2021 年臺灣國際科學展覽會
優勝作品專輯

作品編號　**190011**

參展科別　電腦科學與資訊工程

作品名稱　中文重點文句摘取

得獎獎項　大會獎 四等獎

就讀學校　臺北市立第一女子高級中學

指導教師　陳信希、黃芳蘭

作者姓名　黃莉棋

關鍵詞　中文摘要、演算法、詞嵌入

作者簡介

我是北一女中黃莉棋，從小就對科學和電腦有興趣的我，卻從來沒什麼機會投入其中，加上不是數理資優班，也不是科學班的學生，身為普通班學生的我更難獲取這方面的資源。但是後來參加科教館主辦的未來之星營隊，讓我發現並不是沒有機會，只是我們需要自己去找尋機會。投入科展後才發現，其實很多教授、老師、學長姊，都非常熱心，絕對不會有「缺少資源」的問題，只要肯伸手、肯開口發問，不論自己的實力如何、身分如何，他們都很願意幫助我，讓我非常感激也非常感動。雖然才短短幾個月的時間，也深知自己還有很大的進步空間，但這次的經驗不論如何，都讓我學習到了很多知識，更讓我知道每件事情背後都有一群值得好好感謝的人。

摘要

在資訊爆炸的時代，效率閱讀、整理資料的能力越趨重要。身為高中生，學習時的閱讀量龐大，還須另外自己挑選重點句，重新整理筆記。因此我想如果可以讓電腦自動摘取文章的重點，就能幫助學生效率學習。

大多數現存的自動摘要研究適用於英文文本，本研究利用演算法抓取中文文章的摘要，使學生可以真正實用該演算法於日常學習當中。除此之外，此研究比較了不同方法摘要的準確率以及優缺點。

Abstract

In the era of information explosion, efficient reading and the ability to organize data are becoming more and more important. But as a high school student, I still have to absorb knowledge by read a large amount of data, which is neither efficient nor effective. Therefore, in this research, my goal is to use algorithm to extract key sentences from Chinese articles, in order to apply this research to my everyday life.

Most researches I found about summarization are for English articles, rather than traditional Chinese articles. So in this research, the goal is to design algorithms for summarizing Chinese articles. Apart from that, I also compared different algorithms, by acknowledging their pros and cons, I later on improved the algorithm and increased the accuracy.

壹、 前言

一、 研究動機

身在資訊爆炸的時代，效率學習是不可欠缺的能力。然而在高中階段學習的我們，仍然要從厚重的書本獲得知識，閱讀的份量不但沉重，也欠缺效率，往往一篇課文要花上幾個小時的時間才能整理出重點。因此此研究的動機即是以摘取式的方法快速提取中文文章的重點，仿效多數學生學習的方法，以類似「畫螢光筆」的方式迅速摘取重點，讓此研究得以實際上提升學習的效率。除了為了符合實用功能，選擇摘要中文文本的另一原因是，目前現存的摘要作品大部分都只適用於英文，缺乏關於中文摘要的研究，因此想要往這方面研究。

二、 研究目的

此研究的目的是利用演算法提取中文文章的重點，將一篇文章化繁為簡，以較簡短的句子呈現冗長文章的主旨，並且比較不同演算法對於摘要的準確性，評量各自的優缺點之後，再更進一步改進演算法。相較於其他類似研究，此研究除了在處理的語言上有所不同，更比較不同方法的優劣，分析各自的特性和長處，以此為依據繼續改良摘要方法。

