國立臺灣大學電機資訊學院電信工程學研究所

碩士論文

Graduate Institute of Communication Engineering

College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

去除劇情干擾的動漫與輕小說評論支持度分析系統

A Review Analysis System for Cartoon, Comic, and Light-Novel without Interference of Words Describing the Plot

廖沛俊

Pei-Jun Liao

指導教授：鄭士康 博士

Advisor: Shyh-Kang Jeng, Ph.D.

中華民國 105 年 1 月

Jan, 2016

國立臺灣大學（碩）博士學位論文
口試委員會審定書

去除劇情干擾的動漫與輕小說評論支持度分析系統

A Review Analysis System for Cartoon, Comic, and Light-Novel without Interference of Words Describing the Plot

本論文係廖沛俊君（R02942138）在國立臺灣大學電信工程學研究所完成之碩（博）士學位論文，於民國 105 年 1 月 29 日承下列考試委員審查通過及口試及格，特此證明

口試委員：鄭士康（簽名）

（指導教授）李宏毅　陳信希

所　長　吳家麟（簽名）

致謝

感謝鄭士康教授給了我在選擇題目上極大的自由以及細心指導，鄭老師對於作為指導教授的原則堅持以及引導學生自己去找到方向與資訊的風格給了我很深的影響，也謝謝老師願意特別撥出時間與學生討論論文並指點學生方向。

特別感謝陳信希教授與李宏毅教授百忙之中能擔任口試委員，並在口試前後也給了我許多修改方向的建議，使論文內容更加充實圓滿。

感謝御仁、任瑜、鴻欣三位學長平時 meeting 對我研究的指點以及 Rehearsal 的幫忙，使我口試當天不至於太過緊張。

感謝宗諭學長、惠雯學姐、佳沛、善斌、韋仁、友岳、佩佩等實驗室夥伴在我碩士班期間的一路扶持與幫忙。

感謝十年交情的好友柏聿一路陪伴。

感謝電機系同窗好友飛揚寒星、大魔神、林揚凱、瑪利歐、鄉長、呂棒、小強、王致皓在大學時代課業討論與玩樂的一路相隨。

感謝嘴炮團的庭慶、禹麟、沛恩、舜隍的一路扶持。

感謝大學的好室友黃子洋、吳秋塵、宅民。

感謝少林拳社的游錦鴻老師和王珣學長與社團及北少林長拳門的各位夥伴。

感謝丞淵等其他好友們在我台大生涯中的陪伴與鼓勵。

最後要感謝我的父母弟妹在我的求學生涯中給我的關心與幫忙，感謝父母支撐起這個家，使我可以沒有經濟上的壓力專心求學。

摘要

現在的出版作品，有許多的管道通路可以推廣，而隨著市場的擴大，許多的作家及出版業者也想加入並占有一席之地，知道讀者愛好的趨勢潮流是必須的，網路的作品評論是一個參考的重要指標，許多研究利用關鍵字抽取及句型結構分析等自然語言處理的方法來直接分析評論中支持或反對作品的程度。然而，在劇情類型作品評論判定的過程中時常遇到表示意見的關鍵詞彙並不是用於作品本身評論而是作品中情節之文字描述，為此，本論文在解決此問題上提出使用詞性標註與句型結構樹及文句間關聯性質來做處理，在本篇論文中不同於以往的研究是單純建立意見特徵字典，在本篇論文中我們提出了建立劇情字典來幫助我們將純劇情文句分開以排除作品內容對於意見判斷的干擾而達到改善意見判定正確率的目標。

Abstract

In recent years, there are many author and publishing house want to join the market of cartoon, comic, and novel. So, they may want to find what type of book do readers like. In the Internet, there are so many critics and impressions in it. They may become important indicator to help finding the stream of market. Many researches use key word extraction and analysis of sentence structure. However, in the critics or impressions, we can find the sentence of story discussing or statement in them. It may make us finding the opinion sentence more difficult. In this paper, we use POS tagging and Parsing Tree and relation between nearby sentence to solve this problem. Different from the former work that only had opinion feature word dictionary, we construct a story feature word dictionary to help us taking the story sentence apart. It can improve the accuracy of opinion judgment.

目　錄

第一章 緒論

1.1 研究動機與目的

拜現代科技與資訊流通之賜，愈來愈多的動漫小說作品問世；而如何判斷哪些作品值得繼續投資或擴大市場，是出版商的一個難題。最直覺的評斷標準是在實體書銷量上，而對於未真正有作品出版的市場或者電子商務銷售平台，不容易直接由銷量評估作品在各地區的受歡迎程度。近年網路科技發達，透過海外代購或購買電子版本等手段取得作品內容的許多讀者，往往會在論壇網站或電子佈告欄等地發表對作品的心得及討論。這些也是作品受歡迎程度的重要指標。由於論壇不一定有讀者對作品評分的機制，因此，文字本身的意見探勘技術發展，對於作品評價衡量日趨重要。

目前所知的各種文字意見探勘技術，處理作品評論(包含影評、書評、劇評)時，時常會面臨討論劇情的文字；而這類文字極可能並非對作品本身的評論意見，但卻很容易被意見探勘機器判斷為意見詞，影響作品評論的正負向意見判定。以往的研究，多半未仔細考慮此一因素，直接以意見詞配對特徵詞進行判定。本研究提出：對劇情討論建立屬於自己的詞典，並透過加權去除劇情討論對意見判斷的干擾。

1.2 文獻回顧

意見探勘的研究已經持續了數年，其中不乏從事評論的範疇。2004 Minqing Hu 與 Bing Liu 的研究分析電子產品的評論，並利用 WordNet 詞典與關鍵字，找出句子的意見[1]。2008 年 Ding 等人使用 Amazon 的 3C 產品評論語料，並將句中每一特徵詞算出其意見分數，以分析評論[2]。2013 年 Sohail 等人收集書評中的七種特徵(Occurrence、Helpfulness、Material、Availability、Irrelevancy、Price、Others)，並利用自建的意見詞典分析書評[3]。而在中文方面， 2010 年交大謝鎮宇利用中文句型及 NTUSD 等詞典，分析飯店的評價[4]。2011 年同樣也是交大的邱鴻達，利用同義詞林，建立意見詞與對應之形容對象的特徵配對，實現電影評論系統[5]。 2012 Chien-Liang Liu 等人的研究中，將意見相關的特徵詞，利用 LSA 分析，並利用 SVM 的機器學習方式處理[6]。2015 Chang-Zhi Wang 等人的研究，則是利用微網誌的文章作為語料庫，並利用 CRF Model 對意見與配對的特徵詞分析意見[7]。本論文則以 NTUSD 內分類好的正負意見詞為主，並加入一些人工方法，刪去不適合作為作品評論形容的意見詞，判定意見的正負取向。表 1- 1 和表 1- 2 是先前相關研究的重點整理，並與本研究比較:

表 1-1 各研究比較

	中文			
	[4] 謝鎮宇 2010	[5] 邱鴻達 2011	[7] Chang-Zhi Wang et al. 2015	本研究
外部詞典	BOW, NTUSD ,SINO corpus	同義詞林	HowNet , NTUSD	E-HowNet , NTUSD ,
外部工具	CKIP	OpenNLP	CRF++	Jieba,Niu-Parser
實驗語料	Yahoo 生活飯店評論	Yahoo 電影評論	405 個 micro-blogs	巴哈姆特討論區、輕之國度、PTT 、百度貼吧的動漫輕小評論文章
辨認方法	規則	規則	規則	規則
論文重點	1. 特徵與意見詞庫擴充與比對 2. 使用中文句型計算意見傾向權重 3. 飯店與評論評分計算	1. 考慮副詞與否定語氣 2. 將電影特徵詞分類 3. 電影評分系統	1. CRF Model 對於意見詞及配對特徵詞分析 2. 意見動詞的考慮	1. 針對劇情相關文句作主動剔除，非被動由形容作品特徵詞 2. 意見名詞的考慮 3. 簡易書評系統

表 1- 2 各研究比較(接續表 1-1)

	中文	英文		
	[6] Chien-Liang Liu et al. 2012	[1] Minqing Hu and Bing Liu 2004	[2] Ding et al. 2008	[3] Sohail et al. 2013
外部詞典	SENTIWORDNET	WordNet	WordNet	BabelNet , BOW
外部工具	OpenNLP			Niu-Parser
實驗語料	非特定 Internet Blogs	405 個 micro-blogs	445 篇 3C 產品評論(Amazon)	
辨認方法	規則、學習	規則	規則	規則
論文重點	1. 特徵詞辨識 2. 利用 LSA 對特徵詞作分析 3. 利用 SVM 對正負意見分類	1. 意見詞傾向的判斷 2. 較少見特徵詞的判定	1. 特徵與意見詞庫比對 2. 意見傾向權重計算	1. 七種特徵 2. 權重的計算

1.3 本論文之貢獻

本論文的主要貢獻如下

a. 在與意見詞配對的特徵詞中去除劇情干擾的影響，增進預測意見的 Precision 值。

b. 利用標點符號斷句時的句子關連，找出更多的特徵詞配對，增進預測意見的 Recall 值。

c. 利用兩種不同的句型結構分析方式，對意見詞為動詞\形容詞及名詞的情況，分別進行特徵詞的分析與擷取。

d. 建立作品評論的特徵詞詞典。

e. 實現利用經過測試資料之簡易動漫輕小說作品評論系統。

1.4 論文架構

本篇論文主要在解決劇情影響作品評論的問題，並呈現一簡單之書評系統。章節架構如下:

第一章為緒論與過去研究的整理。第二章介紹自然語言處理與意見探勘的相關技術以及原理。第三章提出新方法、系統架構與設計、以及演算法。第四章介紹所提出系統的實現。第五章則進行模擬實驗，討論並介紹簡單的書評機制與系統。第六章敘述本論文結論，並介紹若干未來研究方向。

第二章 背景知識原理

書評分析是由蒐集到的對某書之評論文章中，抽取所需的資訊，做為採取決策判斷的依據。傳統語音處理需要從語音訊號中以 filter bank 等技術，得到特徵資訊如 MFCC，以便作近一步的分析處理。書評分析中的文字處理，也需要一些機制來取得其對應的特徵。文字的特徵抽取大概有兩種方向：一種不會改變原本的資料型態，人工處理時較有機會判斷出特徵的意義，如關鍵字詞抽取、句型結構分析等；另一方向則是利用機率與統計的方法，將文章變為數據或者高維度向量型態，以便作數值上的分析處理，適合純粹數值演算，不太考慮其意義的機械式系統。本論文主要以不改變資料型態的特徵抽取方法為主。相關原理介紹如後。

2.1 自然語言處理[8]

2.1.1 詞性分析

語言學的研究中，將詞按照類似的語法結構行為或語意類型，分成不同種類。這種詞的類別，一般稱為詞性。最重要的三種詞性為動詞、形容詞、名詞：名詞描述事物或者概念，形容詞描述名詞的狀態屬性，而動詞表達句子中的動作。處理中文時，同一詞彙可能在某一句為某一種詞性，但在另一句又代表了其他的詞性，例如:

我感覺今天天氣很好

真舒服的感覺啊

在「*我感覺今天天氣很好*」句中的「*感覺*」是動詞，而「*真舒服的感覺啊*」中的「*感覺*」則是名詞。這種兼類情況，在語言處理時，需要多加留意。還可以用別的角度把詞分為兩大類：一類是開放類別，例如名詞、動詞、形容詞，這類詞的成員比較大量，且較可能因為時代的變遷而有新詞加入，例如*很夯*。而另一種為功能類別，像是介詞和限定詞等，相較於開放類別詞的成員較少，用法也比較單一。傳統文法詞性分為八類，而語料庫語言學家們為使語言分析更精確，便從八種詞性類別再分為表語形容詞(VA)、"有"動詞(VE)、系動詞(VC)、命名實體(NR)、時間名詞(NT)、其他名詞(NN)、代詞(PN)、DT(限定詞)、CD(基數詞)、 P(介詞)、AD(副詞)、CC(並列副詞)、其他修飾詞(JJ)等。意見判斷主要以 VA、NR、NN、VV、JJ、PN 等字詞分析。

2.1.2 句型結構樹與短語

詞為構成文句的基本要素。但文句並非單純由詞搭配詞性序列等規則串連，而是先組成一些短語，再結合為句子。分析句子時，通常會將句子以結構樹的方式來分析，範例如圖 2-1。

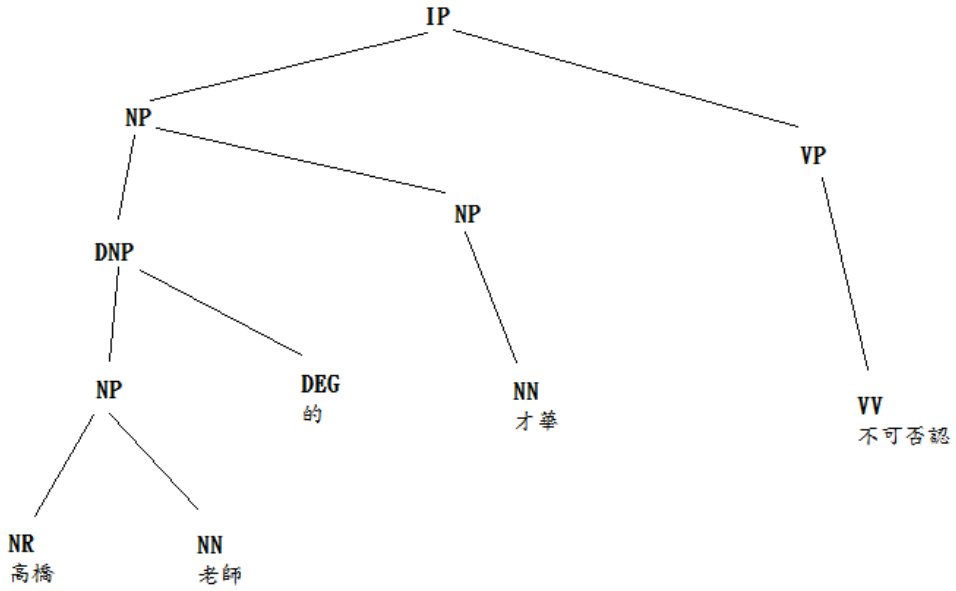

圖 2-1　句型結構樹範例：高橋老師的才華不可否認。

從底層看起： NR 高橋、NN 老師的部分先在結構樹底層組成名詞短語(NP)，再與上一層結合。短語的組成單位可能是由一個或多個詞，甚至是其他的短語，這樣一層一層往上結合，最終可構成一完整句子。

2.1.3 依存關係

句子中的動詞與形容詞，通常有其所憑依或修飾的對象，例如:

我 佩服 作者

這邊「*佩服*」的憑依主格為「*我*」，而憑依受格為「*作者*」，而所憑依或修飾的對象可能扮演主語的角色，也可能作為受語使用。名詞短語(NP)在文句分析中，時常被當作動詞的主語或受語。形容詞的部分，通常討論其所修飾的對象，沒有主受語之分。英文動詞有及物與不及物的分別：及物動詞通常在後面會有受語出現。中文的部分，雖然在本論文所使用工具中，並沒有進一步區分及物與不及物動詞，但其實也有類似用法，例如:

他到達(目的地)了

「*(目的地)*」在口語化的文句中，有時會被省略。本研究為書評分析，對事物作評論時，不及物的情況比較少見，故之後沒有對此點作更近一步的討論。

2.1.4 語義與用法

語義研究是詞語的含意、結構以及應用的方式。研究的主軸可以分為單個詞的意義，以及多個詞組合起來所代表的含意。研究詞義的一條途徑是研究字詞之間是否存在某種連繫的關係。英文的 *WordNet*[9]中，有上位詞與下位詞的關係：上位詞代表一個比較廣泛的含意，而下位詞是特指一項事物或者概念。例如:「*動物(animal)*」是「*狗(dog)*」的上位詞，而「*狗*」是*動物*的下位詞。除了上位詞與下位詞之外，反義詞的也是字詞間的重要連繫關係，例如「*冷*」與「*熱*」、「*正義*」與「*邪惡*」。當然也有同義詞的研究：同義詞可以幫助處理文句中消歧義的部分。此外還有部分與整體關係的雙向討論(例如「*引擎*」與「*車子*」)、以及寫法類似但含意毫不相干的同形異義詞、有相同發音以及相同寫法但毫不相干的同音異義詞等。在 WordNet 中，這些詞彙之間，會以「距離」來定義彼此間的關係。得到了單詞含義後，便需要將其彙整為句子。前述句型結構樹的討論，會將單詞彙整成短語來表達意思。但這其實有困難，因為自然語言在組合意思後，經常不遵守由原單一詞彙的原本意思，例如:

white wine(白酒)　　*(white man)白人*

而事實上白人的膚色並不白，白酒顏色偏黃。，一般英文文句處理上，如果詞語含義與短語含義之間的關係很遠，會將這個短語獨立出來變成一個成語。本論文中因為中文的特性(斷詞問題)，故將所謂的成語，在斷詞階段就先處理，使

其成為一個特定的詞。

句子組成後，下一個大的單位為段落篇章。篇章的研究主要聚焦在文本中句子間隱含的關係。若所分析的文句為談話，許多資訊可能被省略，或者用其他的代稱詞來代表。舉個例子:

A:彼得最近過怎樣

B:他過得還不錯

A:你回去麻煩幫我問聲好

在上述三句中，第二句的「他」代表了第一句的「彼得」，而第三句其實是「*你回去麻煩幫我向彼得問聲好*」的省略。這些文句中的關係，在信息的抽取中是非常重要的。處理上我們會將鄰近的短文視為同一個單位，以方便找到這些被隱含的訊息。

2.1.5 語料庫

自然語言處理除了對文章本身內容作相關的分析外，還有對於文章屬性作更進一步的討論。中研院語言所的平衡語料庫，定義了文類、文體、語式、主題、媒體這五種分類的屬性。文類的部分先分為書面與口語兩大部分：書面類有報導、評論、詩歌等，而口語類則是像劇本、演講、繪畫。文體的部分有記敘、論說、說明、描寫這幾種；語式則分為 written、written-to-be-read、written-to-be-spoken、spoken、spoken-to-be-written 這幾類；而媒體的部分則是使用文本來源(報紙、一般雜誌、視聽媒體等)作為分類標準；主題的部分則是分為哲學、科學、社會、藝術、生活、文學五大類，之後再去作細分。中研院的平衡語料庫研究中，目前是以主題為主訂出內容比例[10]。語料庫中選擇的文本屬性比例，很有可能對分析結果造成很大的影響。例如政府要推出一項新的政策，內容公佈後引起了許多的討論。如果分析群眾意見時，所選的意見來源只限於目前政府所屬執政團隊親近的論壇或網路社群，很容易造成對民眾意見錯誤的判斷；同理，本論文中的書評系統，其關鍵詞分析，若單參考某特定類型作品的評論文章，很容易造成某些類型作品關鍵詞遺漏，造成系統不泛用(只能針對某種特定類型去處理)。所以本研究選用動漫輕小作品的關鍵詞提取時，其文本比例經過一些考量才決定所決定的輕小說與其衍生動漫畫各種屬性比例如表 2- 1 所示。

表 2-1 在本研究中各種元素屬性的動漫輕小說所占比例

作品屬性	奇幻	校園	科幻	貴族	日常	戰鬥	鬥智	穿越
比例	60%	76%	32%	12%	24%	68%	12%	16%

同一部作品可能會有屬性重複的情況，統計時將其重複計算，因此比例總和超過 100%。

2.2 意見探勘

意見探勘的領域中，評論的對象、評論者本身、整體文章意見等的影響，均可進行相當深入的研究。本篇論文中，以評論對象的判斷，以及由單句意見加總，得到的整體文章意見為主。評論的對象一般稱之為特徵詞(Feature)。單句的意見通常會反應在某些詞彙中，這些詞彙稱作意見詞(Opinion Word)。以下介紹特徵詞以及意見詞的應用，並講解將其擷取出來的一些方法。

2.2.1 意見詞分析

意見詞一般將其概括分為正面詞、負面詞以及無意見詞三種。判斷意見詞的意見歸屬一般稱為極性判斷，最常用的方法是利用網路或者其他媒體所得到的文本資訊來判斷 2002 年 *Turney* 提出以網路資料為主，利用不同詞彙共同出現的頻率來推論極性[11]。其想法為：具有類似意見的詞彙，比較常在文章中共同出現。因此，利用改良後的 PMI 來計算新加入意見詞的歸屬。在其研究中，取正向詞彙為"Excellent"，而負向辭彙為"Poor"，經由下列公式來計算:

$$\mathrm{SO(phrase)} = \log_2[\frac{hits(phrase\ NEAR\ \text{"excellent"})hits(\text{"}poor\text{"})}{hits(phrase\ NEAR\ \text{"poor"})hits(\text{"excellent"})}]$$

hits 代表搜尋回傳的網頁數量，而透過此公式計算出來的 SO 數值大於零就是正向意見，反之小於零則是負面意見。在此實驗中，極性的正確率高達 75%，還可以將正負向的意見詞，擴充到十幾個，以增進效能，或使其能應用在不同的語言語系中。

利用語料的部分，最常用的方法是先建構一個基本的意見語料庫(使用人工的標註方法來表示這些詞彙的極性)，之後利用現有的 WordNet、HowNet[12]或者是同義詞林[13]，將基本意見語料庫中的同義詞彙加入意見語料庫的相同意見部分，而具有反義的詞彙則加入相反的部分；反覆進行此部分，將語料庫擴充。研究[1]利用 WordNet 與事先收集的，擁有 30 個極性詞彙之意見語料庫，作為意見詞分類，對於句子的意見極性判斷的實驗正確率為 84%。對於 WordNet 中不存在的詞彙，則無法判斷極性。另一個方法為語意相關場的應用：語意相關場不同於語意相似度，在反映兩個詞語相互的關聯程度時，考量了兩個詞語，在同一個情境共同出現的可能性(概念類似網路資料為主的意見極性判斷)。 2006 朱嫣嵐等人利用 HowNet 結合語意相關場的概念[14]，進行如下實驗:

「假設共有 k 對基準詞，每對基準詞包括一個褒義詞和一個貶義詞。褒義基準詞表示為 key_p，貶義基準詞表示為 key_n，單詞 w 的語意傾向值用 Orientation(w)表示，以 0 作為默認閾值，最終傾向值大於閾值為褒義，小於閾值為貶義。Orientation(w)數值大小代表單詞 w 褒貶強烈程度。單詞 w 的語意傾向值計算公式如下

$$\text{Orientation(w)} = \sum_{i=1}^{k} Similarity(key\_p_i, w) - \sum_{j=1}^{k} Similarity(key\_n_j, w)$$

」

在此實驗中語意相關場計算語意傾向的公式如下

$$\text{Similarity(key, w)} = \frac{|Relavance(w) \cap Relavance(key)|}{|Relavance(w) \cup Relavance(key)|}$$

簡單來說就是將兩個語意概念的交集作為分子，而聯集作為分母，個別計算正負向意見詞的意見極性。此實驗中結合語意相關場與語意相似度的方法，準確率高達 78%。

除了上述方法，若是評論擁有評論者本身的評分，也可以利用這些評分結合該評論中出現的詞彙來建立語料庫。2010 Marneffe 等人便是使用 IMDB 的電影語料產生標註，計算未知極性的意見詞，在個別評分下的文章中所出現的機率，將個別機率乘上評分分數來作極性判斷[15]。此方法受限於分析的文章領域，是否有設計好的評分系統。除此之外，若是評分系統本身沒有受到評論者注意，分類出來的意見詞可能就會發生錯誤。此外還有利用詞彙找出的特徵向量去作分類的方法，在 2009 Yessenov 等人的研究中，把英文評論中的每個字當作特徵向量，而比較極性的部分，選用出現頻率較高的詞、副詞及形容詞，實驗正確率將近 7 成[16]。

2.2.2 特徵詞分析

特徵詞擷取的部分有監督式與非監督式的區別。監督式正確率較高，但需要大量的人工標記，而非監督式的方法雖然正確率較低，但可省下人工標記的步驟。非監督式的學習常採用詞彙的出現頻率為重要的參考指標，以下介紹 TF-IDF[17][18]這個經典方法。

TF-IDF 是一種統計方法，用來分析一個字之於一個文件集，或是一份文件之於一個語料庫的重要程度。其核心概念為：如果某詞或短語在一篇文章出現頻率(TF)相當高，且在其他文章中很少出現(IDF)，就可認為這個詞或短語具有代表性的。TF 的公式如(2-1)。

$$\mathrm{t}f_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \qquad (2\text{-}1)$$

此處$n_{i,j}$ 代表該詞在文件d_j中的出現次數，而分母則是在文件d_j中，所有字詞的出現次數之和。

IDF 為 Inverse Document Frequency(逆向文件頻率)的縮寫，其公式如(2-2)

$$idf_i = \log\frac{|D|}{|\{j:t_i \in d_j\}|} \qquad (2\text{-}2)$$

其中$|D|$ 為語料庫中的文件總數，$|\{j: t_i \in d_j\}|$ 則是包含詞語t_i的文件數目。若是該詞語不在語料庫中，可能導致分母為 0，所以一般使用修正式(2-3)

$$idf_i = \log\frac{|D|}{1+|\{j:t_i \in d_j\}|} \qquad (2\text{-}3)$$

來預防這種情況。

將 TF 與 IDF 結合起來，得到(2-4)。

$$tfidf_{i,j} = tf_{i,j} \times idf_i \qquad (2\text{-}4)$$

找出重要詞語的同時，將常出現的詞語(通常包含一些句法上的常用語)去除，以達到找出關鍵字的目的，除了直接找關鍵字建成詞典外，還可以將這些數值，整合成句子對應的向量，進行其他的處理(例如餘弦相似度)。

通常將句子或文章整合成一向量時，維度大於三。餘弦相似度(cosine similarity)核心概念是兩個高維向量的夾角愈小，這兩個向量愈相似，公式如下:

$$\cos\theta = \frac{\sum_{i=1}^{n}(A_i \times B_i)}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}} = \frac{A \cdot B}{|A| \times |B|}$$

在找出類似文章時，會結合 TF-IDF 與餘弦相似度，演算法大略如下:

(1) 使用 TF-IDF，找出要比對文章的關鍵詞

(2) 每篇文章各選出一些關鍵詞，並計算對文章的詞頻

(3) 生成比對文章的詞頻向量

(4) 計算向量的餘弦相似度

找尋關鍵字，除了 TF-IDF 的方法外，還有 Latent Dirichlet Allocation(LDA)的模型可以使用。LDA 也是一種非監督式的學習技術，主要用以識別較大規模的文檔集或語料庫主題。採用 Bag of Words 的方法，將每一篇文檔視為一個詞頻向量。此法並未考慮詞與詞之間的順序，而採用各主題在一篇文檔中，所組成的機率分布來分析。LDA 的簡單抽取流程如下:

(1) 從 Dirichlet 分布 α 中取樣生成文檔 i 的主題分布θ_i

(2) 從主題分布θ_i中取樣生成文檔 i 第 j 個詞的主題$z_{i,j}$

(3) 從 Dirichlet 分布 β 中取樣生成主題$z_{i,j}$的詞語分布$\phi_{zi,j}$

(4) 從詞語的多項式分布$\phi_{zi,j}$中採樣最終生成詞語$\omega_{i,j}$

整個模型中的可見變量與隱藏變量的機率分布如下:

$$\mathrm{p}(\omega_i, z_i, \theta_i, \Phi|\alpha, \beta) = \prod_{j=1}^{N} p(\theta_i|\alpha)p(z_{i,j}|\theta_i)p(\Phi|\beta)p(\omega_{i,j}|\theta_{z_i,j})$$

在取得所需參數時利用 Gibbs Sampling，參數矩陣公式如下

$$\phi_{k,t} = \frac{(n_k^{(t)}+\beta_t)}{(n_k+\beta_t)} \quad \theta_{m,k} = \frac{(n_m^{(k)}+\alpha_k)}{(n_m+\alpha_k)}$$

ϕ為主題-詞的參數矩陣，θ為文檔-主題參數矩陣。

2.2.3 分析評價方法

我們在意見探勘的評價中經常使用 precision 與 recall 來判斷正確率。precision 為系統選擇出來的答案中，正確者所占之比例；而 recall 為系統所選出來的正確答案，在所有正確答案中，所占之比例，用文氏圖說明如圖 2-2。

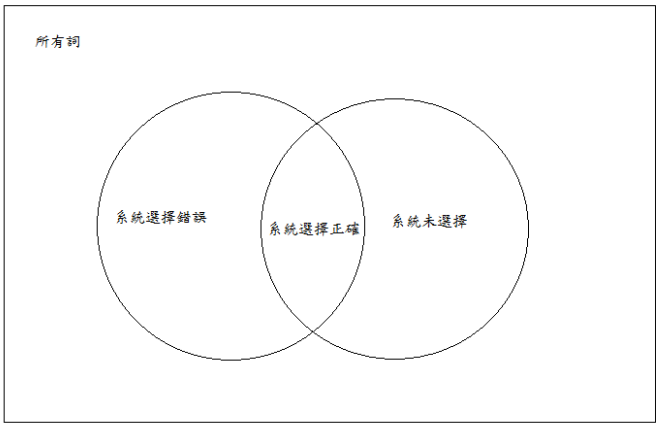

圖 2- 2 Precision&Recall 文氏圖解說範例

公式如下

$$\text{precision} = \frac{\textit{系統選擇正確}}{\textit{系統選擇正確+系統選擇錯誤}} \qquad (2\text{-}5)$$

$$\text{Recall} = \frac{\textit{系統選擇正確}}{\textit{系統選擇正確+系統未選擇}} \qquad (2\text{-}6)$$

有一種常用的評價方法稱為 F-score，結合 precision 與 recall 值來計算，公式如(2-7)。

$$F_\beta = \frac{(\beta^2+1)PR}{\beta^2 P+R} \qquad (2\text{-}7)$$

此處 β 是參數，常用 1 代入(F1-Score)，而 P 為 precision，R 為 Recall。本論文的正確率也是採用 F1-Score 為衡量標準。

第三章 方法及系統架構與設計

3.1 問題探討與解決方法

3.1.1 標點符號問題

與英文在句中，單字之間有間格的結構不同，中文語句的處理上，必須先經過斷句以及斷詞的處理。斷句上標點符號為一重要指標，而楊遠 1962 的研究[19]提到，中國五千篇學生的文章中，僅有 5%的標點符號使用正確。而在所收集的網路文章中，標點符號的問題更是層出不窮。若是直接使用句號或驚嘆號來斷句，可能造成整篇文章被判斷為同一句。而若是直接遇到逗號等標點符號就斷句，又會發生本是同一句的意見詞，因分開而在文句意見判斷上造成錯誤。有鑑於此，標點符號需要特別處理。

