改寫方式是否觸犯抄襲。

關鍵字：改寫、抄襲偵測、語法結構分析、語意相似度

壹、 緒論(Introduction)

為全面提升台灣各大學在全球高等教育的競爭力，教育部委託社團法人台灣評鑑協會，對全國一般大學院校進行校務評鑑，透過質性與量化的指標，衡量各大學之國際化程度，重要指標項目如：學生英文檢定程度、全英語授課課程、發表於國際期刊論文數等(教育部高教司,民 93)。由此可見，英語能力已成為進入全球化時代的必要關鍵之一，因此各大專院校也掀起提升英語能力與推動國際化的潮流，並以各式學術獎勵辦法鼓勵師生多加進行研究，而英語自然成為研究成果發表的主要媒介。

然而，台灣的慣用語並非英語，可能尚有許多學者的英語寫作能力並不足以獨自撰寫出全英語的學術研究成果。Sun(2009)的研究結果顯示，即使台灣學者在學術文件寫作上有豐富的經驗，但他們在改寫以及引用手法的訓練上仍舊不夠純熟，這也造成了部分學者會尋找與自己研究成果相關的資訊，進行改寫或抄襲的動作。

隨著網際網路的進步，數以萬計的文字、程式碼、圖片、音樂、影片以及任何可以想像到的電子資訊，在網路上已越來越容易取得，換言之，電子資訊被濫用的可能性也大大提高，這種情形已讓抄襲行為達到空前的規模(Potthast, Barron-Cedeno, Stein, & Rosso, 2011)。隨著資訊科技與網際網路的日漸發達，紙本資料慢慢建置成電子檔，數位出版與電子文件也成為資訊散播的重要方式之一，這些現象的轉變造成了資訊的取得與傳播變得越來越容易，尤其藉由各種搜尋引擎強大的搜尋功能，使用者所需要的資料大多皆能夠輕易取得。

除了網路的發達使得資料取得越來越容易，文字處理工具的迅速與便利也是造成學者抄襲風氣盛行的原因之一，許多學者往往在撰寫研究成果時，可以直接將所需要的資訊貼上且不註明來源(Oetsch, Puehrer, Schwengerer, & Tompits, 2010)，或是只有稍加修改就將別人的著作納為自己的研究，此種缺乏尊重他人智慧財產權的侵權行為隨著網路的普及而日漸嚴重(Pera & Ng, 2011; Zaka, 2009)。

目前因為教育與社會的進步，保護智慧財產權的觀念已逐漸被人們所重視，若是學者的研究成果來自於抄襲他人的創作，便是顯示本身對智慧財產權的不尊重。然而，學者除了有意竊取他人想法或作品，可能因為學術寫作訓練和基本寫作能力不足，不知道如何適當地引用和改寫資料，甚至不清楚自己正在抄襲，而非有意遮掩引用的資料來源(Pecorari, 2003)。

由於抄襲與非抄襲的劃分邊界模糊，且許多人對於抄襲的認知不正確，認為有對內容稍作修改就不是抄襲，造成許多抄襲者不知道自己已構成抄襲行為。目前市面上鮮少有軟體能教導使用者如何避免英文寫作抄襲，並且清楚告知正確的反抄襲概念。在英文寫作技巧上，有效的反抄襲方法可以教導學者如何適切地改寫句子或文章(Pecorari, 2003; Walker, 2008; Yamada, 2003)。因此，為有效保護他人的創作，除了被動地以加密機制禁止複製及修改文件外，更需主動地協助或教育學者偵測所著文件是否抄襲(Kang,

Gelbukh, & Han, 2006)。

然而，在學者人數眾多且需要比對的資料量龐大的情況下，以人工的方式判斷文件是否抄襲將耗費龐大的時間與人力，因此目前已有許多以電腦技術協助偵測抄襲的系統，例如 Turnitin(iParadigms, 2011)，為普遍使用的線上抄襲偵測系統，提供網路資源及內部資料庫文件的比對服務，可比對出學者抄襲的比重，並以色塊標記出和抄襲原文相同之處，但 Turnitin 只有做逐字或複製貼上的抄襲偵測，並沒有提供改寫偵測的功能(Maurer, Kappe, & Zaka, 2006; Uzuner, Katz, & Nahnsen, 2005)。

Mozgovoy, Kakkonen, & Cosma(2010)表示，現有抄襲偵測系統的不足最根本的原因在於它們都高度依賴於非自然語言的處理方法，只有單純使用字串比對方法，但這樣的方式只能偵測出簡單的抄襲類型，無法偵測出是否有同義字替換的現象，因此，如何偵測字詞重新排序和文句改寫仍然是抄襲偵測系統的一大挑戰。為此，本研究將提出一個能偵測出改寫行為的改寫句子抄襲偵測方法，並以片語作為偵測比對單位，同時考量同義字替代及字詞次序的改變，嘗試找出可能有抄襲情形的改寫文章或句子，詳細分析並自動判斷該文章或句子是否可能為抄襲，期望藉由系統的實際運作，清楚告知學者已使用哪些改寫的手法，藉此讓學者了解自身的改寫是否有觸犯抄襲行為。

本研究的研究範圍與限制有以下幾點：

一、 抄襲偵測分成外顯抄襲偵測和內隱抄襲偵測兩種(Potthast, Stein, Eiselt, Barrón-Cedeño, & Rosso, 2009)，外顯抄襲偵測須先給予一份或多份來源文件(Source documents)，與修改文件進行比對，偵測出修改文件對應的來源文件；而內隱抄襲偵測則只需給予修改文件做抄襲識別。本研究主要處理外顯抄襲偵測，因此需要給予來源文件，偵測修改文件是否可能為抄襲。