貳、 研究方法及過程

一、 研究設備與器材

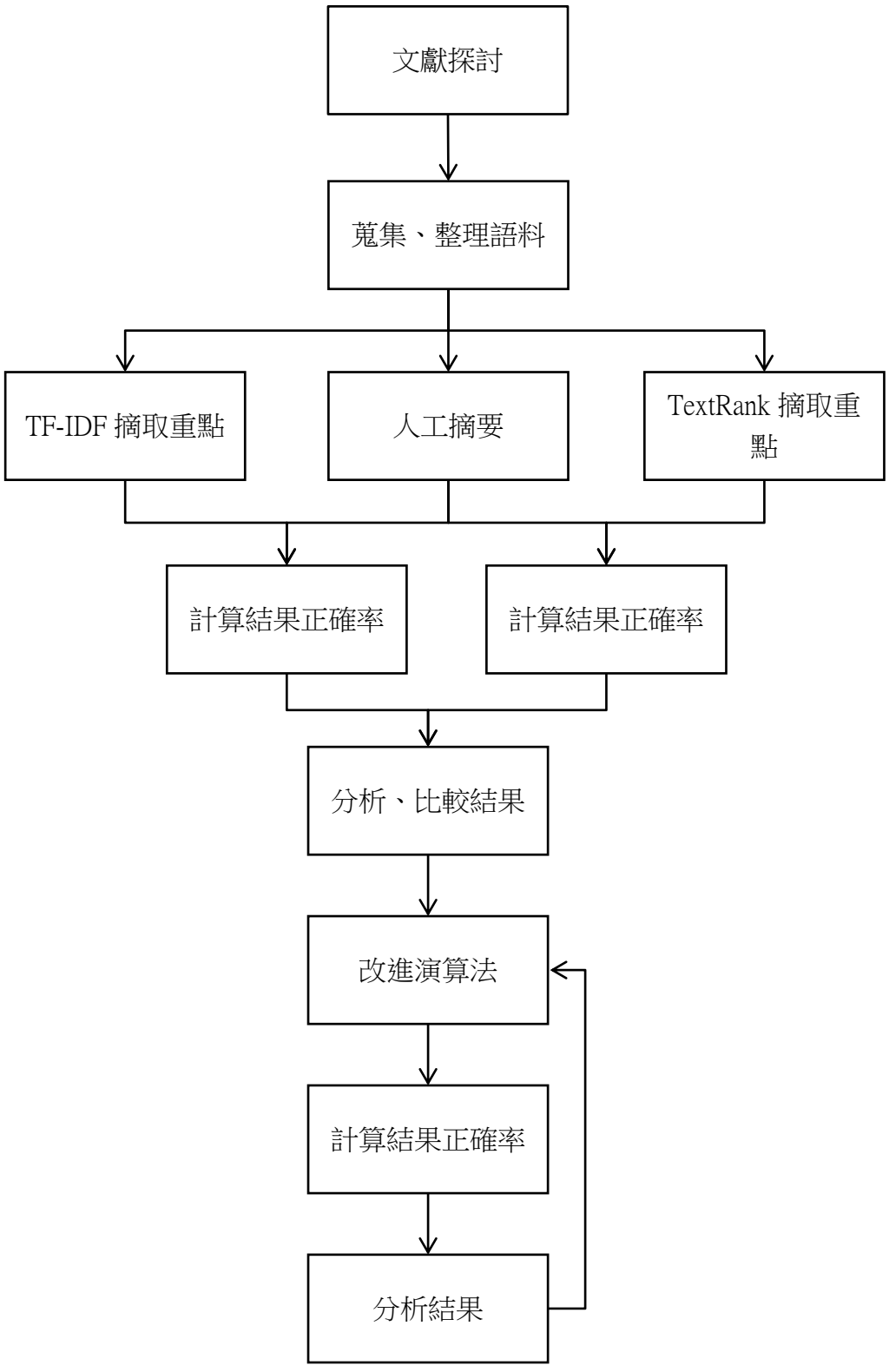
(一)、 硬體設備

1. 個人筆電 CPU Intel Core i5, Ram 8.00GB
2. GPU GTX Titan X

(二)、 軟體環境

1. Python: Jupyter Notebook
2. Microsoft Office: Excel, Word
3. 記事本

二、 研究構思與架構


文獻探討
蒐集、整理語料
TF-IDF 摘取重點
人工摘要
TextRank 摘取重點
計算結果正確率
計算結果正確率
分析、比較結果
改進演算法
計算結果正確率
分析結果

三、 研究過程

(一)、 參考資料與文獻探討

1. 文獻探討

自然語言處理是將人類的語言轉化為電腦可以理解的形式，並利用電腦的運算功能和人工智慧進行對於人類語言的分析和處理，常見的用途為文本摘要、語言翻譯、對話系統、語音辨識等等，範疇十分廣泛。其中文本摘要分為抽取式摘要(Extractive Summarization)和抽象式摘要(Abstractive Summarization)，前者是將文章中的重點部分直接摘取出來，後者則是經由機器學習使電腦理解文章後重新生成一篇摘要。

在查詢資料的過程我發現大部分的文本摘要對象皆是英文，而且大多數的摘要目的是產生新聞標題，因此這些研究都是在 Abstractive Summarization 的範疇。但在我的生活中，我在學習和讀書時經常都是使用 Extractive Summarization 的模式，例如在課本中畫重點句。這種模式不像生成標題這麼的要求嚴苛，只須有效挑選出學習重點，就能幫助我更加效率的學習。雖然這方面的文獻較少，但是身為學生我知道這種需求是確實存在的，因此我決定朝向較少人做的繁體中文抽取式摘要進行研究。

2. 中文語言特性

參考其他文本摘要的研究後，我們發現雖然在抓取重點時，使用的邏輯是相同的，但是中文語文特性有別於英文，因此語料庫的內容也需重新挑選。以下是幾個比較重要的比較。

項目	英文	中文
分句判斷依據	「.」、「!」、「?」、「…」、「\n」等符號	「。」、「!」、「?」、「…」、「\n」等符號
分詞判斷依據	空格符號「 」	需另外判斷
大小寫轉換	進行 lower case，將所有大寫轉小寫	不需進行轉換
詞性標註	進行 pos-tagging，將詞語標註	不須另外標註

中英文語言特性差異頗大，除了常見的英文單、複數，不同時態變形等等，雙方的結構也完全不同。雖然在演算法中的原理相似，但是在預處理文章的部分就存在很大差異。中文語言的結構為黏合結構，詞與詞之間以連結詞或虛辭連結。而英文的則為楯合結構，前後詞語需互相配合、改變。而在分詞部分，英文語文當中我們可以很快的分開每個單詞，但是在中文中，無法找到一個標準的邊界，累積足夠經驗的我們雖然能夠正確將中文文句分詞，但對於電腦，需要大量資料進行比對才能完成這項任務。

(二)、 蒐集並整理文章資料

1. 三民出版社社會科課本

經由三民出版社我們取得公民、歷史、地理三科的高中課本 word 檔，但原始檔案當中仍夾雜表格與圖片，因此我們使用 Python 抓取課文文字部分，排除掉原本的排版和圖片等等。由於課本中沒有明確的對應摘要，因此最後用來評分的「標準摘要」是人為挑選的，為了避免個人主觀影響摘要正確性，我們找了三名人員挑選三組標準摘要。

(三)、 使用演算法抓取摘要

1. 文章預處理

在自然語言處理當中，文章預處理是重要的步驟，因為要將人類的語言轉成電腦可以讀懂的格式之後，電腦才能進一步處理文字。文章預處理主要分為分句、分詞、去停用詞這幾個步驟。其實原本在英文摘要的相關研究中，還有大小寫轉換、詞性標註、時態轉換等等步驟，但因為中英文的語言特性不同，因此在我們的研究中，取掉這些不必要的步驟。另外在一些研究中，會將非中文的字符去除，但在本研究中，我們保留了數字字符，因為在高中教科書中往往會包含年代、數據等等以數字呈現的資料。

甲、 整理文章

首先我們將三民出版社的公民、地理、歷史三個科目課本中，挑出文章並直接以中文字符格式文章整理於 excel 表格中，文章分成兩類，一類是毫無關係的多篇個別文章，另一類是出自同一主題的多篇文章。分成這兩類的目的是要比較演算法對於不同類型文章的摘要能力差別。

以下為了方便解釋，定義「文檔一」和「文檔二」當作範例。文檔一當中具有兩篇互相沒有關係的文章，而文檔二當中的兩篇文章都是關於同一主題。如下:

文檔一	文章一	小明喜歡吃蘋果。蘋果是一種紅色的水果。
	文章二	快點看那裏!大象和長頸鹿是朋友。
文檔二	文章一	小明喜歡吃蘋果。蘋果是一種紅色的水果。
	文章二	我也覺得蘋果很好吃。上次我在市場買了一顆。