本論文採用兩階段的斷句方法，此方法在[5]亦有使用，但本論文在某些標點符號的選用上與[5]稍異：第一階段使用句點、驚嘆號與括號等，作為斷句的依據；第二階段採用逗號與問號及空格來斷句。第一階段結果屬於同一句，但在第二階段被分開的句子，會被記錄下來，作為之後步驟找尋意見詞形容對象之用。

3.1.2 語氣轉折的意見詞選取

文章中常發生在同一句內，出現兩個以上的意見詞的情形。這些意見詞有時候因轉折語氣的連接，而在句中所代表的意見，其實是有所取捨的。例如:

雖然這本書被不少人批評，但有看過書的人，都覺得作者寫得很棒

在此例子中，句子後面的部分，才是偏向本句要表達的意見。若是直接採用意見詞判斷句子極性，容易造成誤判。本論文將意見詞後是否存在轉折詞(但是、不過等)作為參考標準，將轉折語氣後出現的意見詞當作本句的意見詞，而捨去轉折詞前的意見，以解決此情況造成的錯誤。

3.1.3 意見名詞的處理

由於名詞可能會作為主詞或受詞之用，本論文將意見詞分為兩部分:動詞與形容詞和名詞。在名詞的處理上，將中研院系統的 E-HowNet[20]中，帶有表示好壞、意見狀態等之名詞建成詞典，利用此詞典去判斷名詞是否具有意見。

3.1.4 主詞與受詞的判斷

也是將動詞/形容詞與名詞分開：動詞/形容詞的部分採取在句型結構樹中，與意見詞距離最近的名詞短語，當作形容對象，從其下的分支找出主/受詞。距離的定義如圖 3-1 所示

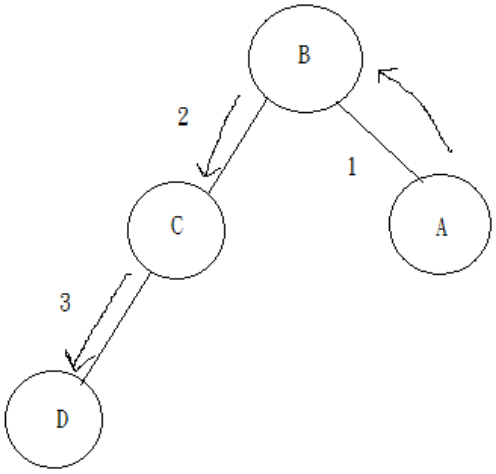

圖 3- 1 句型結構樹中的距離

從節點 A 到節點 D 的最短距離需經過三條連結(A-B、B-C、C-D)，所以 A 到 D 的距離為 3。

名詞的部分則是找到數個(本論文定義為 3)，直接與名詞相連的名詞短語，舉例:

(NP NN(劇情))

來作為形容對象的候選者。

3.1.5 形容詞或動詞找不到主詞或受詞的情況

在這邊主要討論的是主受詞其實是名詞短語的情況。基本上表示意見的詞通常會有一個與其搭配的特徵詞，來表示所形容的對象。有些時候特徵詞並不是單一名詞，而是一個名詞短語所組成。這種名詞短語有可能出現裡面沒有名詞，以至於無法配合特徵詞典的情況，例如:

(IP (NP (CP (IP (VP (VV 吐槽)))
(DEC 的)))
(VP (ADVP (AD 也))
(VP (VP (ADVP (AD 十分))
(VP (VV 卖力)))
(VP (ADVP (AD 十分))
(VP (VA 有趣))))))

在此句中「*(VA 有趣)*」為意見詞，而「*(NP (CP (IP (VP (VV 吐槽)))(DEC 的)))*」為所形容的特徵詞。但在這邊並沒有發現有名詞在內，導致無法從特徵詞典內找出其特徵詞，判斷其是否符合要求。這邊我們將該句最鄰近句的意見(以上一句為優先)，與意見詞對比，如果該意見與本句意見詞的極性接近，則將本句作為意見句，反之則否。

3.1.6 意見詞在該句無描述之對象的情況

由於在斷句時考慮逗號，以至於有些時候，找不到意見詞所形容之對象。本論文 3.1.1 節提到的第一階段斷句，同句紀錄便是用來處理這邊的狀況。此處採用的方法是：如果紀錄中第一階段，同句中的句子，能找到與事先建立的意見特徵詞典中相符的詞彙，則紀錄該句與當前判斷句在文章中的距離(相鄰幾句)。若發現是先前所建立之純作品劇情討論詞典中的詞，則另外紀錄距離。之後分別將兩邊紀錄的距離取倒數後相加，再將意見特徵的結果值減去純劇情討論特徵的結果值。若大於 0 則將當前判斷句作為意見句，反之則否。舉例說明:

美琴的超能力真好用啊

話說劇情也很動人

真是不錯 (當前判斷句)

有意見特徵詞的句子與當前判斷句的距離是 1，而純劇情討論特徵的句子與當

前判斷句的距離是 2，計算的分數為$\frac{1}{1}-\frac{1}{2}=0.5$，此值大於 0，故在這個例子中將當前判斷句加入意見句中。

3.2 系統架構

圖 3-2 是分析作品評論文章的概略圖:

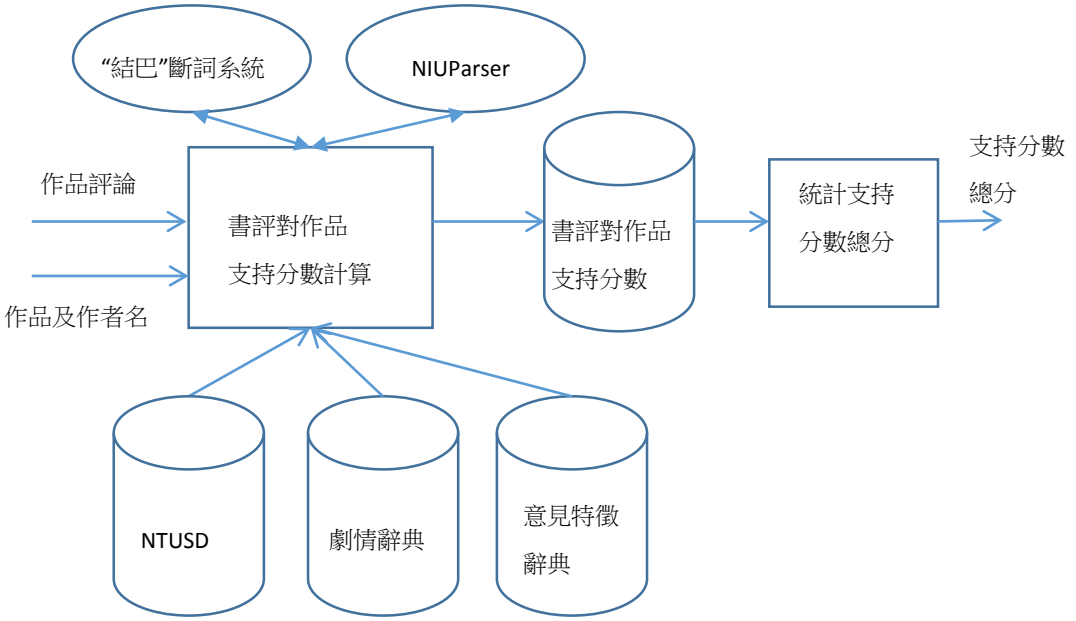

圖 3- 2 評論系統概略圖

輸入的部分是作品評論與該作品名與其作者名，外部支援的工具有結巴斷詞[21]以及 NIUParser[22]。底下有三個支援的詞典：一個是判斷意見用的 NTUSD[23]，其他兩個是用來判斷特徵的特徵詞典。最後統計每一作品的每一篇評論，所得到的支持分數，得到該作品的普遍評價。

詞典的部分除了三個主要詞典之外，還有去除 NTUSD 中不適合作品評論的意見修正詞典、來自中研院 E-HowNet 的判斷意見名詞的詞典、以及轉折語氣詞的詞典。系統使用時，首先將不同作品的評論個別分開，之後將各個作品的評論文章，一則一則放入系統。經過系統判斷後，再將單則評論的分數與該評論儲存起來。待該作品在選用評論資料庫的所有評論文章，全部判斷其極性後，再將其統整，結合作品與作者名稱後輸出。

單則書評系統內部的架構圖如圖 3-3

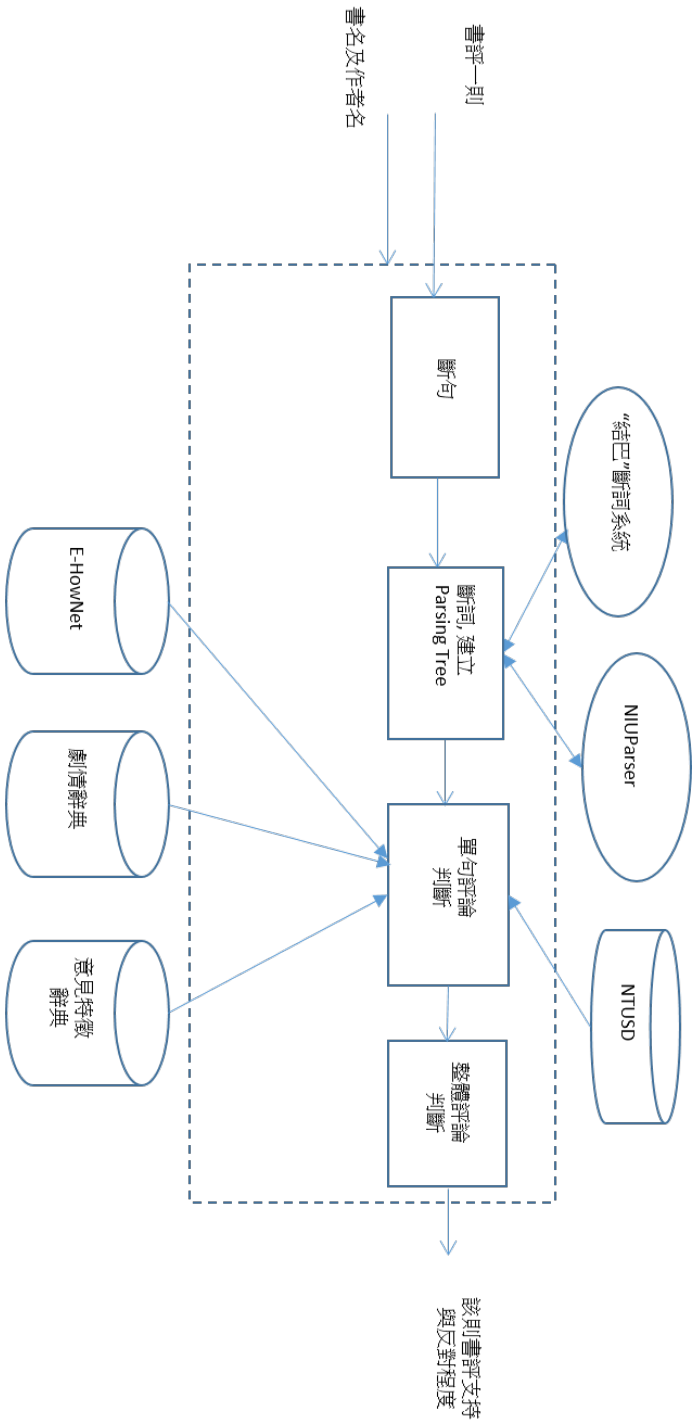

圖 3- 3 評論系統架構圖

將系統內部分成四個部分：斷句、斷詞與 Parsing Tree(句型結構樹)、單句評論判斷、整體評論判斷四個部分。支援詞典主要是在單句的意見判斷時使用，這邊最後輸出為單則評論的判斷。以下介紹各區塊的內部設計與演算法。

3.3 系統設計與演算法

3.3.1 斷句

系統方塊架構如下

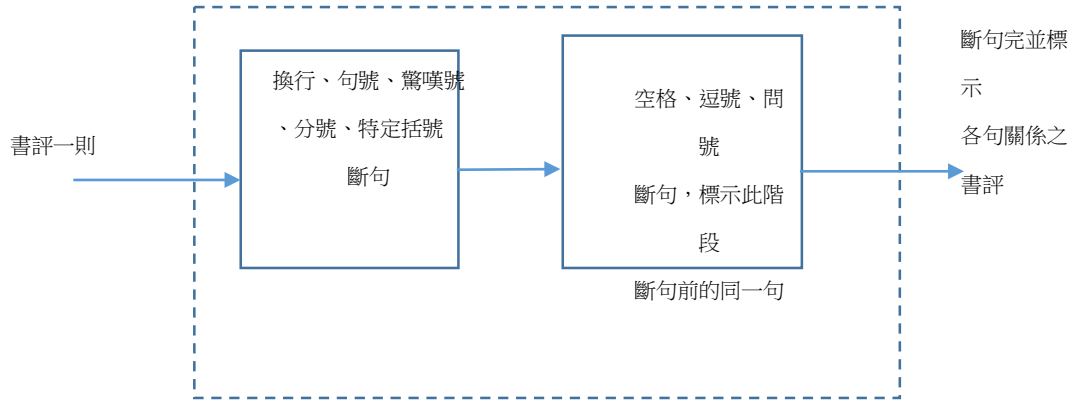

圖 3-4 斷句系統方塊架構圖

在這邊將斷句分為兩階段:第一階段的部分是句號、括號、驚嘆號、分號的斷句，而在原始資料中已經分好的段落句子並不會去作更動，直接作為不同句看待，在括號內句子的部分標註引用的標記；第二階段的部分將逗號、問號、空格作為斷句依據，並紀錄在第一階段時屬於同一句的句子，下圖為虛擬碼型式的演算法

```
Sentence_Seg1(A)
Input : Article A.
Output : Set of segged sentence S_S.
1   For each character in A
2           if find '。' or '!' or '.' or ';'  do Seg_Sentence(A)
3    if find upper in pair of brackets
4       do Seg_Sentence(A)
5       do Seg_Sentence_Set_Quote(A)
6    if find under brackets in pair of brackets
7       do Seg_Sentence(A)
8       do Seg_Sentence_Release_Quote(A)
9   For each line in A
10          S_S ← S_S + current line
11  Return  S_S
```

```
Sentence_Seg2(A_1)
Input : Segged Article A_1.
Output : Set of segged sentence S_S.
1   For each character in A_1
2           if find '，' or '?'  do Seg_Sentence_Set_Relation(A_1)
3   For each line in A_1
4           S_S ← S_S + current line
5   Return  S_S
```