二、 本研究主要針對句子做抄襲比對偵測，因此不考慮文件內文字以外之資訊，例如：圖片、連結等。

三、 在句子改寫手法分析方面，本研究將進一步分析句子的改寫手法以供使用者參考，但僅針對 Walker(2008)所定義的五種拼湊寫作行為進行分析。

貳、 文獻探討

為了避免抄襲，學者往往會將文章內容稍作改寫，但要偵測出這種抄襲行為，必須要有可以偵測出此類改寫手法的技術。本章節將針對研究中的相關領域進行文獻回顧，以了解目前從事抄襲偵測相關之研究發展狀況、技術方法以及應用範圍。以下主要先對本研究所著重的抄襲列出幾種常見的抄襲手法，接著將介紹本研究採用的自然語言處理技術，以及語意相似度計算方式，最後對現有的偵測抄襲方法及系統作詳細的介紹。

一、 英文抄襲的行為

Howard(2001)將常見的抄襲手法分為四種：第一種是欺騙(fraud)，指遞交他人整篇作品作為自己的原著作；第二種是拼湊寫作(patchwriting)，指使用他人的片段文字，再部分改寫組合成新句子；第三種為無法正確的引用(failure to cite sources)，指內文引用的部份缺乏標註資料出處來源；第四種為無法正確的直接引用(failure to quote)，指內文直

接使用來源文字而未標註雙引號。

然而，不適當的改寫文句即被視為拼湊寫作，寫作者將不自覺地落入抄襲的陷阱。Walker (2008)定義拼湊寫作涵蓋從原文中直接抄寫 5 至 9 個字(word string)、同義字的替換(substitution)、從原文中增加 1 至 4 個字(addition)、從原文中減少 1 至 4 個字(deletion)，以及倒裝置換文句架構(reversal)。因此，本研究將針對上述 5 種改寫手法，詳細分析改寫後的文章是否有觸犯拼湊寫作行為，以判斷該文章是否可能為抄襲。

二、 自然語言處理

自然語言處理(Natural Language Processing, NLP)是研究如何讓電腦能夠了解及運用人類的語言來處理問題，屬於人工智慧及語言學領域的分支學科(Losee, 2001)。NLP 的範疇相當廣泛，以下將針對本研究主要會用到的三項技術作簡要的介紹：

表 1 自然語言處理技術

自然語言處理技術	**簡介**
詞性標註	對一個句子中的每個字詞進行詞性的判斷，並加以標註。
語法分析	分析句子的文法規則及架構，並以樹狀結構的模式顯示。
字根還原	由於同樣的字會因單複數、主被動、時態及詞性變化而有所不同，為了避免誤判，便需要將字詞還原成字根。

三、 語意相似度

（一） 英文語意相似工具 - WordNet

WordNet 是由普林斯頓大學的認知研究室所開發的英語辭彙的資料庫。在 WordNet 中的各種詞性的字詞，如名詞(Nouns)、動詞(Verbs)、形容詞(Adjectives)及副詞(Adverbs)，會根據其意義組織成一個同義詞集合(Synonym Sets, Synset)，每個同義詞集合(Synset)都代表一個基本的語意概念(Concept)，而這些集合之間會由各種不同的關係，包括：上義詞(Hypernym)、下義詞(Hyponym)、同義詞(Synonym)、反義詞(Antonym)等，因此一個具有多種語意(Sense)的字詞可能會出現在不同的同義詞集合中。

在 WordNet 中，名詞和動詞會以上下義詞關係組成階層式的架構，然而，形容詞和副詞並沒有上義詞或下義詞的關係，故無法以階層式架構呈現，而四種不同詞性的網路之間並無連接。因此在分析兩個字詞之間的語意相似度時，形容詞和副詞便不能夠使用上下義詞的關係來分析。

（二） 字詞語意相似度計算

要分析兩個字詞之間是否為語意相關，可以透過 WordNet 中的字詞關係來分析。Oliva, Ignacio Serrano, Dolores del Castillo, & Iglesias(2011)歸納出字詞間語意相似度有三種衡量方式：

表 2 字詞語意相似度衡量方式

方式	**描述**	**常見方法**
Path-based measures	利用 WordNet 內字詞的階層架構，分析字詞之間語意相關的程度。	PATH(Rada et al., 1989) WUP(Wu & Palmer, 1994)
Information content measures	透過字詞的個別資訊含量(Information content)，計算出字詞之間的語意相似度。	RES(Resnik, 1995) JCN(Jiang & Conrath, 1997) LIN(Lin, 1997)
Gloss-based	利用字詞在 WordNet 中註解的資訊，計	VECTOR(Patwardhan, 2003)

measures	算字詞間語意相關的程度。	

本研究參考 Li, McLean, Bandar, O'Shea, & Crockett(2006)的做法，使用 Path-based measures 計算名詞和動詞的相似度，同時考量字詞間的最短路徑長度和深度，計算字詞之間的語意相似度。然而，要計算形容詞和副詞的語意相似度，Gloss-based measures 是唯一的方法(Oliva et al., 2011)，因此針對形容詞和副詞，本研究採用 VECTOR 方法計算其語意相似度。

四、 抄襲偵測方法

抄襲偵測方法可分成三大類(Chen, Yeh, & Ke, 2010)，如圖 1 所示：

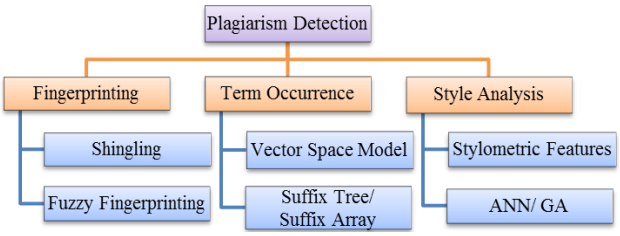

圖 1 抄襲偵測方法的類型(Chen, Yeh, & Ke, 2010)

表 3 抄襲偵測方法類型

方法	**描述**
Fingerprinting	將文件中的字串轉換成獨一無二的數字(Fingerprint)，使用多項式演算法(Rabin, 1981)或雜湊函數計算文件的 Fingerprint，透過比較 Fingerprint 的相似度決定兩份文件是否為抄襲。
Term Occurrence	為最直覺的方法，基於文件相似度做抄襲偵測，兩份文件共同出現的字數越多，它們就越相似。
Style Analysis	跟其他方法不一樣，此方法較著重於內隱的資訊，基本原理是每個寫作者都有他自己的寫作風格，因此在文章內風格應一致。