乙、 分句

我們對整理好的檔案進行了預處理，首先將文本進行分句與分詞。分句是將文章每個句子分開，方便對每個句子進行個別處理，分句的方法為判斷文章中「?」、「!」、「;」、「。」、「…」、「\n」等符號所在位置，後來我們發現輸入的文章可能是全形字符格式，因此新增了「？」、「！」、「；」。我們將這些符號前的詞語抓進陣列中，最後文章會被分成一個二維陣列，每個一維陣列中包含一個句子。以下以上面舉例的文檔一當中的文章一作範例:

分句前	小明喜歡吃蘋果。蘋果是一種紅色的水果。
分句後	[[小明喜歡吃蘋果。], [蘋果是一種紅色的水果。]]

丙、 分詞

自然語言的處理最小單位為「詞」，因此在預處理時需要進行分詞。分詞的目的是將原本

的文章切割，使每一個詞語獨立，之後才能將這些詞語作處理。因為和英文語文特性不同，中文分詞不是以「空格」為基準，因此另外導入 jieba 分詞系統進行分詞。我們選擇 jieba 分詞系統的原因是此分詞系統中包含的中文語料庫，包括繁體中文，而且已具備豐富資料，因此可以依據詞典中的詞語判斷如何對於繁體中文文句進行分詞。

分詞之後將詞語存在陣列中，而最終整編文成為一個二維陣列，原本包含單一句子的陣列，內容變成分詞之後的詞與元素組成。如下:

分詞前	[[小明喜歡吃蘋果。], [蘋果是一種紅色的水果。]]
分詞後	[['小明', '喜歡', '吃', '蘋果', '。'], ['蘋果', '是', '一', '種', '紅色', '的', '水果', '。']]

丁、 去除停用詞及標點符號

接著導入停用詞列表(一行一個停用詞)，將文章中的停用詞移除。停用詞是一些名詞、動詞、形容詞以外其他較不重要且經常出現的詞，例如「的」、「我」、「是」等等。移除停用詞的目的是避免這些非關鍵的詞語成為影響摘要的因素。另外標點符號由於不列入計算的過程中，也需與預處理的過程中去除。

以上面例子沿用，去除停用詞和標點符號如下:

去除停用詞前	[['小明', '喜歡', '吃', '蘋果', '。'], ['蘋果', '是', '一', '種', '紅色', '的', '水果', '。']]
欲去除目標	[['小明', '喜歡', '吃', '蘋果', '。'], ['蘋果', '是', '一', '種', '紅色', '的', '水果', '。']]
去除停用詞後	[['小明', '喜歡', '吃', '蘋果'], ['蘋果', '是', '紅色', '水果']]

預處理完之後，就可以進入演算法環節。

2. 利用 TF-IDF 分數抓取文章摘要

TF-IDF 當中的 TF 代表詞頻(Term Frequency)，而 IDF 代表逆向檔案頻率(Inverse Document Frequency)。詞頻是指一個詞語在整篇文章中的出現頻率，逆向檔案頻率是指總檔案數目除以包含該詞語之檔案的數目，透過這兩個數值我們可以計算每個詞語的重要程度 TF-IDF 分數越高者，代表越重要。

TF 的計算方法如下:

$$\frac{\text{某詞語出現在文章中的次數}}{\text{文章的總字數}}$$

IDF 的計算方法如下:

$$\log\left(\frac{\text{語料庫的文檔總數}}{\text{包含該詞語的文檔總數}}\right)$$

TF-IDF 分數的計算方式如下:

$$\text{TF x IDF}$$

TF-IDF 是一個相對直觀的演算法，將詞語的重要程度和該詞語的詞頻連結在一起，因此 TF 分數越高，最後的分數也會越高。但該演算法也有考慮到該詞語在所有文章中的稀有性，在 IDF 分數計算中，分母為「包含該詞語的文檔總數」，因為在同一主題的多篇文章內，存在很多相同的詞語，因此使用 IDf 分數能提高較稀有但較重要的關鍵詞分數，降低常常出現、詞頻高但相對較不重要的詞語。也就是說，詞頻高不一定等於關鍵信息，IDF 在詞頻方面具有限制效果，預防高頻率出現的詞語過度影響摘要的產生。

值得注意的是，因為 TF-IDF 需要計算一整個文檔的逆向檔案頻率，因此輸入時要先將一整個文檔輸入才能做運算，將每個文章存進陣列。舉上述的文檔二來說:

文章一	小明喜歡吃蘋果。蘋果是一種紅色的水果。
文章二	我也覺得蘋果很好吃。上次我在市場買了一顆。
輸入模式	["小明喜歡吃蘋果。蘋果是一種紅色的水果。", "我也覺得蘋果很好吃。上次我在市場買了一顆。"]

之後進行預處理，分句、分詞、去除停用詞之後，對每個詞語分別進行 TF-IDF 分數計算，降每個詞的分數依照順序存於陣列中。沿用文檔一的文章一第一句為例:

原句	小明喜歡吃蘋果。
預處理後	[['小明', '喜歡', '吃', '蘋果']]
TFIDF 分數	[[0.058, 0.058, 0.058, 0.116]]

因為“蘋果”的分數最高，蘋果為這個文檔中的關鍵詞。最後，找出具有最多關鍵詞的句子為重點句，輸出這些句子即為文章摘要。以下舉出實際摘要結果範例:

	原文檔	摘要
文章一	你曾聽說過任何關於難民的新聞嗎？被視作中國公民而遭瑞士拒絕難民身分的流亡藏人、因美國移民禁令而遭遣返的伊拉克難民？難民的認定和處置為何如此重要？在國內遭遇內戰、隔離政策，甚至大屠殺的人們，當他們長途跋涉逃至其他國家，卻可能因為沒有簽證和居留資格，而被其他國家以「非法入境」的理由遣返，送回迫害的源頭。因應許多透過國家暴力進行族群歧視的事件，二戰後世界人權宣言（Universal Declaration of Human Rights）規定「被庇護」為一項人權，不管要不要把難民視作本國人對待，至少讓他們能稍事歇息。	因應許多透過國家暴力進行族群歧視的事件，二戰後世界人權宣言（Universal Declaration of Human Rights）規定「被庇護」為一項人權，不管要不要把難民視作本國人對待，至少讓他們能稍事歇息。
文章二	1951 年聯合國通過關於難民地位公約（Convention Relating to the Status of Refugees）處理戰後歐洲地區內的難民。又於 1966 年通過關於難民地位議定書（Protocol Relating to the Status of Refugees），將範圍擴及全球，且不受時間、特定事件的限制。目前，普世通用的難民定義為：所有因種族、宗教、政治理念等原因，出於畏懼，不能或不願回到自己原本居住國家的人。	目前，普世通用的難民定義為：所有因種族、宗教、政治理念等原因，出於畏懼，不能或不願回到自己原本居住國家的人。
文章三	「公民」的概念發展至今，國籍和公民權利幾乎一體兩面，是一個人安身立命的門檻，這也是為何世界人權宣言會把「擁有國籍」視為基本人權。矛盾的是，要不要將「授予國籍和公民資格」歸屬於國家的權力，是國際法無法強制、不容干涉的內政事務。因此，目前關注無國籍問題的國際公約都不太受到重視，各國政府仍傾向保留「賦予國籍」的特權，不願輕易給予無國籍人居留或公民	「公民」的概念發展至今，國籍和公民權利幾乎一體兩面，是一個人安身立命的門檻，這也是為何世界人權宣言會把「擁有國籍」視為基本人權。