圖 3- 5 兩階段斷句之演算法

3.3.2 斷詞與句型結構樹建立

這個部分使用上一階段斷句的最終輸出當作輸入，先斷詞，得到詞性標註，最後產生句型結構樹。之後作品評論以各句附上句型結構樹型態輸出，設計的內部示意圖如圖 3-6

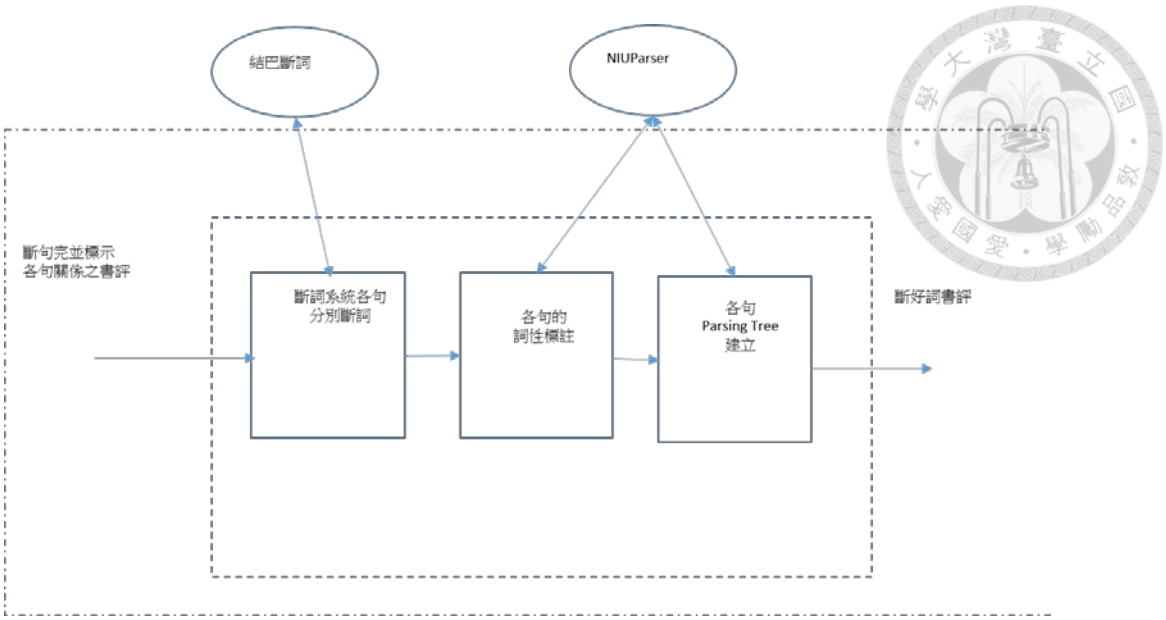

圖 3- 6 斷詞系統架構圖

演算法如圖 3-7

- Word_Seg_Tree(*AS_seg*)
- Input : Segged Article *AS_seg*.
- Output : Set of word segged sentence *S_W*.
- 1 For each sentence *s* in *AS_seg*
- 2 Jieba(*s*)
- 3 For each each sentence *s* in *AS_seg*
- 4 *S_W* ← Jieba(*s*)
- 5 Return *S_W*

圖 3- 7 斷詞系統演算法

3.3.3 單句評論判斷

在此步驟，斷詞之後的句子結構樹一句一句輸入，並作判斷，故稱單句評論判斷。一開始將輸入句型結構樹的文句資料，先去除引用句，接下來利用 NTUSD 決定句子中是否含有意見詞，以及該意見詞所屬詞性(利用先前標註的詞性)，藉以判斷句子的可能極性。若有意見詞存在該句，便將其加入意見判斷的處理句中；若無則直接標示此句為無意見。意見判斷的處理句會先經由 3.1 節判斷所形容的對象是否存在於該句，並儲存。之後依照意見詞詞性，以及形容對象是否存在於當下，分析句子作判斷。圖 3- 8 及圖 3- 9 分別為系統方塊圖與演算法。

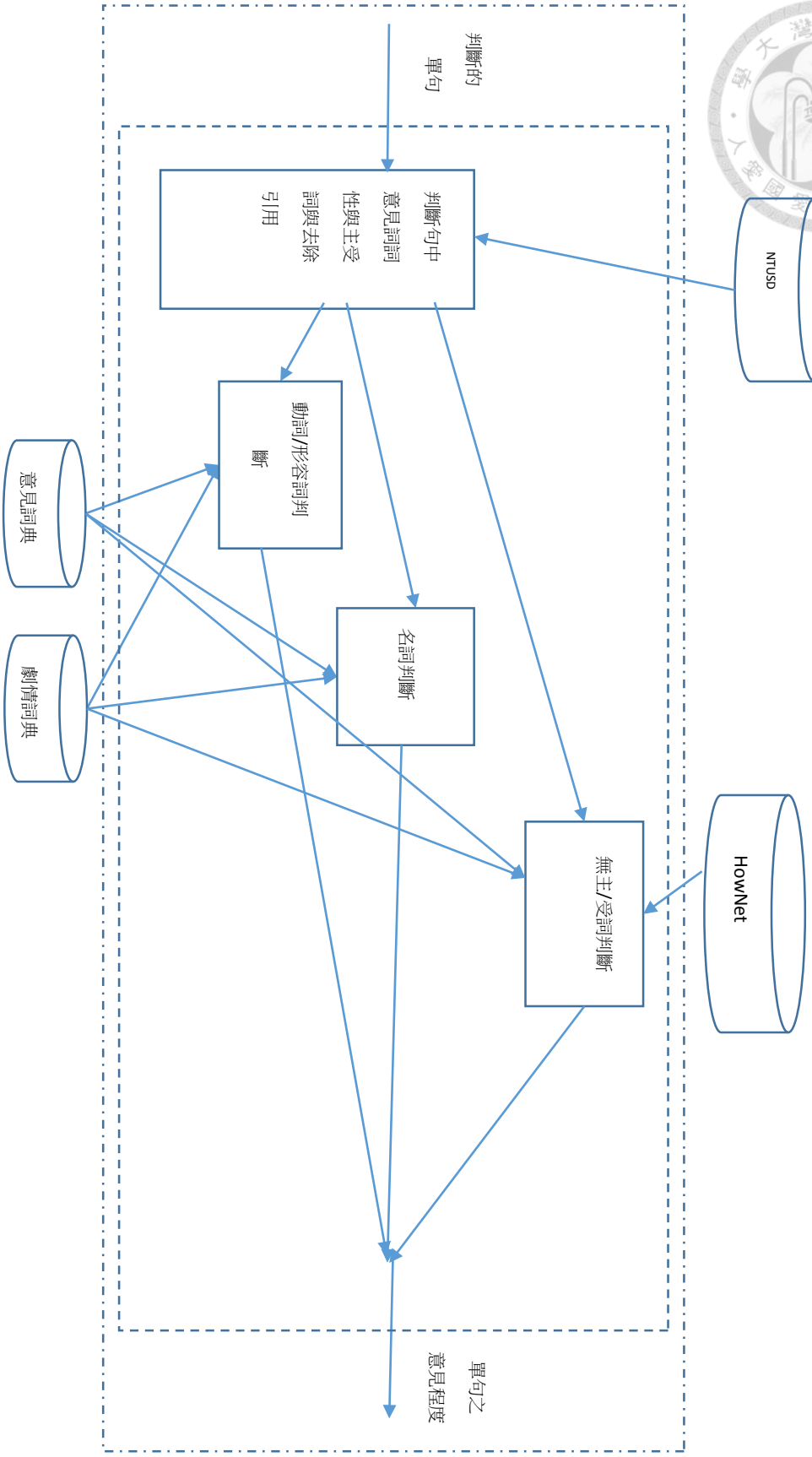

圖 3-8 單句意見判斷之系統方塊架構圖

```
Sentence_Judgment(s,NS)
Input : Current sentence s. Relation Sentence NS.
Dictionary : The dictionary of Subject and Object words that relation to Opinion words in critics C_D. The dictionary of Subject and Object words that relation to story concerningwords in critics S_D. HowNet dictionary H_D. Opinion dictionary O_D.
1   Remove_Quote(s)
2   for each w in s
3       if Find_Opinion_Word(w,O_D) is true   # Check Opinion word
4           part ← Check_Part(w)   # Check Part of Speech of word
5           if part ∈ V/Adj   # Handle Verb and Adjective Opinion word
6               SO ← Find_OS(w, Sub_Ob(w) , NS)
7           V_Adj_Judgment(w,SO,NS)
8           else if part ∈ N   # Handle Noun Opinion word
9           N_Judgment(w, Tree_Structure(s) , NS)
10          else   # Handle word which cannot find its subject or object
11          N_SO_Judgment(w , NS)
12          if Is_Opinion_Word(w) is True do break for
```

圖 3- 9 單句意見判斷演算法

接下來進一步介紹處理動詞/形容詞、名詞以及找不到形容對象的句子的內部架構以及演算法。

動詞/形容詞處理

演算法與系統架構如圖 3- 10 及圖 3- 11。

```
V_Adj_Judgment(W,SO,NS)
Input : Set of the opinion words in current sentence W.  Set of subject and object to the opinion words in current sentence SO. Set of simplest noun-phrases(with nouns in it) in all relation sentence with current sentence. Relation Sentence NS.
Dictionary : The dictionary of Subject and Object words that relation to Opinion words in critics C_D. The dictionary of Subject and Object words that relation to story concerningwords in critics S_D. HowNet dictionary H_D.
Output : Set of opinion sentence OS.
1   for each w ∈W
2       for each so ∈SO(w)      do Noun_Check(so),      V_Adj_SpecialCheck(so)  # Check Special case
3   for each w ∈W
4       for each V_Adj_SpecialCheck(so) id False
5           Tag_Story(SO(w),S_D)  # Tag Story words in SO(w)
7           o_flag ← Find_Critics(SO(w),C_D)  # Find Opinion Subject or Object words in SO(w)
8           Set_Opinion_Flag(w, o_flag)  # Remind that the word is opinion word or not
9       for each V_Adj_SpecialCheck(so) id True and Opinion_HowNet(w,H_D) is True  # Check whether w  is opinion word in HowNet
10      o_flag ← Check_Relation_Opinion(w,OS,NS)  # Check the opinion in relation sentences
11          Set_Opinion_Flag(w, o_flag)  # Remind that the word is opinion word or not
12  for each w ∈W
13      if ∃w such that  Check_Opinion_Flag(w) is True
14          OS ← OS + current sentence   # Put the opinion sentences in OS
15          break for
16  Return OS
```

圖 3- 10 判斷動詞/形容詞之演算法

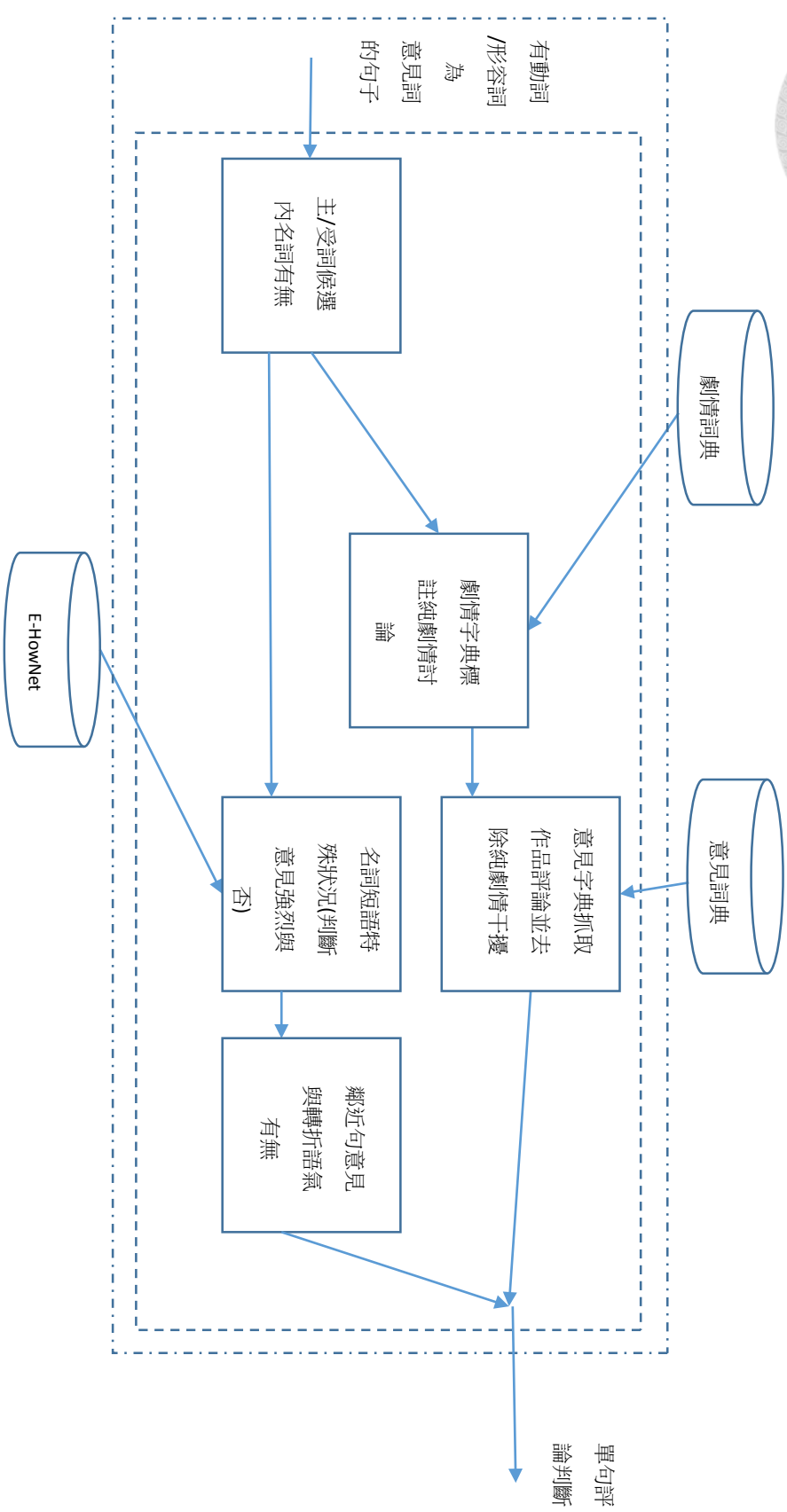

圖 3-11 判斷動詞/形容詞演算法之系統方塊圖

先判斷形容對象的名詞短語內是否有名詞存在：若無則利用 3.1 節敘述的方法處理；如果有名詞存在，則經由純劇情討論的特徵詞典標註純劇情詞，之後經過意見特徵詞典，找出匹配的所有特徵詞，並與之前標註的純劇情詞加權，計算去除劇情討論的判斷干擾。若加權值大於零則紀錄此句為意見句並將意見詞的意見作為該句的意見

名詞處理

演算法與系統方塊圖如圖 3- 12、圖 3- 13。

- N_Judgment(*W*, *TS NS*)
- Input : Set of the opinion words in current sentence *W*. Current Parsing Tree sentence *TS*. Relation Sentence *NS*.
- Dictionary : The dictionary of Subject and Object words that relation to Opinion words in critics *C_D*. The dictionary of Subject and Object words that relation to story concerning words in critics *S_D*. HowNet dictionary *H_D*.
- Output : Set of opinion sentence *OS*.
- 1 for each *w* ∈*W* and Opinion_HowNet(*w*,*H_D*) is True do *CWS*(*w*) ← Find_Relation_Subject(*w*,*TS*) # Check whether *w* is opinion word in HowNet
- 2 for each *w* ∈*W*
- 3 for each *s* ∈*CWS*(*w*)
- 4 Tag_Story(*so*,*S_D*) # Tag Story words in *SO(w)*
- 5 *o_flag* ← Find_Critics(*so*,*C_D*) # Find Opinion Subject or Object words in *SO(w)*
- 6 Set_Opinion_Flag(*w* , *o_flag*) # Check the opinion in relation sentences
- 7 for each *w* ∈*W*
- 8 if ∃*w* such that Check_Opinion_Flag(*w*) is True
- 9 *OS* ← *OS* + current sentence # Put the opinion sentences in *OS*
- 10 break for
- 11 Return *OS*