五、 現有的抄襲偵測系統

截至目前為止，現有的抄襲偵測系統非常多種，介面多數為英文，目前較為常見的有：CopyCatch(CFL Software Limited, 2011)、EVE2(Canexus Inc., 2011)、Turnitin(iParadigms, 2011)等，而繁體中文的部分，尚有中山大學開發的中文數位反抄襲偵測比對系統、PPvS 論文抄襲檢查網等，其中只有 Copycatch 為免費使用，其餘則須付費。但這些現有的抄襲偵測系統尚有些許問題：

（一） Turnitin

系統無法處理少於 20 字的短句，一旦上傳少於 20 字的文字檔，則系統會提醒無法作業，且 Turnitin 只能偵測出完全符合的文句，部分文章可能會利用同義字替換，如此一來 Turnitin 便無法偵測出抄襲。

（二） CopyCatch

使用者將文件以 word 檔的形式上傳，系統比對內部資料庫後，標記可能抄襲的文句，再以表格方式呈現比對結果。CopyCatch 使用電子郵件的方式通知使用者文件比對後的結果，但這總共需要約兩天的檢核時間(Culwin & Lancaster, 2000)。

（三） EVE2

使用者需將軟體下載至個人電腦使用，無法直接在線上操作系統。系統比對的資料來源

為網路資源，並沒有內部資料庫，也不會把兩份文件作比對，而且只有 copy-paste 的偵測功能(Bull, Colins, Coughlin, & Sharp, 2001)。

參、 研究方法

為了改善抄襲偵測的準確性，並偵測出句子的改寫行為，本研究提出一套利用語法結構與語意相似度的改寫句子抄襲偵測方法。透過句子的語法結構可以找出句子中所有片語，接著以片語為比對單位進行語意相似度比較，並考慮其次序的改變，計算句子相似度以偵測出可能抄襲的句子。以下章節將對本研究的架構與流程做詳細的說明。

一、 研究架構

圖 2 為本研究的系統架構圖，主要分成三個區塊，分別為語法處理階段(Syntactical Processing Phase)、片語識別階段(Phrase Recognition Phase)、語意比對階段(Semantic Matching Phase)和改寫手法分析階段(Paraphrased Analysis Phase)。整個系統流程如下：

（一） 語法處理階段(Syntactical Processing Phase)

此階段任務為將蒐集到的來源文件集(Source Documents)以及使用者修改文件(User Modified Document)進行句子切割，透過語法結構分析擷取句子裡的片語，並開始進行詞性標註和字根還原等前處理(Pre-processing)的動作。

（二） 片語識別階段(Phrase Recognition Phase)

本研究主要以片語作為抄襲偵測比對的單位，但從上一階段擷取出來的片語在抄襲比對上並不一定具有意義，故需要先透過句子的比對以及機器可讀字典的查詢以篩選出符合資格的片語。不符合資格的片語則需要進行拆解，而拆解後的片語又會重新回到篩選的動作，如此不斷循環，直到所有拆解出來的片語都已經過篩選或被拆成單字為止。

（三） 語意比對階段(Semantic Matching Phase)

此階段任務為以片語為單位進行語意相似度的比較，並考慮句子中片語和字詞次序的改變，以計算句子的相似度。相似度高於門檻值的句子，會進一步分析該句子的改寫手法，判斷句子是否可能為抄襲。

（四） 改寫手法分析階段(Paraphrased Analysis Phase)

對於被視為抄襲的成對句子，利用 Walker (2008)所定義的五種併湊寫作手法進行分析，清楚告知使用者其改寫手法是否觸犯抄襲。

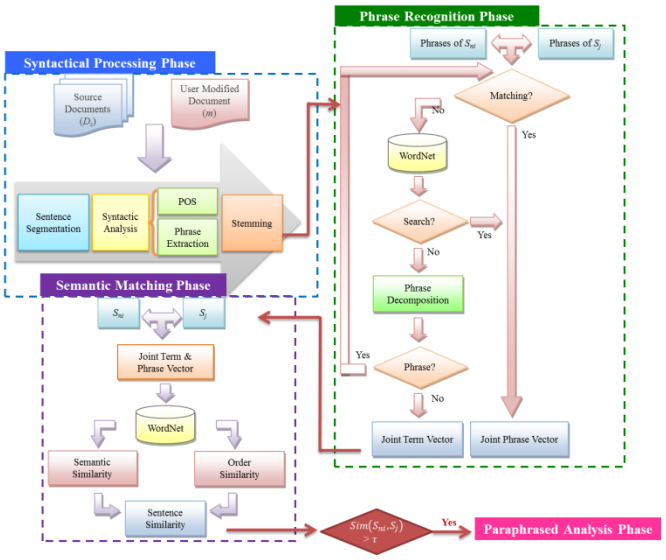

圖 2 本研究架構圖

二、 語法處理階段

我們先將所有蒐集到的來源文件集以及使用者修改文件進行斷句，並對每個句子做語法分析，以擷取句子的字詞和片語。接著進行前處理的動作，包含詞性標註、字根還原等。整個流程如圖 3 所示。

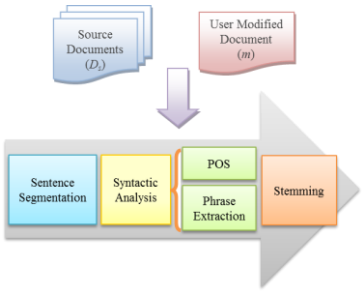

圖 3 語法處理階段流程圖

（一） 斷句

首先，我們先準備一個使用者參考之來源文件集合(Source Documents, D_S) ，以及一份使用者修改文件(User Modified Document, m)，$D_s = \{d_1, d_2, \cdots, d_n, \cdots\}$，其中$d_n$為第$n$份來源文件。由於每份文件是由一個或以上的句子(Sentences, S)所組成，因此在做語法結構分析以前，必須先把文件切割成獨立的句子，在此把上述之文件定義如下：

$d_n = \{S_{n1}, S_{n2}, \cdots, S_{ni}, \cdots\}$	$m = \{S_1, S_2, \cdots, S_j, \cdots\}$