	資格。難民和無國籍人乍聽之下遠在天邊，但事實上，有一群移工小孩，因為雙親不是臺灣人，一出生就成為滯留臺灣的無國籍兒童。臺灣歷史長河中也出現過許多難民—1976 年來自越南和中南半島的難民，2014 年獲得政治庇護的法輪功教徒，皆以專案方式處理。臺灣媒體鮮少報導相關新聞，因此，我國國籍法的修正與難民法草案的推展在社會中受到的關注較少。然而，在全球人口流動頻繁、講究國際觀的今天，這應是值得重視的問題。	

3. TextRank 演算法抓取文章摘要

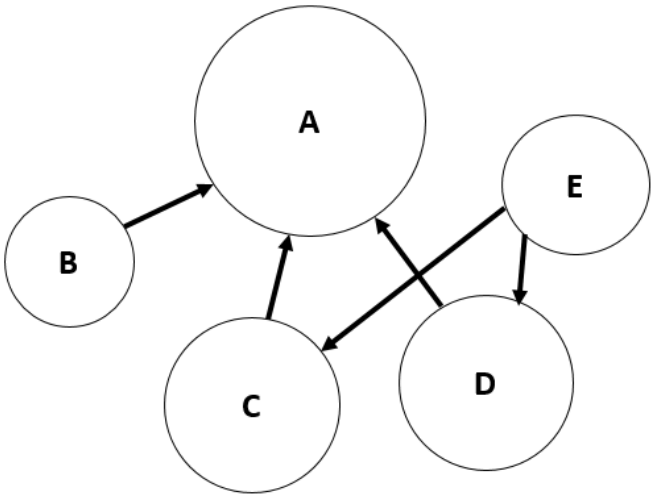
TextRank 為 PageRank 的延伸，利用不同詞語間「互相投票」找出關鍵文字，意即以文句間的連結強度判別文句的重要程度。PageRank 原為用在優化網頁搜尋的演算法，將每個網頁看作一個節點，將網頁間的超連結看作邊，節點之間以邊連結，建構出一個類似網路狀的架構。當一個節點有較多的邊，意即該節點和較多其他節點有連結，這個節點的權重就會較高。除此之外，權重還會被該節點的重要性影響，具有越多連結的節點重要性越高，權重也越高，因此也會拉高和該節點有連結的其他節點權重。

舉例來說，現在存在網頁 A、B、C、D、E，他們之間的關聯如下(0 代表兩網站間沒有關聯，1 則代表有):

	A	B	C	D	E
A		1	1	1	0
B	1		0	0	0
C	1	0		0	1
D	1	0	0		1
E	0	0	1	1	

圖(一)網頁關聯表:

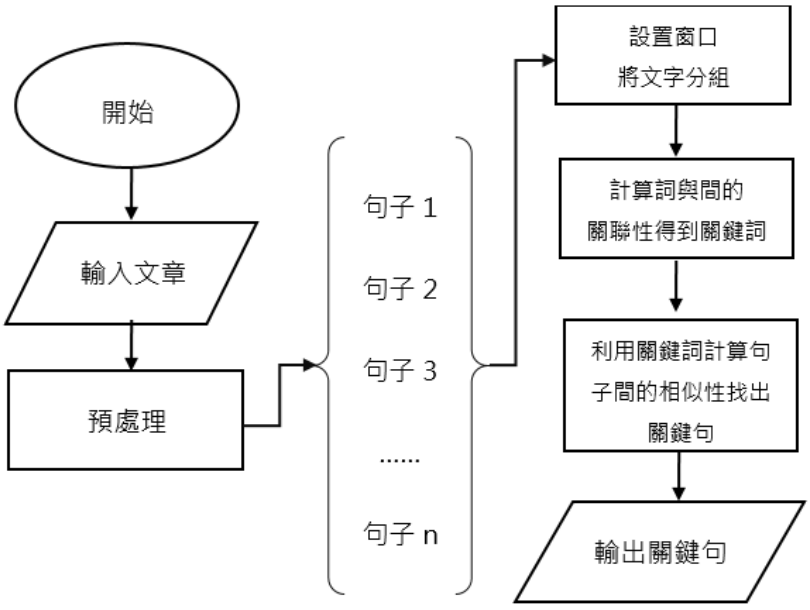
0 代表兩網頁間沒有關聯；1 則代表兩網頁間有關聯



圖(二)網頁關聯轉換成 DAG 圖:

DAG 圖為有向無環圖(directed acyclic graph)，在此使用 DAG 圖將網頁的連結關係表現出來。圖中圓圈大小代表節點權重相對大小，可知連結越多的節點權重越大。

值得注意的是，PageRank 的邊是有向性的，因為超連結在網頁中是有方向性且不可逆的，但在文章中的文字之間是沒有向性的。TextRank 運用 PageRank 的相同原理於文本摘要中，將文章去停用詞之後的每個句子當作節點，兩個句子之間的相似度為邊。句子的相似度利用不同句子之間的重複詞語來做運算。最後找出權重最高的句子，此即為關鍵句，當作摘要輸出。完整流程如下:



圖(三)TextRank 流程圖

甲、 設置窗口大小將文字分組

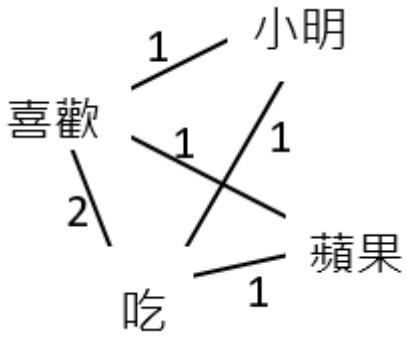
預處理完之後，將各個詞語分別對應於一個節點，並且指定滑動窗口大小。滑動窗口大小即代表「處理範圍大小」，若窗口大小為 n 時，則每一次則抓取連續的 n 個詞語來處理。每次抓取完之後，將選取範圍往下一個詞移動，直到該次詞組的最後一個詞是整個句子的最後一個詞，意即文字分組範圍只限於該詞存在的句子中，不會有跨句組詞的狀況。舉下例來說，當窗口大小設為 3(以黑框代表窗口):

預處理後文句	[['小明', '喜歡', '吃', '蘋果']]
第一次詞組	[['小明', '喜歡', '吃', '蘋果']]
	[['小明', '喜歡', '吃']]
第二次詞組	[['小明', '喜歡', '吃', '蘋果']]
	[['喜歡', '吃', '蘋果']]

乙、 計算各個詞組當中的詞語關聯得到關鍵詞

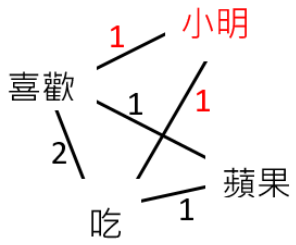
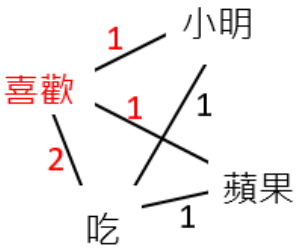
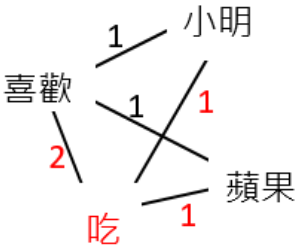
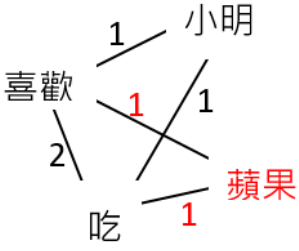
接續以上範例依照每次詞組找出詞語之間的關聯，用不具向量的邊呈現。值得注意的是，以下以括號呈現的「邊」只是形式上的表達，實際程式中並不會如此儲存，而是直接計算個詞語間的關聯次數，在進一步計算相似性，下面會再做說明。

第一次詞組	[['小明', '喜歡', '吃']]
第一次形成的邊	('小明', '喜歡'), ('喜歡', '吃'), ('小明', '吃')
第二次詞組	[['喜歡', '吃', '蘋果']]
第二次形成的邊	('喜歡', '吃'), ('吃', '蘋果'), ('喜歡', '蘋果')



圖(四)TextRank 詞語間的 DAG 關聯圖:

上圖是依照窗口大小為 3 時，節點間的邊畫成詞語關聯圖，每個邊旁的數字代表同樣兩個節點之間的連結次數。

小明		1+1=2
喜歡		1+1+2=4
吃		1+1+2=4
蘋果		1+1=2

統計後發現“喜歡”和“吃”這兩個詞的連結數皆為 4，而“小明”和“蘋果”則只有 2。因此可以得到“喜歡”和“吃”為這句話的關鍵詞。

丙、 利用關鍵詞和相似度找出關鍵句

接下來開始計算句子間的相似度。將分詞後且去除停用詞的句子之間互相比對計算，把兩兩句子間的相似度數值記錄在矩陣當中，而相似度最高的句子為關鍵句，即為提取式摘要。計算句子間的相似度方法如下:

$$Similarity(S_i, S_j) = \frac{|\{w_k | w_k \in S_i \& w_k \in S_j\}|}{log(|S_i|)+log(|S_j|)}$$

圖(五)TextRank 計算相似度的方式:

S_i 和 S_j各代表一個句子，W_k代表兩個句子中共同擁有的關鍵詞數量。TextRank 計算句子間的相似度方式以文字表示即為「兩句子中交集的關鍵詞數除以兩句子 log(字詞數量)相加」。

分母部分為兩個句子的字數各取 log 之後的和，此舉的目的是限制字數多的句子，因為若一個句子的字數很多，該句子和其他句子具有重複詞語的機率就會較高，容易被誤判為重點句。

相似度計算完之後，和先前的關鍵詞 DAG 圖一樣，每個句子此時被當作節點，相似度則為邊，最後統計擁有最高權重的句子即為該文章的重點文句。另外，我們在演算法中增設可調整重點句數的參數，舉例來說，若欲輸出三句摘要，演算法即會挑揀權重最高的三個句子輸出，如此一來，就可以根據個人需求和喜好摘取不同比例的摘要。以下舉出實際摘要的結果:

原文章	摘要
身為一個公民，是一種身分或是一種認知呢？舉辦公民對談時，常常有人迷惘地問起這件事，如果公民的權利很大，那在社會中沒有公民身分，但有公民認知的人，能不能被	公民對談的精神就在於開始問問題，去建立理解不同觀點與經驗所需要的各種邏輯，並發明／發現聆聽他人的方法，和聆聽不同觀點更好的習慣，去產生對話的可能，與更多不同