圖 3- 12 判斷名詞演算法

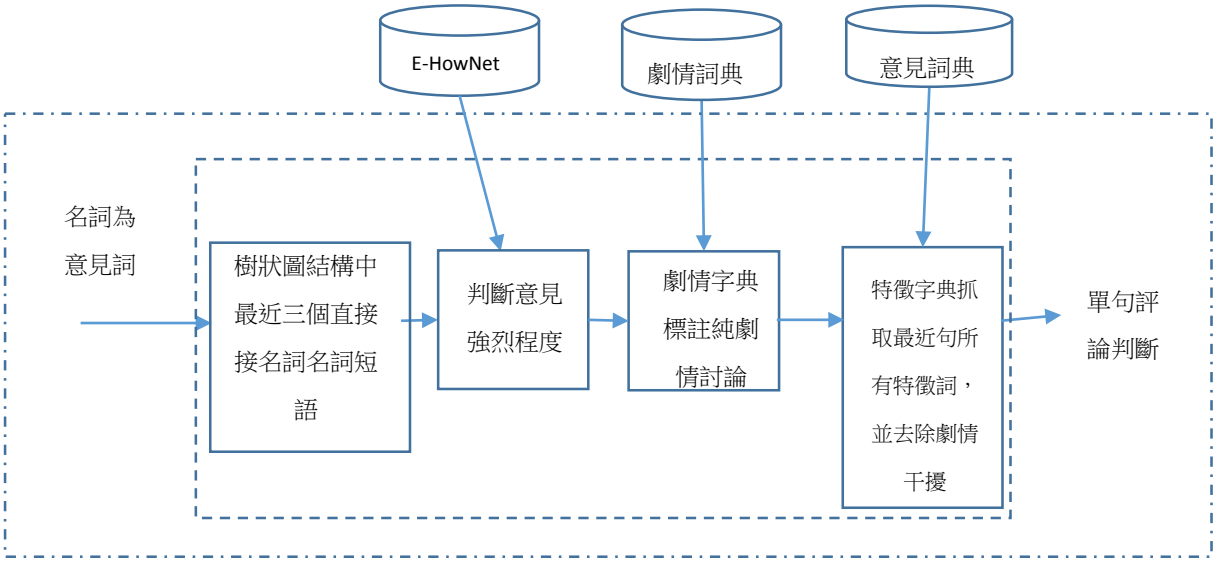

圖 3- 13 判斷名詞演算法之系統架構方塊圖

名詞的部分，先判斷意見詞是否適用為作品評論的，之後標註劇情特徵詞，接著尋找意見特徵詞以及去除純劇情討論干擾。此處採用句中與結構樹中距離最近的意見詞當作特徵詞，最後的步驟與動詞/形容詞的處理相同。

句內無形容對象判斷

演算法與系統架構如圖 3- 14 及圖 3- 15。

- N_SO_Judgment(*W*, *NS*)
- Input : Set of the opinion words in current sentence*W*. Relation Sentence *NS*.
- Dictionary : The dictionary of Subject and Object words that relation to Opinion words in critics *C_D*. The dictionary of Subject and Object words that relation to story concerningwords in critics *S_D*. HowNet dictionary *H_D*.
- Output : Set of opinion sentence *OS*.
- 1 for each $w \in W$ do Opinion_HowNet(*w*,*H_D*) # Check whether *w* is opinion word in HowNet
- 2 for each $w \in W$ and Opinion_HowNet(*w*,*H_D*) is True
- 3 *s_w* ← Find_Score_Story(*S_D*,*NS*) # Find score of Story words in *NS*
-
- 4 *s_c* ← Find_Score_Critics(*C_D*,*NS*) # Find score of Opinion Subject or Object words in *NS*
- 5 *s_t* ← *s_c* - *s_w*
- 6 for each $w \in W$ and $0 < s\_t$
- 7 Set_Opinion_Flag(*o_flag* , *w*) # Remind that the word is opinion word
- 8 for each $w \in W$
- 9 if ∃*w* such that Check_Opinion_Flag(*w*) is True
- 10 *OS* ← *OS* + current sentence # Put the opinion sentences in *OS*
- 11 break for
- 12 Return *OS*

圖 3- 14 該句內無形容對象判斷之演算法

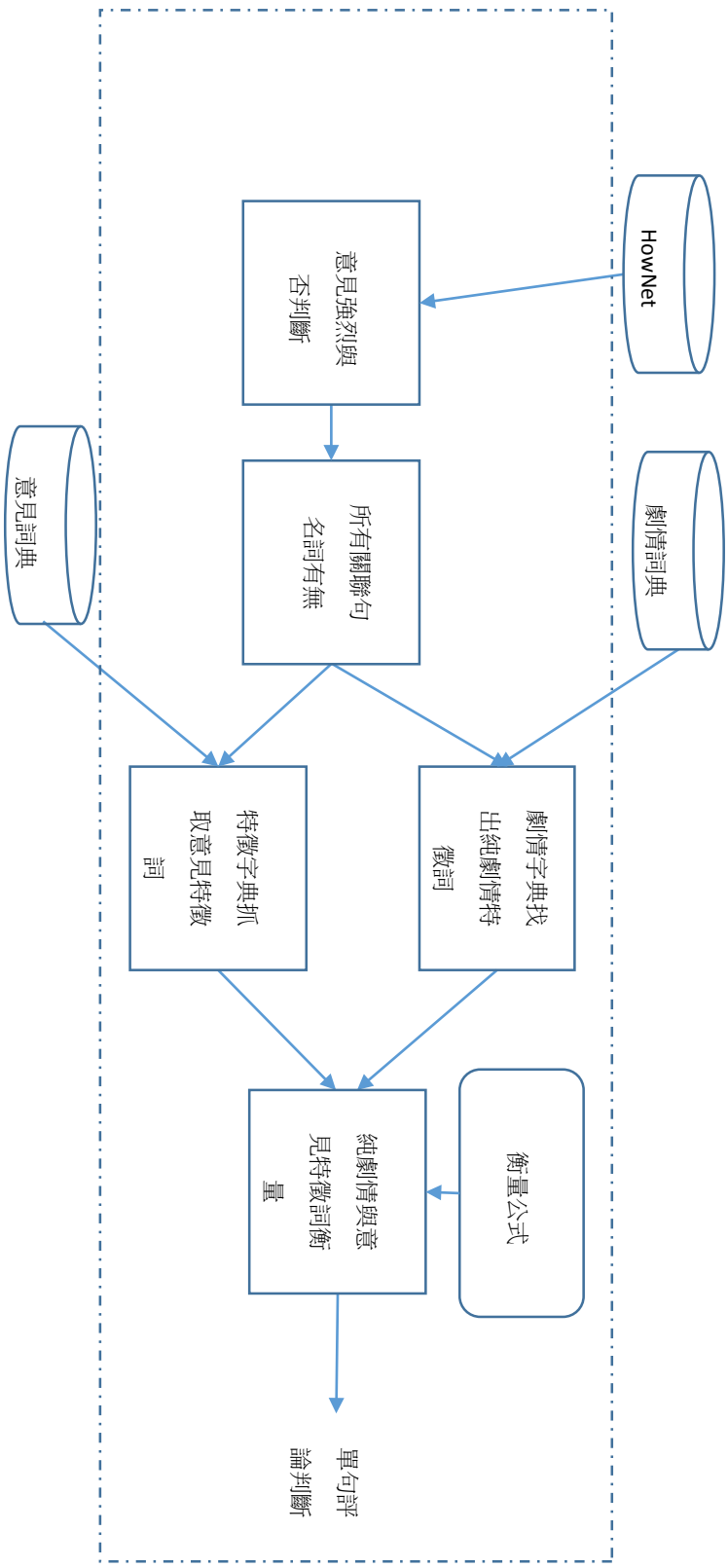

圖 3- 15 該句內無形容對象判斷之系統架構圖

如果意見詞為名詞，一開始會判斷是否為 E-HowNet 中的評論用詞，之後如 3.1 節所敘述，分別將第一階段斷詞，與所得詞在同一句的句子(稱為關聯句)，經由純劇情討論詞典，與意見特徵詞典，找尋是否含有相對應的特徵值，以及與當下判斷句的距離，最後與算出來的衡量分數相減，以得到的值判斷此句是否為意見句(數值須大於 0)。隨後與動詞/形容詞和名詞相同，紀錄此句意見。

3.3.4 整體評論判斷

系統方塊圖如圖 3-16。

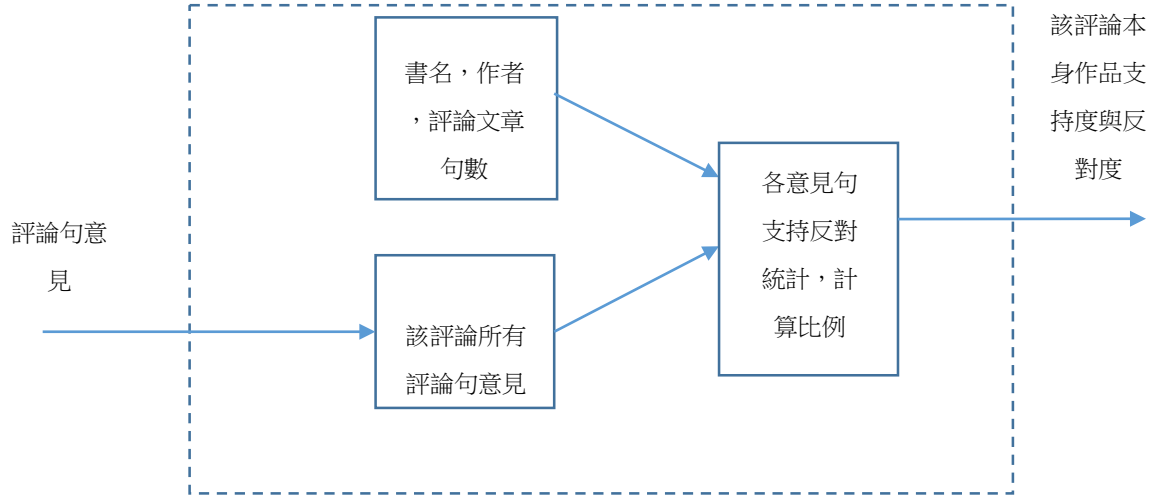

圖 3- 16 整體評論判斷系統方塊圖

這個步驟比較單純，將各句意見統整，並統計各句意見，加總得到評論意見。總句數比較多的評論，會給與較高的加權值。這邊因為步驟較簡單，故不列出演算法。

3.3.5 作品評論判斷

有了各作品的評論判斷後，先將加權的所有評論支持度分數加總，得到作品的評分等第。

第四章 系統實現

4.1 總體實現

以下是整體系統的流程圖(圖 4-1)與說明演算法(圖 4-2)。

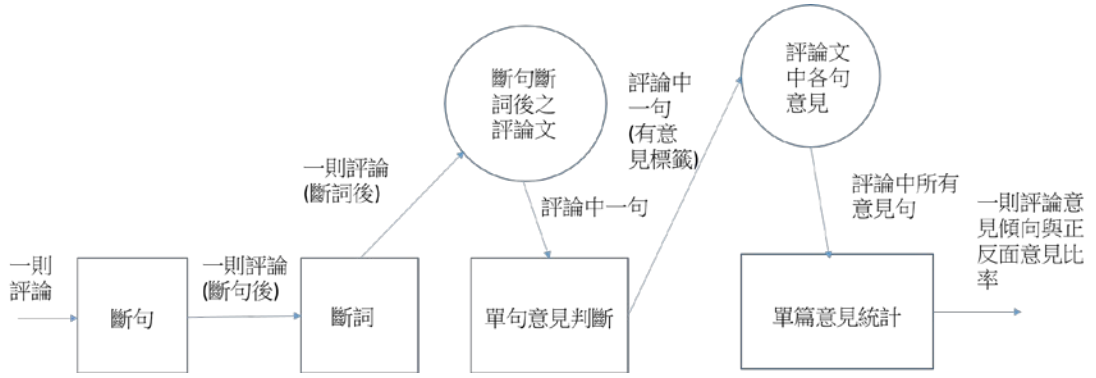

圖 4- 1 系統流程圖

- Opinion_Judgment_Machine(*A*, *AN*, *BN*)
- Input : Crtitics article *A* , Author name *AN* , Book name *BN*
- Output : Opnion answer *OA*
- 1 Put article *A* in Block_0 to get segged sentences *S_S2*
- 2 Put article *S_S2* in Block_1 to get sentences with segged Words and Parsing Tree structure *S_W* # Manually
- 3 for each line with Parsing Tree structure *l* in *S_W*
- 4 Put *l* in Block_2 to get opinion in each and put it to *OS*
- 5 put *OS* , *AN* , *BN* to Block_3 to get *OA*
- 6 Output *OA*

圖 4- 2 系統演算法

再簡述一下整體的流程(詳細部分已經於第三章說明)：一則評論進到系統，經過斷句、斷詞以及建立結構樹後，在暫存區中；接著將評論的各個句子分別取出，依序放入單句意見判斷，將各句判斷後的意見作為標籤，附註在各句後；判斷完整篇評論的所有句子後，統計正面意見與反面意見，若正面意見句多於反面意見句，則判斷評論為正向，反之則判斷評論為負向，若正反意見句數目相同，則判斷為沒有意見。本系統各個區塊圖之間，以手動的方式來接合(在斷

詞的部分，斷詞與結構樹的部分也是透過手動的方式整合)，以便於在不同系統環境下的整合。斷詞方面，如第三章敘述，使用結巴斷詞，加上自定義的名詞處理。由於中文與英文在結構等有不小的差異，中文詞性與句型標註，目前沒有很成熟的系統。現今能使用的系統，大多應用類似處理英文的方法，略作改變。本研究因時間與資源的限制，無法實現以中文結構來標註詞性與句型的系統，因此採用中國大陸東北大學的 NIUParser，標註詞性，並得到句型的結構樹狀圖。

本論文系統中，最關鍵的部分為單句意見判斷與字典，關於這些部分的實現會在下方說明。

4.2 單句意見判斷與字典

4.2.1 句型結構樹的讀取

由於單句意見判斷需要讀取句型結構樹，而在 NiuParser 處理後輸出的結構樹格式如下

(IP

(NP (NN 人物)

(NN 刻劃))

(VP (ADVP (AD 也))

(ADVP (AD 很))

(VP (VV 生動))))

本系統將句型結構樹以自定義 Tree 的類別型態儲存起來，底層為斷好的各個詞與其詞性標註，而 NP、VP 等短語為底層的 parent，IP 又為更上一層的 parent。以上述句子為例(圖 4-3)：

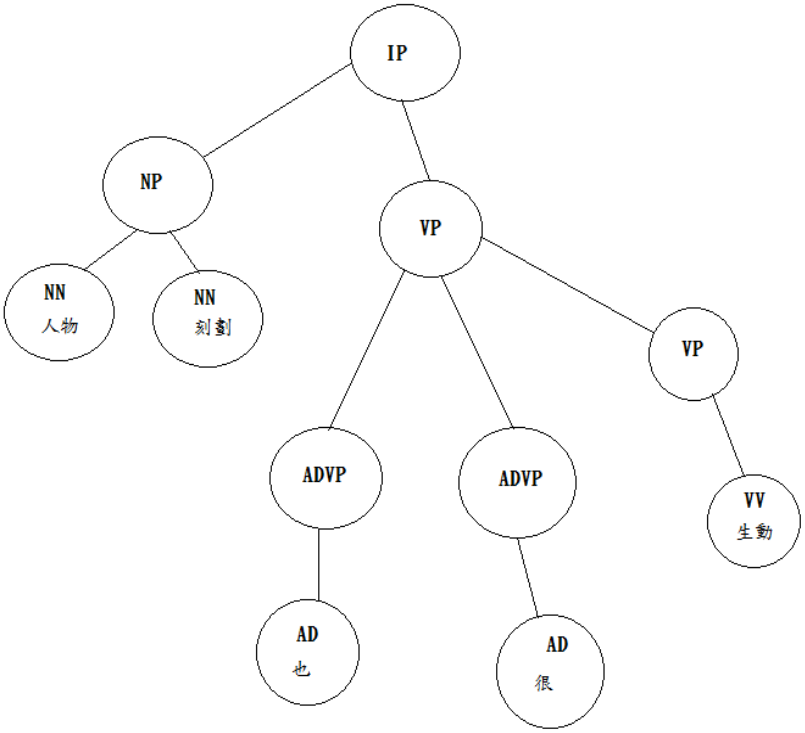

圖 4-3 句型結構樹實例

底層的 Node 儲存詞以及詞性，而上層的 parent node 則儲存短語之類的類別。最上層的 Node 標註為 Root。讀取句子並建立 Parse Tree 時，採用由上而下的方法：遇到左括弧"("時，Tree 就往下一層，而遇到右括弧")"時，便往上一層。因此在程式實作中，遇到括號"("時，就建立新的 Node，並以之為 Child，連結到目前這一個 Node，而目前指標也改指到這個新的 Child Node；遇到括號")"就將目前指標改到目前 Node 的 parent，其演算法如圖 4-4。

```
• Parsing_Tree (S_POS)
• Input: Sentence with part of speech S_POS.
• 1 Tree_Initial(T)
• 2 S ← Get_Tokens(S_POS)
• 3 for each s ∈ S
• 4   if s is '('
• 5             n ← Bulid_New_Leaf(Word,POS)
• 6             Link_New_Leaf(n,T)
• 7             Update_Tree_Itr(T)
• 8   else if s is ')'
• 9             To_Parent(T)
• 10            Update_Tree_Itr(T)
• 11 Return T
```