其中S_{ni}為第n份來源文件的第i句； S_j為使用者修改文件的第j句。

（二） 語法分析

1. 詞性標註

把每份文件裡已經切割完成的句子S_{ni}和S_j，使用 Stanford Parser 進行句子的詞性標註及語法分析步驟。首先得到每個字詞的詞性，每個句子會由許多字詞(Terms, t)所組成，定義如下：

$S_{ni} = \{t_{ni1}, t_{ni2}, \cdots, t_{nik}, \cdots\}$	$S_j = \{t_{j1}, t_{j2}, \cdots, t_{jl}, \cdots\}$

其中t_{nik}為第n份來源文件第i句中第k個字詞；t_{jl}為使用者修改文件第j句中第l個字詞。

2. 片語擷取

Uzuner et al.(2005)表示，以逐字為基礎來計算相似度的抄襲偵測方式，對於改寫的文章並不適用，因為同義字的替換會把相似度降低，而無意義字詞上的重疊卻會錯誤提高相似度，故本研究將使用片語作為抄襲偵測之比對單位。因此，我們把上述的結果S_{ni}及S_j擷取出所有的片語，步驟如下所示：

(1) 擷取 Stanford Parser 樹狀結構(Parse)中連續之片語(Noun Phrase, NP、Verb Phrase, VP)，在此定義為$tree_p$：

$tree\_p_{ni} = \{tree\_p_{niab}, tree\_p_{niab}, \cdots\}$ $tree\_p_{niab} = \{t_{nia} \cdots t_{nib}\}, 1 \le a < b$	$tree\_p_j = \{tree\_p_{jab}, tree\_p_{jab}, \cdots\}$ $tree\_p_{jab} = \{t_{ja} \cdots t_{jb}\}, 1 \le a < b$

其中$tree_p_{ni}$為由第n份來源文件中第i句中，從樹狀結構擷取的片語集合，而$tree_p_{niab}$代表第n份來源文件中第i句中從第 a 個字詞到第 b 個字詞所組成的片語；$tree_p_{ni}$為由使用者修改文件中第j句中，從樹狀結構擷取的片語集合，而$tree_p_{jab}$代表使用者修改文件中第j句中從第 a 個字詞到第 b 個字詞所組成的片語。

(2) 因為及物動詞片語在句子中可能會被分開，如“John switched the radio on.”中，受詞“the radio”位於動詞片語“switched on”之間，導致在樹狀結構中無法擷取，因此，我們需要利用 Stanford Parser 產生相依關係(Typed dependencies)，並將 WordNet 作為輔助工具，尋找可能潛在的片語，在此定義為dep_p：

$dep\_p_{ni} = \{dep\_p_{nicd}, dep\_p_{nicd}, \cdots\}$ $dep\_p_{nicd} = \{t_{nic}, t_{nid}\}, 1 \le c < d$	$dep\_p_j = \{dep\_p_{jcd}, dep\_p_{jcd}, \cdots\}$ $dep\_p_{jcd} = \{t_{jc}, t_{jd}\}, 1 \le c < d$

其中dep_p_{ni}為由第n份來源文件中第i句中，從相依關係擷取的片語集合，而dep_p_{nicd}代表第n份來源文件中第i句中第 c 個字詞和第 d 個字詞所組成的片語；dep_p_j為由使用者修改文件中第j句中，從相依關係擷取的片語集合，而dep_p_{jcd}代表使用者修改文件中第j句中第 c 個字詞和第 d 個字詞所組成的片語。

(3) 我們把$tree_p$和dep_p組成一個片語集合P，把剩餘的單一字詞組成一個字詞集合T，目的是將句子中單一字詞和片語區分開來，區分結果如下：

$S_{ni} = \{t_{ni1}, t_{ni2}, \cdots, t_{nik}, \cdots\} = \{T_{ni}, P_{ni}\}$ $tree\_p_{niab} \in tree\_p_{ni} \in P_{ni}$ $dep\_p_{nicd} \in dep\_p_{ni} \in T_{ni}$ If $t_{nik} \notin tree\_p_{niab}$ & $t_{nik} \notin dep\_p_{nicd}$ then $t_{nik} \in T_{ni}$	$S_j = \{t_{j1}, t_{j2}, \cdots, t_{jl}, \cdots\} = \{T_j, P_j\}$ $tree\_p_{jab} \in tree\_p_j \in P_j$ $dep\_p_{jcd} \in dep\_p_j \in T_j$ If $t_{jl} \notin tree\_p_{jab}$ & $t_{jl} \notin dep\_p_{jcd}$ then $t_{jl} \in T_j$

其中T_{ni}為第n份來源文件中第i句的字詞集合，P_{ni}則為該句的片語集合，T_j為使用者修改文件中第j句的字詞集合，P_j則為該句的片語集合。

（三） 字根還原

語法處理階段的最後步驟為字根還原，將之前擷取出來的字詞集合T_{ni}, T_j、片語集合P_{ni}, P_j內的所有字詞，利用 WordNet 進行字根還原，避免同一字詞因單複數或時態的不同而造成誤判。

三、 片語識別階段

本研究主要以片語作為比對的單位，在上一階段我們已經把句子中所有的片語擷取出來，分別是$tree_p$和dep_p。dep_p裡面的片語其實已經過 WordNet 的過濾，但$tree_p$只是單純的從樹狀結構擷取，對於抄襲比對而言可能並沒有意義，故本階段的目的是透

過來源文件和使用者修改文件內句子的比對，篩選出具有意義之片語，最後將兩句子之字詞集合和片語集合聯合起來，產生 Joint Term Vector 和 Joint Phrase Vector，以供後續階段使用。整個流程如圖 4 所示。