保障和擁有參與社會事務的發言權？如果身為一個公民責任也很大，那所謂的公民責任又是什麼呢？是服務和延續社會機制的運作？或也同樣是在知識、對人類行為與人權理解的智慧為基礎上，去挑戰與修繕制度之於每個當代的不足呢？公民對談的精神就在於開始問問題，去建立理解不同觀點與經驗所需要的各種邏輯，並發明／發現聆聽他人的方法，和聆聽不同觀點更好的習慣，去產生對話的可能，與更多不同途徑與結果的過程。對自己眼見、耳聞和感受到的事情問問題，是一種觀察的練習；對著自己使用詞彙的方式，和對詞彙背後真正的由來與意義和其演變，深入去整理，是對自己成長不向外求的進步。也因為這樣的動機帶來的對話經驗，我們也許會發現，對話有時並非為了說服他人或是宣揚自己現有的價值觀，對話更在於破壞—破壞對於各種話題的想像和既定認知—卻並非以消滅不同立場為最大的目標，進而達成了不同觀點與思考彼此「鑿溝互通」的可能。讓更多不同背景條件的人能在一個社會裡，也擁有與公民同樣擁有尊重、表達自己、哪怕是犯錯的權利。與人對話的學問與藝術，就在於這樣的一種破壞：一個不斷產生新的結果與新的組合的機會。	途徑與結果的過程。舉辦公民對談時，常常有人迷惘地問起這件事，如果公民的權利很大，那在社會中沒有公民身分，但有公民認知的人，能不能被保障和擁有參與社會事務的發言權。也因為這樣的動機帶來的對話經驗，我們也許會發現，對話有時並非為了說服他人或是宣揚自己現有的價值觀，對話更在於破壞—破壞對於各種話題的想像和既定認知—卻並非以消滅不同立場為最大的目標，進而達成了不同觀點與思考彼此「鑿溝互通」的可能。

(四)、 評估與比較演算法

1. 評分

高中課本平均一個章節由十句以內的句子組成，因此我們從文本中隨機挑選出 7 篇由 3 句以上、十句以內組成的文章，做為測試樣本。演算法從各篇文章中摘出 3 句摘要，評分方法如下:

$$\frac{\text{匹配標準摘要的句數}}{3} \times 100\%$$

「標準摘要」是人為挑出的摘要，而為了消除個人主觀所造成的誤差，我們找了三組人員選出摘要，分別為「已經學習過該內容的人」、「正在學習該內容的人」、以及「沒有學習過該內容的人」，每個人在每篇文章中挑出三句重點句，儲存在 excel 表格中。

舉例來說:

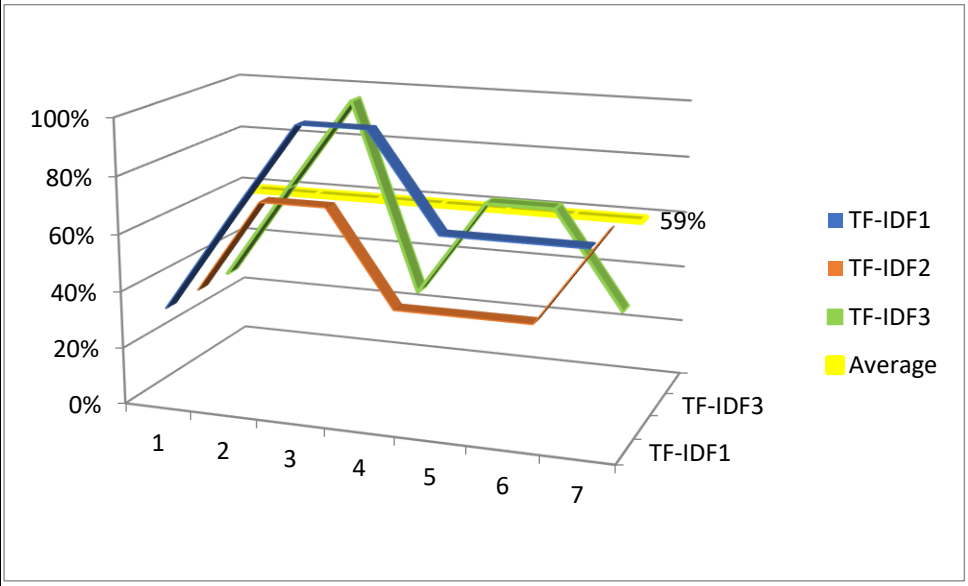
原文	政治權的核心在於透過讓人民享有參與政治生活的權利，以進一步落實並保障人民各項權利。在過去，人民只能參選地方行政首長及民意代表，中央民意代表只有部分增額補選，對於政府與政策的影響力有限。隨著政治民主化，逐步落實憲法對人民選舉、罷免、創制（人民直接投票立法）、複決（人民投票決定是否同意立法機關所通過的法案）、應考試、服公職的權利保障。中央民意代表全面改選，人民直選總統與副總統，修法放寬罷免限制，也落實公民投票的權利（第 5 課會進一步介紹）。而針對原住民族、婦女等族群，則透過憲法增修條文保障原住民族的立委席次、婦女的不分區立委當選名額；透過地方制度法保障直轄市議員、縣（市）議員、鄉（鎮、市）民代表婦女的當選名額。而隨著新住民人數增加，你贊成透過立法保障新住民在政治參與上的相關權利嗎？	人為摘要一	政治權的核心在於透過讓人民享有參與政治生活的權利，以進一步落實並保障人民各項權利。隨著政治民主化，逐步落實憲法對人民選舉、罷免、創制（人民直接投票立法）、複決（人民投票決定是否同意立法機關所通過的法案）、應考試、服公職的權利保障。而針對原住民族、婦女等族群，則透過憲法增修條文保障原住民族的立委席次、婦女的不分區立委當選名額；透過地方制度法保障直轄市議員、縣（市）議員、鄉（鎮、市）民代表婦女的當選名額。
		人為摘要二	政治權的核心在於透過讓人民享有參與政治生活的權利，以進一步落實並保障人民各項權利。隨著政治民主化，逐步落實憲法對人民選舉、罷免、創制、複決、應考試、服公職的權利保障。而隨著新住民人數增加，你贊成透過立法保障新住民在政治參與上的相關權利嗎？
		人為摘要三	政治權的核心在於透過讓人民享有參與政治生活的權利，以進一步落實並保障人民各項權利。隨著政治民主化，逐步落實憲法對人民選舉、罷免、創