圖 4- 4 讀取結構樹演算法

4.2.2 動詞與形容詞的句型結構樹判斷實現

第三章提到單句意見判斷可以分成動詞與形容詞、名詞、該句中無特徵詞等三部分。動詞與形容詞的部分如先前所述，找出離該意見詞最近的 NP (該 NP 不得包含目前指定的意見詞)。流程說明為先將目前 Node 標註一個旗標，再將目前指標改到目前 Node 的 parent；接著在現在指標所指 Node 的所有 Child 中，找尋在 Tree 中 ，與之前紀錄的 Node 最接近(第三章有 Tree 中兩 Node 的距離的定義)之含有 NP 的 Node(不包含紀錄"已經過"的 Node 本身)。若是找到具有 NP 的 Node，則將該 Node 儲存後停止搜尋的流程。若是所有的 Child 都沒有具有 NP 的 Node，則紀錄目前的 Node 為已經過(traverse)，並將目前指標再改到目前指標 Node 的 Parent，並重複以上步驟，直到找到具有 NP 的 Node 為止。找尋 NP 的流程圖(圖 4-5)與演算法虛擬碼(圖 4-6)如下。

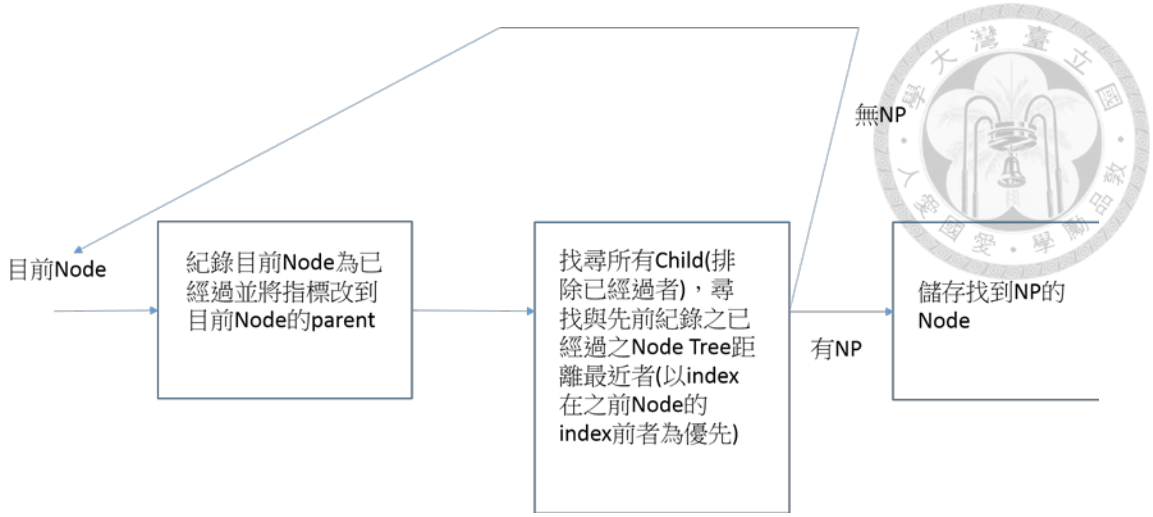

圖 4- 5 動詞&形容詞意見找尋主詞/受詞流程圖

- Find_NP_VAdj(Node,Sentence,NP_Flag)
- Input: Node in Sentence Parsing Tree , Current Sentence , The Flag that remind find NP
- 1 Set_PathFlag(Node)
- 2 Node <- Parent of Node
- 3 for each child c of Node without PathFlag
- 4 if the header of c is NP
- 5 Store_NP(c,Sentence)
- 6 Set_True(NP_Flag)
- 7 break for
- 8 if NP_Flag is True
- 9 stop Find_NP_Vadj
- 10 else
- 11 Find_NP_VAdj(Node,Sentence,NP_Flag)

圖 4- 6 動詞&形容詞意見找尋主詞/受詞演算法

從演算法虛擬碼可以看出：找尋 NP 和名詞，都是使用遞迴來處理。

找到 NP 之後(無 NP 的情況，判斷為該句內無特徵詞)，便在 NP 中找尋其在 Tree 結構中底部的所有名詞(找不到名詞的情況，在第三章已敘述處理方法)。先設置一個特徵分數為 0，若找到符合**意見**特徵詞的詞則加上一個正分，若是找到符合**劇情**特徵詞的詞則加上一個負分(本論文中每次所加正分會稍大於每次所加負分的絕對值)。找完所有 NP 之 Node 下，Tree 之所有底層詞後，將特徵分數大於 0 者視為特徵句，並標示在該句中。流程圖(圖 4-7)與虛擬碼演算法(圖 4-8)如下所示。

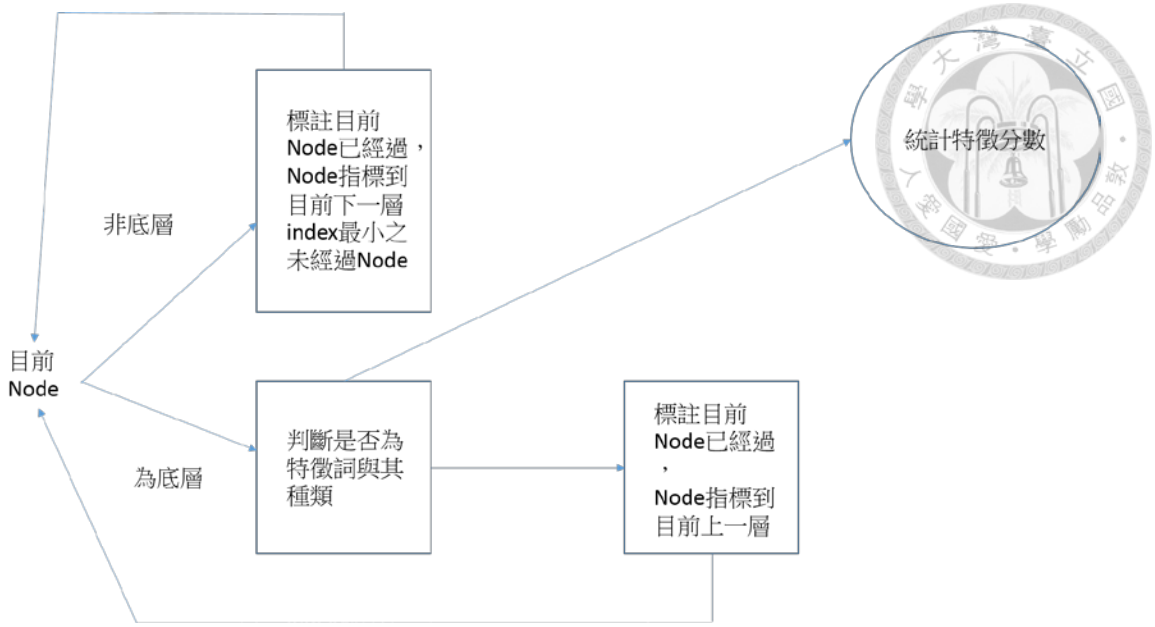

圖 4- 7 動詞&形容詞意見判斷流程圖

- Find_SO_VAdj(Node,Sentence)
- Input: Node in Sentence Parsing Tree , Current Sentence
- 1 if Is_Bottom(Node)
- 2 Set_PathFlag(Node)
- 3 Check_Feature(Node,Sentence)
- 4 Node <- Parent of Node
- 5 else
- 6 Set_PathFlag(Node)
- 7 Node <- The Child with smallest index of Node which do not have PathFlag
- 8 Find_SO_Vadj(Node)

圖 4- 8 動詞&形容詞意見判斷演算法

動詞與形容詞的找尋特徵詞部分，也使用遞迴實現，接下說明名詞的特徵詞找尋。

4.2.2 名詞的句型結構樹判斷實現

名詞的部分在特徵詞的處理上，與動詞及形容詞不太相同。動詞與形容詞直接使用離意見詞在 Tree 中距離最近的 NP，底下所有的詞來找尋特徵詞。而名詞的處理方式，是先找出數個(在本論文是三個)在 Tree 中離意見詞距離最近的 NP，但在這邊的 NP，必須直接連至底層的詞(在第三章有較詳盡的說明)。流程中，首先紀錄目前的 Node 為已經過，接著將目前指標指向目前 Node 的 Parent，標示目前 Node 為已經過，接著變更目前指標為底下 index 最小之未經過的 Child，檢查其所有 Child。若所有 Child 皆為底層(發現的底層皆標示已經過)，則將此 Node 儲存在一個 Set 中。接著將目前指標指向目前 Node 的 Parent，若發現有非底層，且未經過的 Child(發現的底層皆標示已經過)，則將目前指標指向底下 index 最小之未經過的 Child，繼續重複以上步驟，直到找到所有 Child 皆為底層的 Node 為止。若是目前 Node 的所有 Child 皆為已經過，則將目前指標指向目前 Node 的 Parent，標示目前 Node 為已經過，之後重複以上搜尋動作，直到所有底層都為已經過為止。之後將 Set 中所有的 NP，依照 Tree 中與意見詞的距離排序。流程圖(圖 4-9)與演算法(圖 4-10)如下。

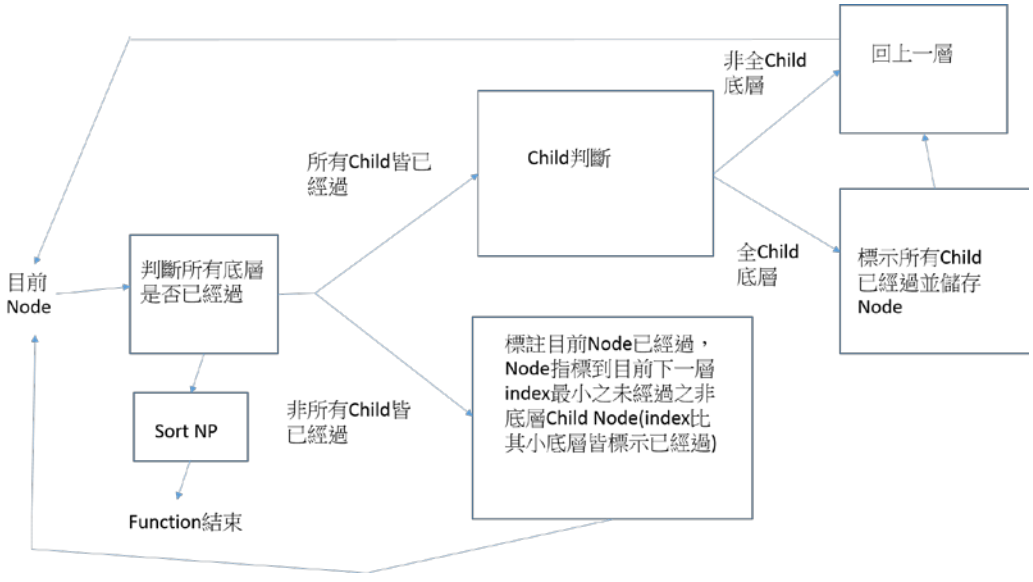

圖 4- 9 名詞意見找尋主詞/受詞流程圖

```
• Find_NP_N(Node,Sentence)
• Input: Node in Sentence Parsing Tree , Current Sentence
• 1  if Check_AllBottom_PathFlag is true
• 2       Sort_NP(Sentence)
• 3       Stop Find_SO_N
• 4  if Check_AllChild_PathFlag is true
• 5       if Check_AllChild_Bottom is true
• 6                 Set_PathFlag(Node)
• 7                 Store_NP(Node, Sentence)
• 8                 Node <- Parent of Node
• 9  else
• 10      for each child in Node without is true
• 11                if child is Bottom
• 12                          Set_PathFlag(child)
• 13                else
• 14                          Node <- child
• 15                          Find_SO_N(Node,Sentence)
```

圖 4- 10 名詞意見找尋主詞/受詞演算法

挑出數個(本論文中為三個)與意見詞句離最近之 NP，接著檢查其中所有的底層詞。接著與動詞相同，先設置一個分數值，分別加減劇情特徵詞與意見特徵詞的分數，若數值結果大於零則判斷其為意見句。流程圖(圖 4-11)與演算法(圖 4-12)如下。

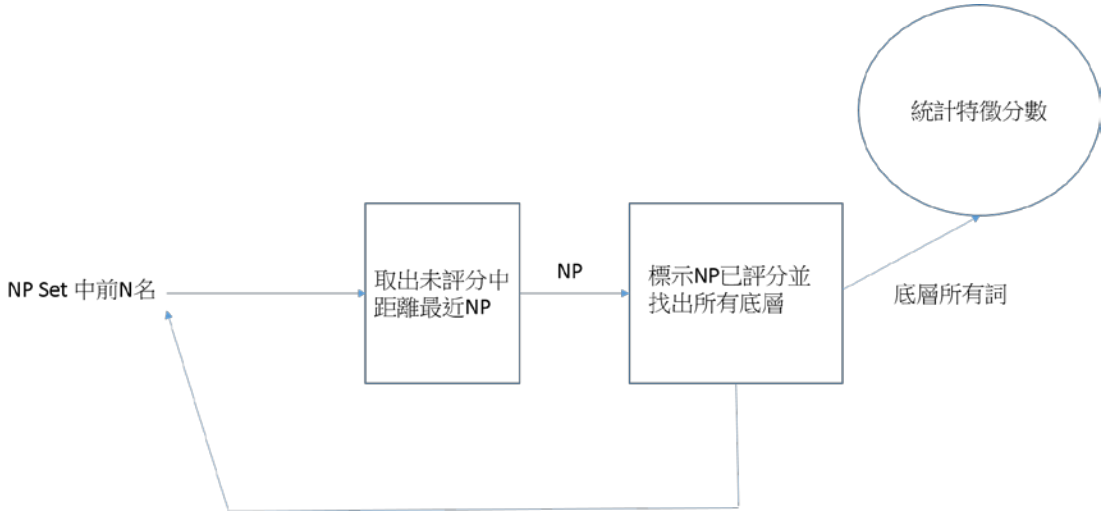

圖 4- 11 名詞意見判斷流程圖

- Find_SO_N(Best_N_Set,Sentence)
- Input: Best N NP, Current Sentence
- 1 for each NP in Best_N_Set without Score
- 2 for each Word in NP
- 3 Check_Feature(Word of Node,Sentence)
- 4 Score(NP)