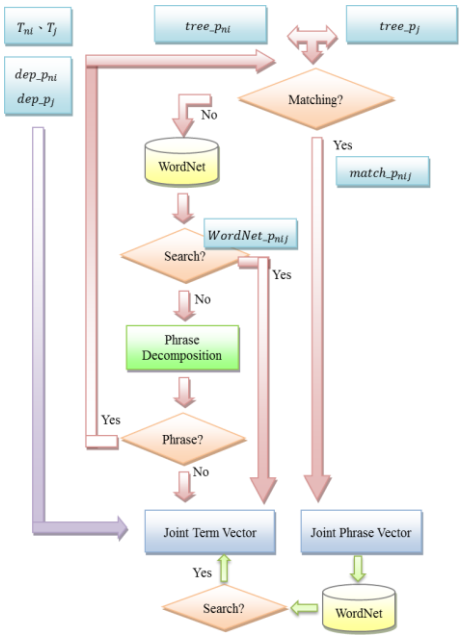

圖 4 片語識別階段流程圖

片語識別詳細步驟如下：

（一） T、dep_p可直接放入 Joint Term vector 中。

（二） $tree_p$則需要通過識別，我們把$tree_p_{ni}$跟$tree_p_j$一一進行比對，若配對成功，則會將此$tree_p$紀錄為$match_p_{nij}$，並放入 Joint Phrase vector 中；但若沒有配對成功，則會放入 WordNet 做查詢。

（三） 若在 WordNet 中確有此字，則會將此$tree_p$紀錄為$WordNet_p_{nij}$放入 Joint Term vector；反之，若 WordNet 中查無此字，則將此$tree_p$進行拆解。

（四） 片語拆解時，若有部分配對的單一字詞或連續字詞，則可直接拆解出來；但若沒有部分配對，則需透過 Stanford Parser 的相依關係，找出潛在之片語；

（五） 將經過拆解的字詞作分析，若為單一字詞，則放入 Joint Term Vector；反之，若仍為片語，則必須再以此片語進行來源文件與使用者修改文件比對，並重複上述步驟 2, 3, 4，直到所有字詞都已放入兩個 Vector 為止。

（六） 最後把 Joint Phrase vector 中的片語放入 WordNet 查詢，若可以在 WordNet 中查詢到，則將此片語視為單一字詞，放入 Joint Term Vector 中。

四、 語意比對階段

在語意比對階段，主要以片語為單位進行語意相似度的比較，並把句子次序相似度加入考量，以計算兩句子的相似度。但在計算句子相似度前，必須先定義字詞與片語之語意相似度、句子語意相似度以及句子次序相似度的計算方式，整個流程如圖 5 所示：

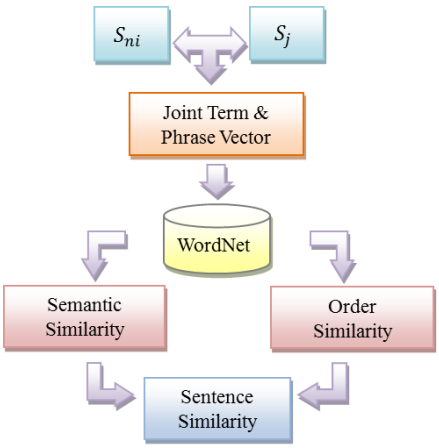

圖 5 語意比對階段流程圖

（一） 字詞、片語之語意相似度

因為本研究使用片語為抄襲比對的單位，所以在進行句子語意相似度計算時，會遇到 3 種比對情況：字詞與字詞的比對、字詞與片語的比對，以及片語與片語的比對。經過片語識別階段後，因為dep_p和$WordNet_p$皆已經過 WordNet 的查詢，所以可以視為單一字詞，字詞已被放入 Joint Term Vector，片語已被放入 Joint Phrase Vector。

表 4 為所有的比對狀況，狀況 A 是 2 個T做比對、狀況 B 為 1 個T和 1 個dep_p或$WordNet_p$做比對，以此類推：

表 4 字詞、片語所有可能比對狀況

		Joint Term Vector		Joint Phrase Vector
	狀況		dep_p或$WordNet_p$	$match\_p$
字詞與字詞	**A**	2	0	0
	B	1	1	0
	C	0	2	0
字詞與片語	**D**	1	0	1
	E	0	1	1
片語與片語	**F**	0	0	2

以下將針對每一種比對狀況，詳細說明其語意相似度計算方式，以作為後續進行句子語意相似度計算的基礎。

1. 字詞與字詞之語意相似度**(狀況 A、狀況 B、狀況 C)**

假設$w_1, w_2 \in T \in$ Joint Term Vector，而在 WordNet 中，每一個字詞都包含一種或以上的語意(Sense)，可把字詞表示為 $w_1 = \{sense_1, sense_2, \cdots, sense_X, \cdots\}$、$w_2 = \{sense_1, sense_2, \cdots, sense_Y, \cdots\}$。

因為在 WordNet 中，名詞和動詞會以上下義詞關係組成階層式架構，然而，形容詞和副詞則無法以階層式架構呈現，因此，我們在計算字詞之間的語意相似度時，須先判斷兩個字詞的詞性，並採用不同的相似度計算方法。如果兩字詞為名詞或動詞，將使用 Path-based measure 計算相似度；但如果兩字詞為形容詞或副詞，則只能使用 Gloss-based measures 來計算。

本研究將以 WordNet 中名詞和動詞的階層架構為基礎，同時考量字詞間最短路徑的長度和深度，結合 PATH(Rada et al., 1989)和 WUP(Wu & Palmer, 1994)的方法來計算名詞和動詞的語意相似度，而形容詞和副詞則使用 VECTOR 方法來計算語意相似度。方法詳細內容及公式如下：

(1) PATH 方法使用兩字詞在 WordNet 中最短路徑長度來計算，如公式 3-1 所示：

$$PATH\big((w_1, sense_X), (w_2, sense_Y)\big) = \frac{1}{path\_length((w_1, sense_X), (w_2, sense_Y))} \quad (1)$$

(2) WUP 方法則是考慮了兩字詞的深度，以及它們的最小共通父節點(LCS)在 WordNet 中的深度，來計算字詞相似度，如公式 3-2 所示：

$$WUP\big((w_1, sense_X), (w_2, sense_Y)\big) = \frac{2 \times depth(\mathrm{LCS})}{depth(w_1, sense_X) + depth(w_2, sense_Y))} \quad (2)$$