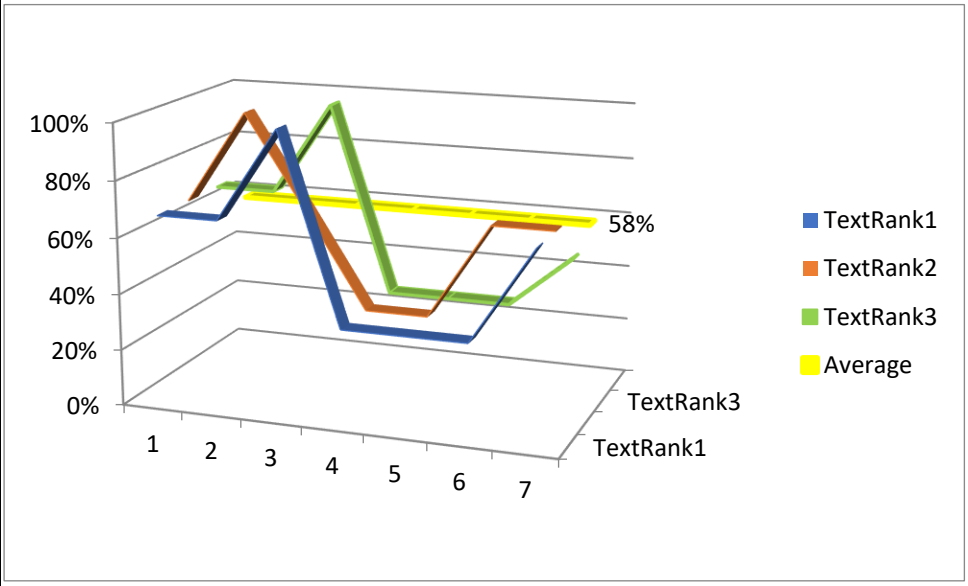
			制（人民直接投票立法）、複決（人民投票決定是否同意立法機關所通過的法案）、應考試、服公職的權利保障。而針對原住民族、婦女等族群，則透過憲法增修條文保障原住民族的立委席次、婦女的不分區立委當選名額；透過地方制度法保障直轄市議員、縣（市）議員、鄉（鎮、市）民代表婦女的當選名額。

可以看出對於相同的文章，每個人的摘要選擇也不同，但是也有大家共同認為的重點，因此採取多人的摘要較能消除個人的主觀意見，並凸顯真正的重要句子。

接著計算每篇「演算法抓出的摘要」對於「標準摘要」的 ROUGE-1 分數，以百分比呈現。另外，文章分為「獨立文章」和「主題文章」，「獨立文章」代表互相沒有關係的多篇獨立文章，而「主題文章」代表圍繞同一主題的多篇文章此舉目的是用來比較 TextRank 和 TF-IDF 的不同功能。

以下是多篇獨立文章分別利用 TextRank 和 TF-IDF 進行摘要的評分結果:

TF-IDF 由三組人員的標準答案算出的評分結果及平均準確率	

TextRank 由三組人員的標準答案算出的評分結果及平均準確率	

最後將三次評估的平均數值如下:

TextRank	TF-IDf
58%	59%

2. 評估結果

從評估可知，在獨立文章的摘要抓取中，TextRank 和 TF-IDF 的摘要能力差不多，皆將近達到 60%的準確率，雖然兩者都以詞頻高低為判斷重點的依據，但是 TF-IDF 的結果略好於 TextRank，因為 TF-IDF 具有 IDF 分數限制文本中高頻單字對重要性的影響，而 TextRank 則沒有。

同樣的原因，TextRank 在上面的摺線圖中起伏變化也較大，因為它容易受待摘要文章中的內容影響，若內容較為分散，則難以找到真正的重點。

(五)、 改良演算法

由上述評估結果，TextRank 的表現受限於待摘要文章的內容，且 TextRank 演算法具有較

高的改進彈性(相較於 TF-IDF 單純的算法)，我們決定以改進 TextRank 為主要目標。TextRank 主要依賴計算彼此之間的相似度來找出重點，而在原本的演算法中，計算相似度的範圍僅限在與摘取重點的文章中，但是如此一來，判定的範圍只有短短一篇文章，若這篇文章有較為偏頗的用詞或寫法，則會影響該篇文章的重點判定。尤其是此研究的主要目標是高中教科書課本，而高中課本中擁有獨特的用詞和寫法，因此若能完整考慮整體內容，再將其和計算相似度的過程結合，就可以有效提高摘要的正確率。

因此我們設計了結合詞向量的演算法，首先先將高中課文的內容建立成文本，進行 word2vec 模型的詞嵌入向量訓練，以高中課本的內容建立詞向量，讓 TextRank 演算法進行相似度運算時能更符合整體的狀況，而不是只由單一文章決定摘要。向量建構完畢後，再將其導入 TextRank 演算法中，在演算個個詞與的相似度時，除了運算該文章範圍內的詞語，還另外加上詞向量中詞語的權重。下文會再多做說明。

1. 建立詞向量

甲、 整理文本

我們使用先前提到三民出版社的高中教科書作為文本。內容包含歷史、地理、公民三個科目各三本課本，一共九本教科書。由於原始文本為 Word 檔，且當中包含排版、照片、圖表等等元素，無法直接進行使用，因此我們使用 Python 將檔案中的純文字部分抓出，並將九本課本內容合併於一個 txt 檔案當中。

乙、 文本預處理

文本整理完畢，進行文本的預處理。因為這次的文本主要是要輔助計算詞語間的相似度，因此須將文本進行斷詞處理，方便之後訓練。這一次我們使用了中研院研發繁體中文專用的 CkipTagger 斷詞工具，雖然之前用的 jieba 也可以對繁體中文進行斷詞，但由於不是專為繁體中文設計的，在斷詞的準確性上，CkipTagger 還是較 jieba 略勝一籌。

斷詞如上述預處理步驟相同，一樣將文章存成二維陣列，範例如下:

分詞前	小明喜歡吃蘋果。蘋果是一種紅色的水果。
分詞後	[['小明', '喜歡', '吃', '蘋果', '。'], ['蘋果', '是', '一', '種', '紅色', '的', '水果', '。']]

之後為了減少雜訊，去除掉停用詞及標點符號。

2. 使用 Word2vec Skip-gram 模型建立詞向量

使用上述預處理完畢的文本，放進 Word2Vec 的 Skip-gram 模型中進行訓練，將所有詞語以 250 維的向量表示。

3. 導入詞向量

將生成的詞向量導入 TextRank 中，將該句內所有詞語的向量相加並除以該句子的長度，得到該句的平均向量。

4. 計算句子間的相似度

用上述算好的每句向量平均值以 Cosine Similarity 計算相似度，公式如下(A、B 分別代表兩個句子的向量):

$$\frac{A \bullet B}{|A||B|}$$

將相似度儲存於一個大小為(句數×句數)的二維矩陣，接著將此二維矩陣轉換為圖，以句子為節點、相似度為邊。

5. 代入 TextRank 演算法

$$h(V_i)=(1-d)+d \bullet \sum_{V_j} \frac{w_{ji}}{\sum_{V_k \in Out(V_j)} w_{jk}} h(V_j)$$