圖 4- 12 名詞意見判斷演算法

以上為名詞尋找特徵的部分，判斷完意見的句子會先儲存起來。為因應動詞與形容詞的特殊情況(第三章所述)，處理完初步判斷後，會再進行特殊情況的判斷，已在第三章詳述。

4.2.3 句內無特徵詞之判斷的實現

如第三章所述，句中無特徵詞者，需依賴鄰近句子來輔助判斷其意見。而鄰近的句子則採用第三章所定義之關連句：首先給予各意見特徵詞及劇情特徵詞一個初始分數 0。衡量標準是距離(斷完句後文中的距離)該需要判斷的句子 N 句的句子中，如果出現一個意見特徵詞，則在意見特徵詞加$\frac{1}{N}$的分數(因為本句沒有特徵詞，所以並不會有分母為 0 的情況出現)。若出現一個劇情特徵詞，也加上$\frac{1}{N}$。之後將意見特徵詞分數減去劇情特徵詞分數，若結果大於 0 則判斷該句屬於意見句。流程圖(圖 4-13)與演算法(圖 4-14)如下所示。

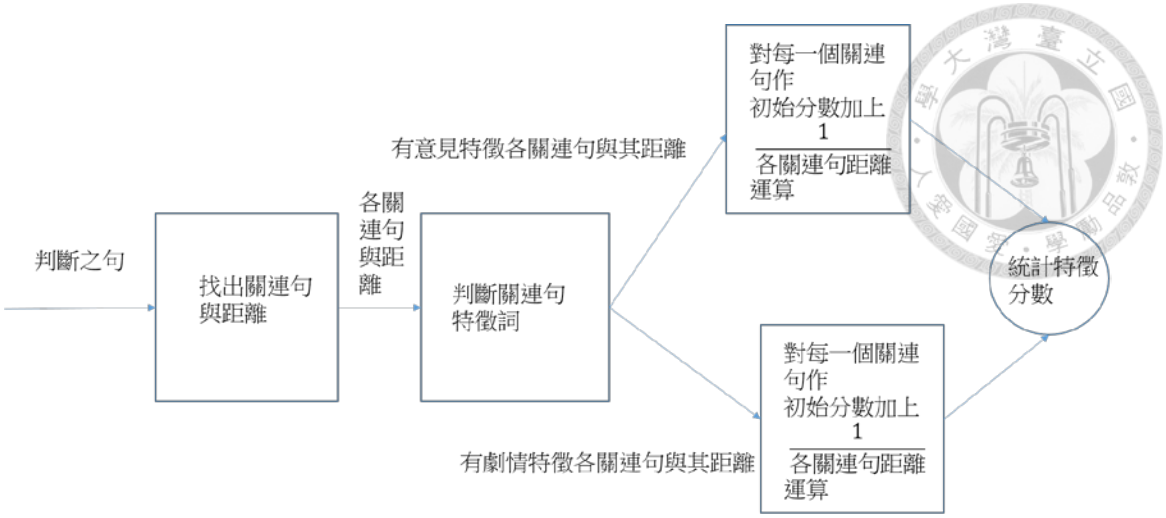

圖 4- 13 該句內無特徵詞意見判斷流程圖

- Find_NSO(Sentence)
- Input: Current Sentence
- 1 for each Relation Sentence r of Sentence
- 2 if Opinion_Feature(r) is true
- 3 for each Opinion Feature Word in r
- 4 Add_Score_OpinionFeature(1/distance(r,Sentence))
- 5 if Story_Feature(r) is true
- 6 for each Story Feature Word in r
- 7 Add_Score_StoryFeature(1/distance(r,Sentence))
- 8 Check_Feature(OpinionFeature, StoryFeature)

圖 4- 14 該句內無特徵詞意見判斷演算法

本系統中，句中無特徵詞的判斷是在動詞、形容詞、名詞的判斷結束之後，才開始進行。

4.2.4 三種意見句情況比較

表 4-1 判斷意見句之三種 Case 比較表

意見句情況	動詞&形容詞	名詞	無特徵詞
找 NP	找一個	找 N 個	無
找特徵詞	NP 內所有加權	N 個 NP 所有加權	鄰近句加權
加權分數配比	意見特徵較高	意見特徵較高	兩種特徵相同，以文字距離決定
遞迴使用有無	找 NP 與特徵詞皆有	找 NP 有，特徵詞無	無
意見詞詞性	動詞、形容詞	名詞	所有
系統中順序	1	2	3

表 4-1 為三種意見句判斷方式的比較。在動詞&形容詞、名詞的實現上，需要使用 Parsing Tree 的結構，而無特徵詞的情況則不必。而無特徵詞處理需用到鄰近句，動詞&形容詞處理只有在考慮特殊情況時需要鄰近句意見。此外，無特徵詞的部分。

4.2.5 字典建立

本論文主要採取字典比對的方式，實現意見句的判斷。在判斷意見詞的部分，使用了陳信希教授實驗室的 NTUSD。 NTUSD 中有些分類好的意見詞，其實並不適合用於作品評論的判斷。例如 NTUSD 中”覺得”這個詞被分為正向，但其實在作品評論判斷上，這個詞並沒有意見表示的成分。但這個詞在作品評論中時常被用到，可能會造成意見判斷上的錯誤，故使用手動的方式將原本的 NTUSD 縮減，並加入簡體中文的部分，結果為正面詞與負面詞各為 2600、3200 個左右的意見詞典。除了 NTUSD 之外，另外收集了 300 篇作品評論的文章(非實驗所用)，使用結巴斷詞的 TF-IDF 工具，配合手動篩選，得出正面詞與負面詞分別為 58 與 51 個(與之前的 NTUSD 簡化字典無重複)。

除了意見詞的部分外，特徵詞的部分也建立了相對應的詞典。同樣使用結巴斷詞的 TF-IDF 工具，配合手動篩選及簡體中文，加入所得特徵詞。之後意見特徵詞的部分，再配合中研院的 E-HowNet，進行同義詞擴充，結果分別為 614 與 576 個特徵詞彙。

為處理轉折語氣與否定詞，本研究也加入簡易轉折語氣(“但是”、”不過”等)以及否定詞(“不是”、”並非”等)的簡易詞典。否定詞的判斷規則為與意見詞在文句中距離為 5 個詞之內即算否定(考慮雙重否定)。轉折詞的部分則為出現在目前判斷意見詞後面，則不考慮目前判斷的意見詞(以轉折語氣後出現的意見詞為主)。

建立以上所需的程式與詞典後，透過以上流程與詞典，取得所需判斷之評論文章的意見句，並判斷其意見極性。正反意見句的數目與最後判斷的評論文章意見結果等，都會儲存起來，以便後續實驗或書評系統應用。

第五章 模擬實驗與討論

本章先就判斷作品評論的系統進行實驗並討論結果，之後會簡單示範一個利用判斷完成含有意見傾向標註的評論書評系統。

5.1.1 實驗語料收集

系統製作時，使用從百度貼吧與 PTT 上所收集的語料，作為抽取關鍵詞之用。本研究的實驗語料來源中，繁體中文為《巴哈姆特》網站以及電子佈告欄《PTT》的作品評論，簡體中文則採取《輕之國度》論壇小說測評區以及《百度貼吧》的文章，圖 5-1 到圖 5-7 是文章來源與收集結果的舉例圖。

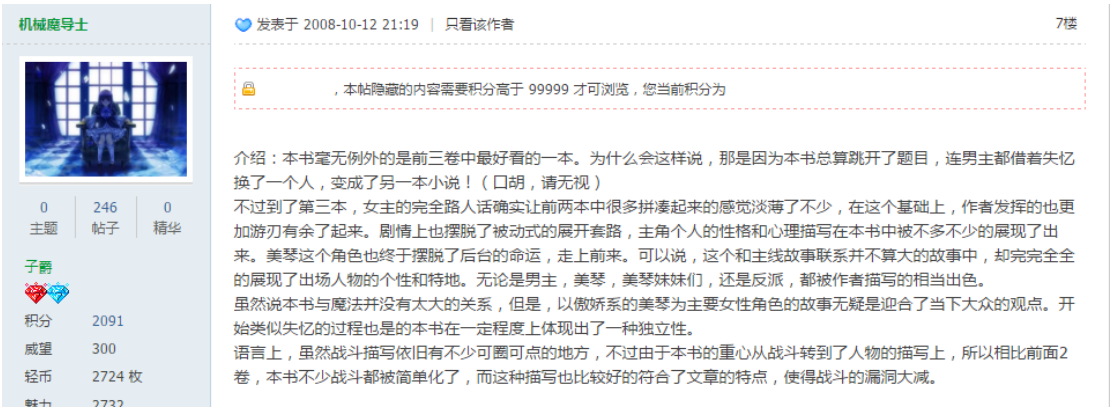

圖 5-1 《輕之國度》評論文章

圖 5-2《巴哈姆特》評論文章

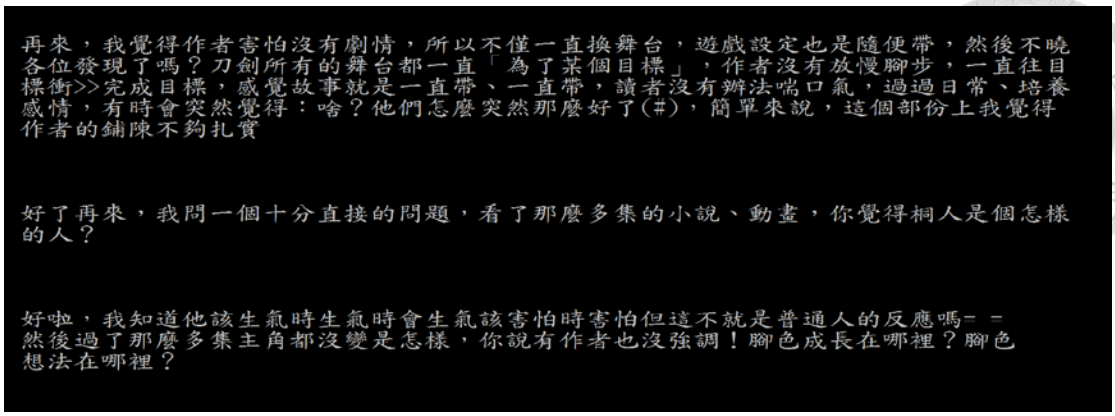

圖 5-3《PTT》評論文章

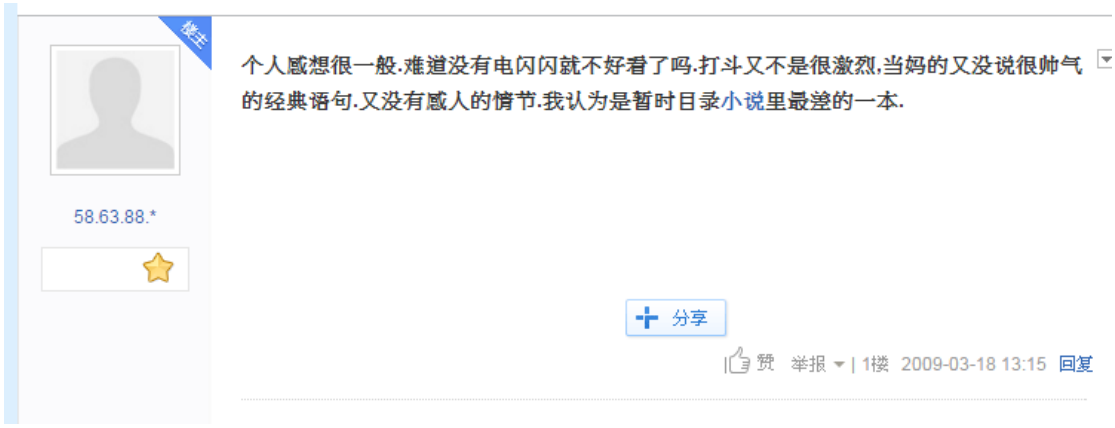

圖 5-4《百度貼吧》評論文章

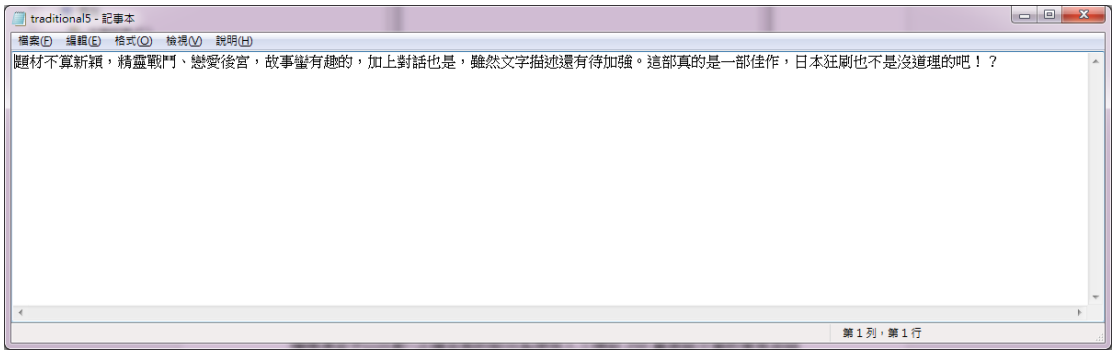

圖 5-5 繁體評論文章範例

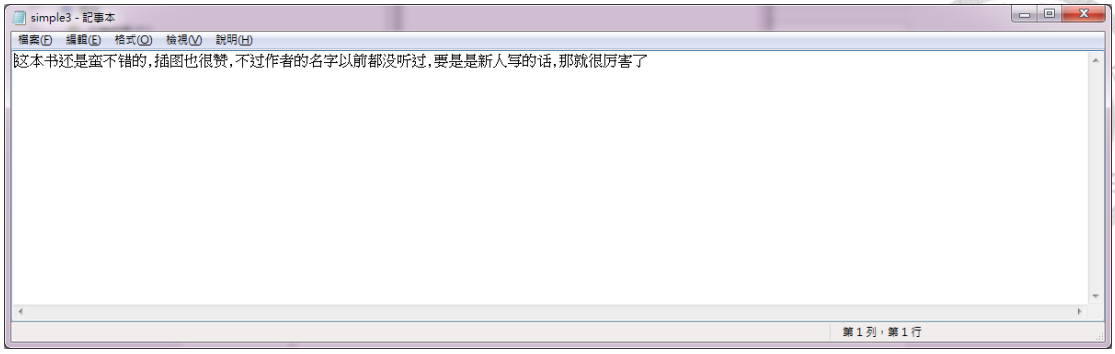

圖 5-6 簡體評論文章範例

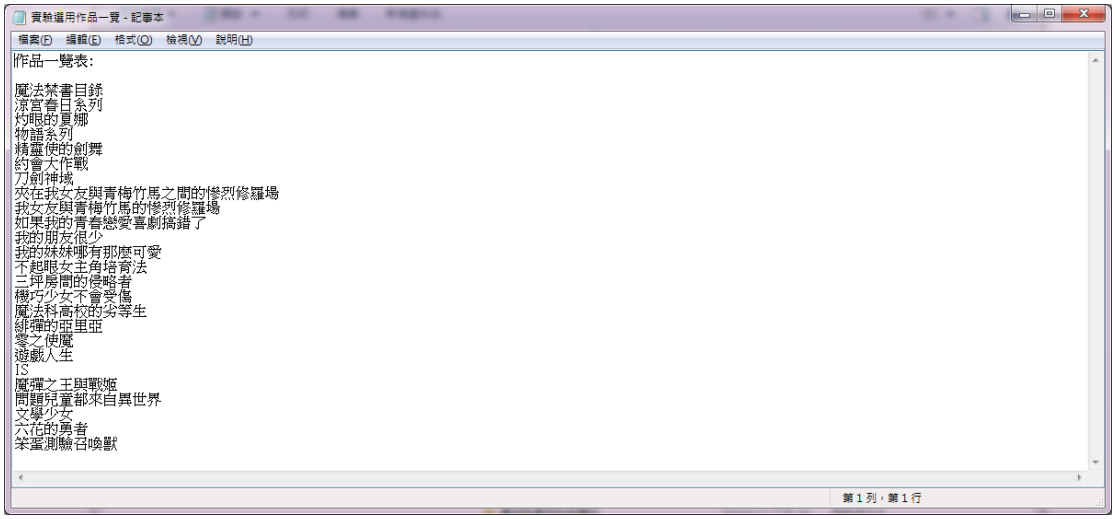

圖 5-7 實驗選用動漫輕小說作品一覽

本研究選用了 25 部輕小說作品，與包含其改編之動漫畫的評論為實驗語料。每部作品在繁體中文與簡體中文的評論各選擇 10 篇，全部的評論共 500 篇。

5.1.2 比較標準

第二章結尾提到 F-Score (2-5、2-6、2-7)，結合 Precision 與 Recall，作為準確性的衡量。本論文採用 Accuracy 與 F-Score 的分數以及 ROC space 圖作為標準，並予以比較。正確答案(ground truth)為三位同學人工標記 500 篇測試評論文章的意見，並投票決定作為判斷。衡量方法如(5-1)

$$\text{Opinion} = \text{Voting}\big(\text{文章人工判斷 } 1, \text{文章人工判斷 } 2, \text{文章人工判斷 } 3\big) \quad (5\text{-}1)$$