(3) VECTOR 方法利用兩字詞在 WordNet 中的註解組成向量$\overrightarrow{v_1}$、$\overrightarrow{v_2}$，再使用 Cosine 計算字詞的相似度，如公式 3-3 所示：

$$VECTOR\big((w_1, sense_X), (w_2, sense_Y)\big) = \frac{\overrightarrow{v_1} \cdot \overrightarrow{v_2}}{|v_1||v_2|} \quad (3)$$

Li et al.(2006)認為單純使用 PATH 方法來衡量字詞之間的語意相似度，準確度可能會很低，因為在 WordNet 中，越上層的節點越廣義，而越下層的節點越狹義。因此，在計算字詞相似度時，字詞的深度(Depth)也該納入考量，故本研究把 PATH 和 WUP 兩個

方法做結合，計算名詞和動詞的語意相似度，而形容詞和副詞則使用 VECTOR 計算。因此，整體而言，$sim(w_1, w_2)$的語意相似度的計算如公式 4 ~ 6 所示：

$$sim(w_1, w_2) = \begin{cases} len(w_1) \text{ or } len(w_2), & \text{if } w_1 = w_2 \text{ or } w_1, w_2 \text{ in the same synset} \\ 0, & \text{if } w_1, w_2 \text{ have different POS} \\ NVsim(w_1, w_2), & \text{if } w_1, w_2 \in Noun, Verb \\ AAsim(w_1, w_2), & \text{if } w_1, w_2 \in Adjective, Adverb \end{cases} \quad (4)$$

$$NVsim(w_1, w_2) = \max_{X,Y} \frac{2 \times PATH((w_1, sense_X), (w_2, sense_Y)) \times WUP((w_1, sense_X), (w_2, sense_Y))}{PATH((w_1, sense_X), (w_2, sense_Y)) + WUP((w_1, sense_X), (w_2, sense_Y))} \quad (5)$$

$$AAsim(w_1, w_2) = \max_{X,Y} VECTOR((w_1, sense_X), (w_2, sense_Y)) \quad (6)$$

2. 字詞與片語之語意相似度**(狀況 D、狀況 E)**

字詞與片語之語意相似度是以字詞為基礎來計算的，假設$w \in T \in$ Joint Term Vector，$p \in$ Joint Phrase Vector，我們需要逐一比對p中的字詞，找出當中相似度最大值，假設$p = \{p_t_1, p_t_2, \cdots, p_t_N, \cdots\}$，其中$p_t_N$為片語$p$的第 N 個字詞，此時$sim(w, p)$的計算方式如公式 7~10 所示：

$$sim(w, p) = \max_N sim(w, p\_t_N) \quad (7)$$

$$\text{其中} sim(w, p\_t_N) = \begin{cases} 1, & \text{if } w = p\_t_N \text{ or } w, p\_t_N \text{ in the same synset} \\ 0, & \text{if } w, p\_t_N \text{ have different POS} \\ NVsim(w, p\_t_N), & \text{if } w, p\_t_N \in Noun, Verb \\ AAsim(w, p\_t_N), & \text{if } w, p\_t_N \in Adjective, Adverb \end{cases} \quad (8)$$

$$NVsim(w, p\_t_N) = \max_{X,Y} \frac{2 \times PATH((w, sense_X), (p\_t_N, sense_Y)) \times WUP((w, sense_X), (p\_t_N, sense_Y))}{PATH((w, sense_X), (p\_t_N, sense_Y)) + WUP((w, sense_X), (p\_t_N, sense_Y))} \quad (9)$$

$$AAsim(w, p\_t_N) = \max_{X,Y} VECTOR((w, sense_X), (p\_t_N, sense_Y)) \quad (10)$$

3. 片語與片語之語意相似度**(狀況 F)**

片語與片語之語意相似度也是由前面兩種相似度為基礎來計算的，假設$p_1, p_2 \in$ Joint Phrase Vector，我們需要一一比對p_1, p_2中所有字詞，找出當中相似度最大值，假設$p_1 = \{p_1_t_1, p_1_t_2, \cdots, p_1_t_M, \cdots\}$，其中$p_1_t_M$為片語$p_1$的第 M 個字詞，$p_2 = \{p_2\_t_1, p_2\_t_2, \cdots, p_2\_t_N, \cdots\}$，其中$p_2_t_N$為片語$p_2$的第 N 個字詞，此時$sim(p_1, p_2)$的計算方式如公式 11 ~ 14 所示：

$$sim(p_1, p_2) = \max_{M,N} sim(p_1\_t_M, p_2\_t_N) \quad (11)$$

$$\text{其中} sim(p_1\_t_M, p_2\_t_N) = \begin{cases} len(p_1\_t_M) \text{ or } len(p_2\_t_N), & \text{if } p_1\_t_M = p_2\_t_N \text{ or } p_1\_t_M, p_2\_t_N \text{ in the same synset} \\ 0, & \text{if } p_1\_t_M, p_2\_t_N \text{ have different POS} \\ NVsim(p_1\_t_M, p_2\_t_N), & \text{if } p_1\_t_M, p_2\_t_N \in Noun, Verb \\ AAsim(p_1\_t_M, p_2\_t_N), & \text{if } p_1\_t_M, p_2\_t_N \in Adjective, Adverb \end{cases} \quad (12)$$

$$NVsim(p_1\_t_M, p_2\_t_N) = \max_{X,Y} \frac{2 \times PATH((p_1\_t_M, sense_X), (p_2\_t_N, sense_Y)) \times WUP((p_1\_t_M, sense_X), (p_2\_t_N, sense_Y))}{PATH((p_1\_t_M, sense_X), (p_2\_t_N, sense_Y)) + WUP((p_1\_t_M, sense_X), (p_2\_t_N, sense_Y))} \quad (13)$$

$$AAsim(p_1\_t_M, p_2\_t_N) = \max_{X,Y} VECTOR((p_1\_t_M, sense_X), (p_2\_t_N, sense_Y)) \quad (14)$$