Vi 為節點，即為句子

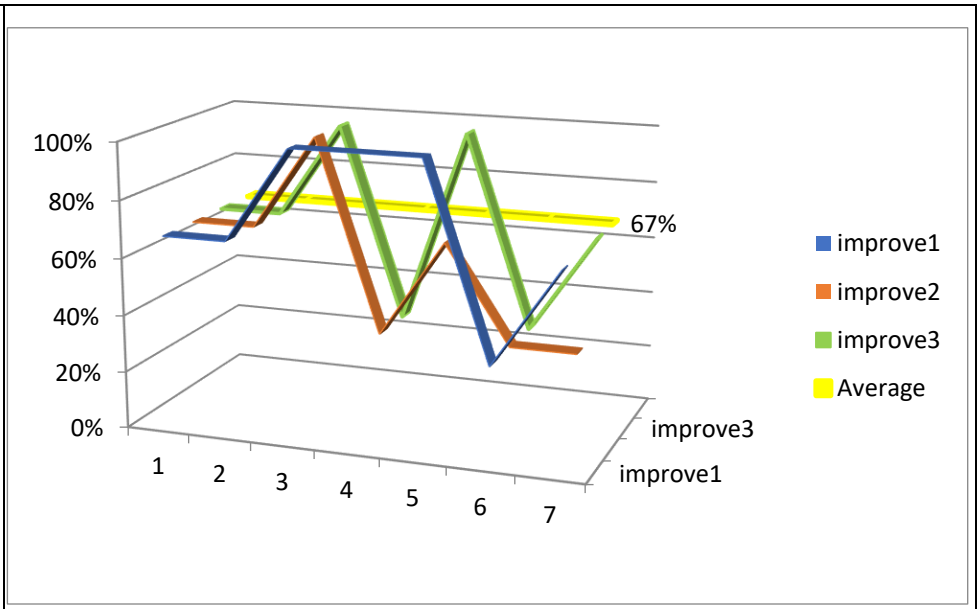
h(Vi)為某個節點的 TextRank 分數

d 為阻尼係數，為定值且介於 0~1 之間，常被設定成 0.85

Wji 為節點之間相連的權重，即為上述的相似度

最後 TextRank 分數最高者即為重點句

6. 結果

經過改良後 TextRank 由三組人員的標準答案算出的評分結果及平均準確率	

由上圖可知，相較於先前的 TextRank，準確率高了將近 10%，也比 TF-IDF 還來的高。因此此改良成功。

參、 研究結果與討論

一、 TF-IDF 算法缺點

TF-IDF 主要依賴計算詞頻來當作判斷依據，但實際上，詞頻不一定代表一個詞與的重要程度，雖然在同主題文章中，有 IDF 數值對詞頻影響力修飾、限制，但在多主題文章或單篇獨立文章中 IDF 就顯然無法起到作用。因此這個方法雖然直觀、方便，在摘要準確率上了的潛力卻有限，若要提高摘要的準確率，應該增加更多變數，分析常見的寫作架構，例如多文章的首句和末句都會包含重點訊息，因此我們可以再句首和句末提高權重。

二、 TextRank 演算法缺點

雖然 TextRank 是依據前後語句的關聯性來找出重點文句，但分詞的方式和文字排列的順序皆會影響摘要結果。例如前面敘述的窗口分組計算，第一個詞只會被計算到一輪，可能會因此被低估它的權重。改善方法之一為將原本的計算方法改成一個完整的循環計算，也就是從句子結尾的詞語再回繞道開頭運算，讓每個詞語被計算到的次數均等。

三種演算法都依賴詞頻為判斷重要程度的依據，而非語義上的關聯，因此難免會有失真的狀況。但是由改良後的演算法可知，若完整考慮文本整體內的詞語關聯，還是能得到不錯的結果。當然，重點的判定因人而異，這部分也是造成準確率波動的影響。另外，由於依賴詞頻，斷詞對於準確率的影響也很大，因此改用 CkipTagger 後，準確率也提高了。

肆、 結論與應用

(一)、 成功使用多種演算法摘取繁體中文文章摘要

(二)、 改良後的演算法比先前準確率高了將近 10%

伍、 參考資料

一、何晗　自然語言處理　2019 年 10 月 1 版　人民郵電出版社　366 頁　2019 年

二、TF-IDF:

https://medium.com/voice-tech-podcast/automatic-extractive-text-summarization-using-tfidf-3fc9a7b26f5

三、Textrank:

https://www.analyticsvidhya.com/blog/2018/11/introduction-text-summarization-textrank-python/

四、NLPCC 2017 Single Document Summarization Dataset:

http://tcci.ccf.org.cn/conference/2017/taskdata.php

五、LSTM:

https://zh.wikipedia.org/wiki/%E9%95%B7%E7%9F%AD%E6%9C%9F%E8%A8%98%E6%86%B6

六、ROUGE score:

https://www.ccs.neu.edu/home/vip/teach/DMcourse/5_topicmodel_summ/notes_slides/What-is-ROUGE.pdf

七、Ranking Sentences for Extractive Summarization with Reinforcement Learninghttps://arxiv.org/pdf/1802.08636.pdf

八、Using semantic folding with TextRank for automatic summarization SIMON KARLSSONhttp://www.diva-portal.org/smash/get/diva2:1116765/FULLTEXT01.pdf

九、Using TF-IDF to Determine Word Relevance in Document Queries https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.121.1424&rep=rep1&type=pdf

十、 Adapted TextRank for Term Extraction: A Generic Method of Improving Automatic Term Extraction Algorithms

【評語】190011

本作品從生活經驗發想，結合電腦科學理論與工具，進行中文重點文句的摘要。本作品展現良好的科學研究方法，採用不同的斷詞工具與詞語分析技術，透過實驗設計，驗證方法的可行性。本作品若能強化文獻探討與相關既有系統（中文或其他語言）的調查，將能更凸顯本研究的特色與貢獻，也能更加提升科學精神的展現。