（二） 句子相似度計算

我們定義了計算字詞和片語的語意相似度的方法後，接著需要將 Joint Term Vector 和 Joint Phrase Vector 按照次序排序、去除重覆的項目，組合成 Joint Term & Phrase Vector，再跟來源文件S_{ni}與使用者修改文件S_j進行語意比對及次序比對，假設 Joint Term & Phrase Vector $=\{JTP_1, JTP_2, \cdots, JTP_f\}$，$S_{ni}=\{T_{ni}, P_{ni}\}=\{TP_1, TP_2, \cdots, TP_g, \cdots\}$，$S_j=\{T_j, P_j\}=\{TP_1, TP_2, \cdots, TP_h, \cdots\}$，我們參考 Li et al.(2006)所定義的句子相似度計算方法，同時考量同義字替代及字詞次序的改變，結合語意相似度(Semantic Similarity)與次序相似度(Order Similarity)，來計算S_{ni}和S_j整體的相似度，計算步驟如下：

1. 計算句子語意相似度

Joint Term & Phrase Vector 所有的JTP_f，與句子S_{ni}內所有的TP_g進行比對：

$$\boldsymbol{sem_1}=\{\max_g sim(JTP_1, TP_g), \max_g sim(JTP_2, TP_g), \cdots \max_g sim(JTP_f, TP_g)\} \quad (15)$$

Joint Term & Phrase Vector 所有的JTP_f，與S_j內所有的TP_h進行比對：

$$\boldsymbol{sem_2}=\{\max_h sim(JTP_1, TP_h), \max_h sim(JTP_2, TP_h), \cdots \max_h sim(JTP_f, TP_h)\} \quad (16)$$

最後兩句子的語意相似度，需要使用 cosine similarity 來計算：

$$Sim_{sem}=\frac{\boldsymbol{sem_1}\cdot\boldsymbol{sem_2}}{\|\boldsymbol{sem_1}\|\cdot\|\boldsymbol{sem_2}\|} \quad (17)$$

2. 計算句子次序相似度

在計算$\boldsymbol{sem_1}$和$\boldsymbol{sem_2}$的同時，我們需要把每一個產生最大相似度的g和h紀錄起來，但若相似度皆為 0，order 就是 0，組成$\boldsymbol{order_1}$和$\boldsymbol{order_2}$，最後兩句子的次序相似度為：

$$Sim_{order}=1-\frac{\|\boldsymbol{order_1}-\boldsymbol{order_2}\|}{\|\boldsymbol{order_1}+\boldsymbol{order_2}\|} \quad (18)$$

3. 計算句子整體相似度

給予不同的權重，結合以上兩種相似度計算方式，計算句子整體相似度：

$$Sim(S_{ni}, S_j)=\alpha Sim_{sem}+(1-\alpha)Sim_{order} \quad (19)$$

其中$\alpha \leq 1$，而 Li et al.(2006)認為α應該要大於 0.5，$\alpha \in (0.5, 1]$。若$Sim(S_{ni}, S_j)$>門檻值τ，該修改文件則會被視為有抄襲情形。系統將會利用 Walker(2008)所定義的五種改寫手法作進一步的分析，讓使用者了解自己的改寫方式是否觸犯抄襲。

五、 改寫手法分析階段

本研究使用 Walker (2008)所定義的五種併湊寫作手法，詳細分析改寫句子已使用哪些改寫手法，並清楚告知使用者改寫時違反了哪些改寫原則。改寫原則如下：

（一） 改寫時不能夠從原文中直接抄寫超過 5 個字詞(No Patchwriting-word string)

（二） 改寫時需要有同義字的替換(Patchwriting-substitution)

（三） 改寫時需要從原文中增加字詞(Patchwriting-addition)

（四） 改寫時需要從原文中刪減字詞(Patchwriting-deletion)

（五） 改寫時需要倒裝置換文句架構(Patchwriting-reversal)

肆、結論

本研究預期採用的資料集為 PAN-PC-09(Potthast, Eiselt, Barrón-Cedeño, Stein, & Rosso, 2009)、Clough & Stevenson(2011)在其研究“Developing a corpus of plagiarised short

answer”中所產生的資料集，由於資料集中已有定義文件是否抄襲，故本研究以此結果為標準答案。而為了驗証本研究所提出以片語為單位之改寫句子抄襲偵測方法的品質，本研究將使用 *Precision*、*Recall* 和 *F-Measure* 作為評估指標，並與傳統只使用單一字詞的偵測方法作比較，期望在相較之下會有較好的結果。最後將針對相似度高於門檻值的成對句子進行分析，清楚告知使用者已使用哪些改寫手法，藉此讓使用者了解自身的改寫手法是否觸犯抄襲。

The project was funded by Taiwan NSC (NSC 100-2631-S-006-001-CC3)

參考文獻

1. Canexus Inc. EVE2 - Essay Verification Engine., Nov 17, 2011 (available online at http://www.canexus.com/)
2. CFL Software Limited. CopyCatch., Nov 17, 2011 (available online at http://cflsoftware.com/)
3. Chen, C.-Y., Yeh, J.-Y., & Ke, H.-R. Plagiarism Detection using ROUGE and WordNet. *Journal of Computing, 2*(3), 2010, pp. 34-44.
4. Clough, P., & Stevenson, M. (2011). Developing a corpus of plagiarised short answers. *Language Resources and Evaluation, 45*(1), 5-24.
5. Culwin, F., & Lancaster, T. A review of electronic services for plagiarism detection in student submissions. Paper presented at the *LTSN-ICS 1st Annual Conference*, 2000.
6. Jiang, J. J., & Conrath, D. W. Semantic Similarity Based on Corpus Statistics and Lexical Taxonomy. Paper presented at the *International Conference Research on Computational Linguistics (ROCLING X)*, 1997.
7. Kang, N., Gelbukh, A., & Han, S. PPChecker: Plagiarism pattern checker in document copy detection. In P. K. I. P. K. Sojka (Ed.), *Text, Speech and Dialogue, Proceedings, 4188*, 2006, pp. 661-667.
8. Li, Y. H., McLean, D., Bandar, Z. A., O'Shea, J. D., & Crockett, K. Sentence similarity based on semantic nets and corpus statistics. *IEEE Transactions on Knowledge and Data Engineering, 18*(8), 2006, pp. 1138-1150.
9. Lin, D. Using syntactic dependency as local context to resolve word sense ambiguity. Paper presented at the Proceedings of *the 35th Annual Meeting of the Association for Computational Linguistics and Eighth Conference of the European Chapter of the Association for Computational Linguistics*, 1997.
10. Losee, R. M. Natural language processing in support of decision-making: phrases and part-of-speech tagging. *Information Processing & Management, 37*(6), 2001, pp. 769-787.
11. Maurer, H., Kappe, F., & Zaka, B. Plagiarism - A Survey. *Journal of Universal Computer Science, 12*(8), 2006, pp. 1050-1084.
12. Mozgovoy, M., Kakkonen, T., & Cosma, G. Automatic student plagiarism detection:

future perspectives. *Journal of Educational Computing Research, 43*(4), 2010, pp. 511-531.

13. Oetsch, J., Puehrer, J., Schwengerer, M., & Tompits, H. The system Kato: Detecting cases of plagiarism for answer-set programs. *Theory and Practice of Logic Programming, 10*, 2010, pp. 759-775.
14. Oliva, J., Ignacio Serrano, J., Dolores del Castillo, M., & Iglesias, A. SyMSS: A syntax-based measure for short-text semantic similarity. *Data & Knowledge Engineering, 70*(4), 2011, pp. 390-405.
15. Patwardhan, S. Incorporating Dictionary and Corpus Information Into a Context Vector Measure of Semantic Relatedness. University of Minnesota, Duluth, 2003.
16. Pecorari, D. Good and original: Plagiarism and patchwriting in academic second-language writing. *Journal of Second Language Writing, 12*(4), 2003, pp. 317-345.
17. Pera, M. S., & Ng, Y.-K. SimPaD: A word-similarity sentence-based plagiarism detection tool on Web documents. *Web Intelligence and Agent Systems, 9*(1), 2011, pp. 27-41.
18. Potthast, M., Stein, B., Eiselt, A., Barrón-Cedeño, A., & Rosso, P. Overview of the 1st International Competition on Plagiarism Detection. Paper presented at the *PAN-09 3rd Workshop on Uncovering Plagiarism, Authorship and Social Software Misuse and 1st International Competition on Plagiarism Detection*, 2009.
19. Potthast, M., Eiselt, A., Barrón-Cedeño, A., & Stein, B., Rosso, P. Overview of the 3rd International Competition on Plagiarism Detection. Notebook Papers of *CLEF 2011 LABs and Workshops, 19–22 September*, Amsterdam, Netherlands, 2011.
20. Rabin, M. O. Fingerprinting by Random Polynomials. *Center for Research in Computing Technology, Harvard University, Report TR-15-81*, 1981.
21. Rada, R., Mili, H., Bicknell, E., & Blettner, M. Development and application of a metric on semantic nets. *IEEE Transactions on Systems Man and Cybernetics, 19*(1), 1989, pp. 17-30.
22. Resnik, P. Using information content to evaluate semantic similarity in a taxonomy. Paper presented at the Proceedings of *the 14th international joint conference on Artificial intelligence*, 1995
23. Sun, Y.-C. Using a two-tier test in examining Taiwan graduate students' perspectives on paraphrasing strategies. *Asia Pacific Education Review, 10*(3), 2009, pp. 399-408.
24. Uzuner, Ö ., Katz, B., & Nahnsen, T. Using Syntactic Information to Identify Plagiarism. *Proceedings of the 2nd Workshop on Building Educational Applications Using NLP*, 2005, pp. 37-44.
25. Walker, A. L. Preventing Unintentional Plagiarism: A Method for Strengthening Paraphrasing Skills. *Journal of Instructional Psychology, 35*(4), 2008, pp. 387-395.
26. Wu, Z., & Palmer, M. Verb semantics and lexical selection. Paper presented at *the 32nd.*

Annual Meeting of the Association for Computational Linguistics, 1994.
27. Yamada, K. What prevents ESL/EFL writers from avoiding plagiarism?: Analyses of 10 North-American college websites. *System, 31*(2), 2003, pp. 247-258.
28. Zaka, B. Empowering Plagiarism Detection with a Web Services Enabled Collaborative Network. *Journal of Information Science and Engineering, 25*(5), 2009, pp. 1391-1403.
29. Bull, J., Colins, C., Coughlin, E., & Sharp, D. Technical review of plagiarism detection software report, Nov 2001, (available online at http://www.jisc.ac.uk/uploaded_documents/luton.pdf)
30. Howard, R. M. Plagiarism: What Should a Teacher Do? 2001 (available online at http://wrt-howard.syr.edu/Papers/CCCC2001.html)
31. iParadigms. TurnitIn.com. Digital assessment suite. Nov 17, 2011 (available online at http://turnitin.com)
32. 教育部高教司，大學校務評鑑規劃與實施計畫ㄧ評鑑手冊，民 93，取自：http://academic.ntou.edu.tw/service/dia/ntou/book1.pdf

A plagiarism detecting method of paraphrased sentences using syntactical structure and semantic similarity

Ka Weng Sou
National Cheng Kung University Institute of Information Management
soukaweng@hotmail.com

Hei Chia Wang
National Cheng Kung University Institute of Information Management
hcwang@mail.ncku.edu.tw

Abstract

Recently, academic researches have been getting more and more attention. Along with the advance of Internet, it is easier to obtain information on the web. It leads researchers to copy someone's ideas intentionally or unintentionally. In fact, many researchers' recognition of plagiarism is incorrect. Hence, there is a requirement of an environment which can help researchers and educate them detect whether their documents are plagiarized or not. However, the existing plagiarism detecting systems can only detect simple plagiarism, and the systems are not able to instruct researchers to paraphrase sentences properly.

In this study, a new plagiarism detection method for detecting the paraphrase behavior of modified sentences will be proposed. Using the syntactical structure to reveal the semantic similarity in order to identify the plagiarism, automatically analyze and determine whether the paraphrased documents are plagiarized or not. We are looking forward to tell user which

paraphrased technique they has used, let them know if their paraphrased way is plagiarized.

Keywords: paraphrase, plagiarism detection, syntactical structure, semantic similarity