Ground truth 的判斷結果有 134 篇評論為負面，剩下皆為正向意見。表 5- 1 是比重百分率之整理表。

表 5- 1 測試用資料之對照答案各意見所占比例

意見	+	-
所占比例	73.2%	26.8%

為了使正負評論的數量達到平均以利於分析，在這邊選用了正負評論各 125 篇來作為實驗測試之用。單句部分，因為系統主要偏重評論文章最後結果的正反意見，故沒有對各句意見判斷作準確率之衡量。若在一篇評論文章中，正負評論句的數目相等，則依照正負詞典中詞彙數量的比例來給予不同的機率後以此機率來隨機判斷此評論為正向或負向。在本論文實驗中，對照組與本論文的方法皆使用同一隨機結果以排除因不同時間點下隨機值不同對於比較造成干擾。

F-Socre 的公式如第二章敘述

5.1.3 實驗結果與討論

本研究使用兩組資料，作為本論文系統方法的對照組，對照組一為不考慮關連句意見的數據(完全不考慮劇情詞典)，對照組二為直接使用關連句中是否有意見詞典之詞彙存在來判斷關連句意見，結果如表 5- 2 與表 5- 3。

表 5- 2 本論文系統實驗之結果

分類方法	評論意見(評論篇數)	Ground truth 正向	Ground truth 負向
對照組一	系統判斷 正向	71	5
	系統判斷 負向	3	68
對照組二	系統判斷 正向	79	7
	系統判斷 負向	5	76
本論文	系統判斷 正向	82	4
	系統判斷 負向	2	79

表 5- 3 實驗結果比較

組別	對照組一	對照組二	本論文
Precision	94.6%	92.8%	96.4%
Recall	57.4%	65.1%	66.0%
F1-Score	71.4%	76.5%	78.3%

由以上實驗結果，可知本論文在 Precision 與 Recall 表現皆較對照組為優。可以得出濾掉劇情討論的干擾後，單篇評論的意見誤判情形隨之減少，因此 Precision 與 Recall(Recall 主要因意見誤判為相反的情況減少而得到進步)皆上昇。在 F1-Score 的部分本論文相較對照組二進步些微進步 1.8%。

本論文的意見詞在以上實驗只單純考慮句中第一個出現的意見詞(轉折語氣已考慮)，接下來的實驗考慮句中所有的意見詞並與先前的方法作比較。

表 5- 4 本論文系統意見詞考慮實驗之結果

分類方法	評論意見(評論篇數)	Ground truth 正向	Ground truth 負向
本論文方法 只考慮第一個意見詞	系統判斷 正向	82	4
	系統判斷 負向	2	79
本論文方法 考慮句中所有意見詞	系統判斷 正向	83	4
	系統判斷 負向	1	79

表 5- 5 本論文系統意見詞考慮實驗結果比較

組別	本論文方法 只考慮第一個意見詞	本論文方法 考慮句中所有意見詞
Precision	96.4%	97.0%
Recall	66.0%	66.1%
F1-Score	78.3%	78.6%

由以上實驗結果顯示，兩種方法的表現幾乎沒有差異，可能的原因為相同意見的詞若除去轉折語氣影響有較大的機率出現在文本中相近的區域(2002 年 *Turney* [11])，以至於考慮句中所有意見詞並不會造成太多的效能改善。

除了劇情造成之雜訊之外，尚有其他造成準確率下降之原因，接下來會予以討論。

5.1.4 實驗其他問題討論

以下討論造成無法經由本論文方法篩去的錯誤之可能原因:

1. 詞典涵蓋的不足

本論文採取詞典比對，而意見詞與特徵詞必須滿足在詞典中同時被發現的條件，才能計算為意見。這個部分可能造成一些未在詞典中出現的意見詞或特徵詞被忽略，影響準確率。

2. 斷詞效果問題

本論文使用結巴斷詞執行 Word 分割的工作。若斷詞出現不符合我們期望的結果，可能會影響到意見詞的正確率，例如:

這本書真給力啊

期望=>*這本 書 真 給力 啊*

斷詞後=>*這本 書 真給 力 啊*

"給力"為出現在意見詞典中的詞彙，但如果斷詞錯誤，很有可能造成找不到意見詞，或選取到與原句整體意思相反意見詞的情況，造成錯誤。

3. 詞性標註問題

本論文使用 NIUParser 進行詞性標註，而詞性標註可能會出現錯誤(部分是因為斷詞的影響)，例如:

一堆/AD 設定/VV 也/AD 挺/AD 認真/VA 的/DEC

在本句中的”設定”其實指的是故事或角色設定，屬於名詞，而 NIUParser 的部分判斷為動詞 VV。本論文中的特徵詞，只考慮名詞的部分(依照正常的結果，名詞比較適合做為動詞與形容詞的特徵詞或者是作意見名詞的敘述對象)。若是詞性本身已經標示錯誤，特徵詞判定上自然也會受到影響。

對於三種錯誤的部分目前設想了一些解決方法:

1. 詞典涵蓋的不足&斷詞效果問題

擴充動漫輕小說作品的人名與常用詞彙到斷詞詞典，並給予較高的權重值。其缺點在於需要涵蓋大量的作品，才會實際。

2. 詞性標註問題

目前詢問中國大陸東北大學的結果,NIUParser 無法讓使用者自行調整詞性標註。解決此問題最好的方法，是根據中文的句型結構分析方法，設計自己的詞性標註器。

5.2 作品評論系統展示

本論文的最後，利用評論判斷系統，自動得出 500 篇作品評論傾向(為先前收集之 500 篇評論)，實現了一個作品評論系統。由於受限於中文字編碼的問題，故作品評論的文字，無法直接顯現於程式執行視窗中，所以利用呼叫該文檔的方式來顯示評論內容之文字。執行程式中，會顯示該作品評論意見的總和傾向比例與單一評論的意見傾向比例，圖 5-8 為系統使用流程示意圖

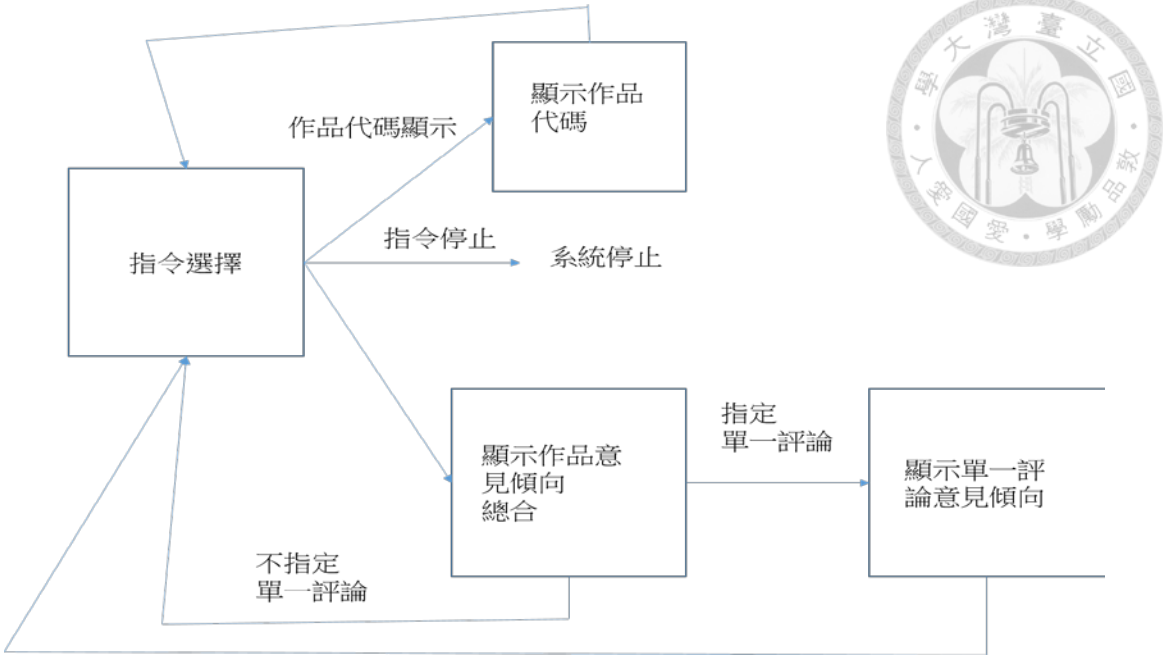

圖 5-8 作品評論系統流程圖

接下來，圖 5-9 到圖 5-13 展示系統的操作與結果。

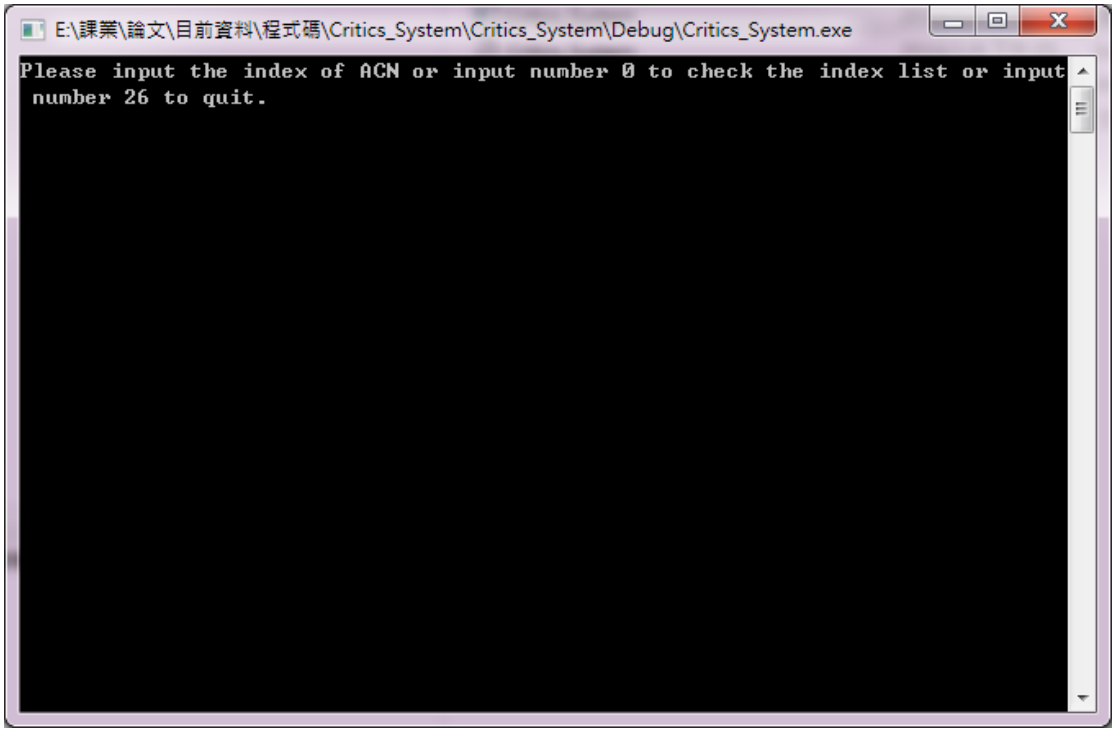

圖 5-9 初始使用介面

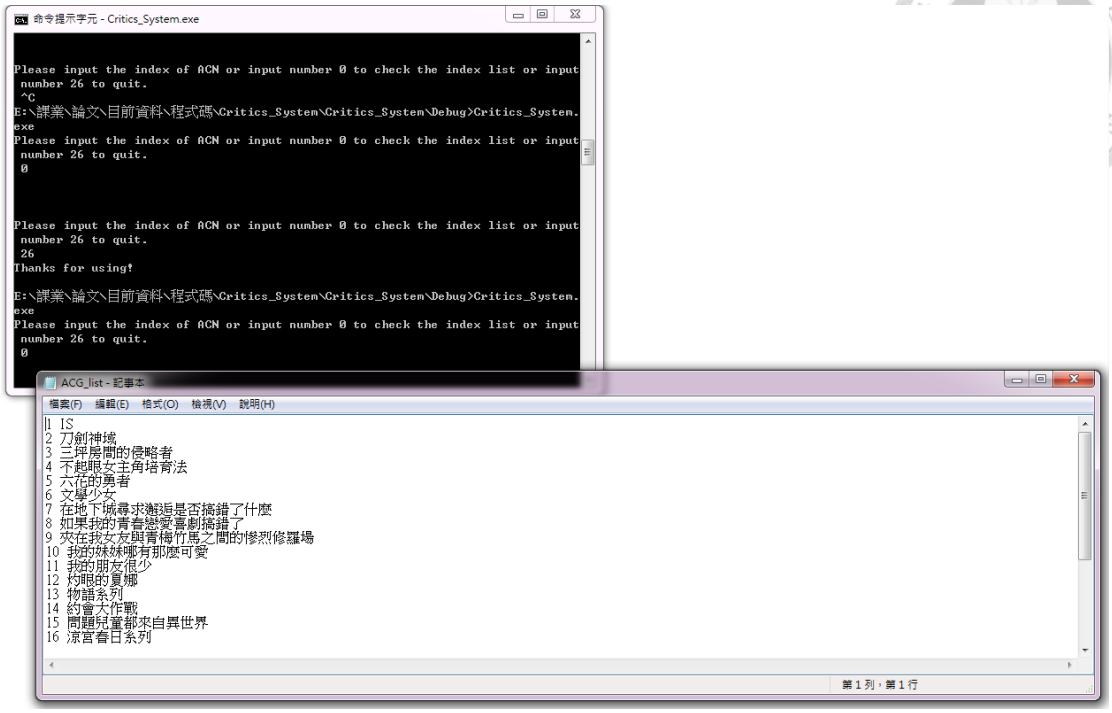

圖 5-10 顯示作品表

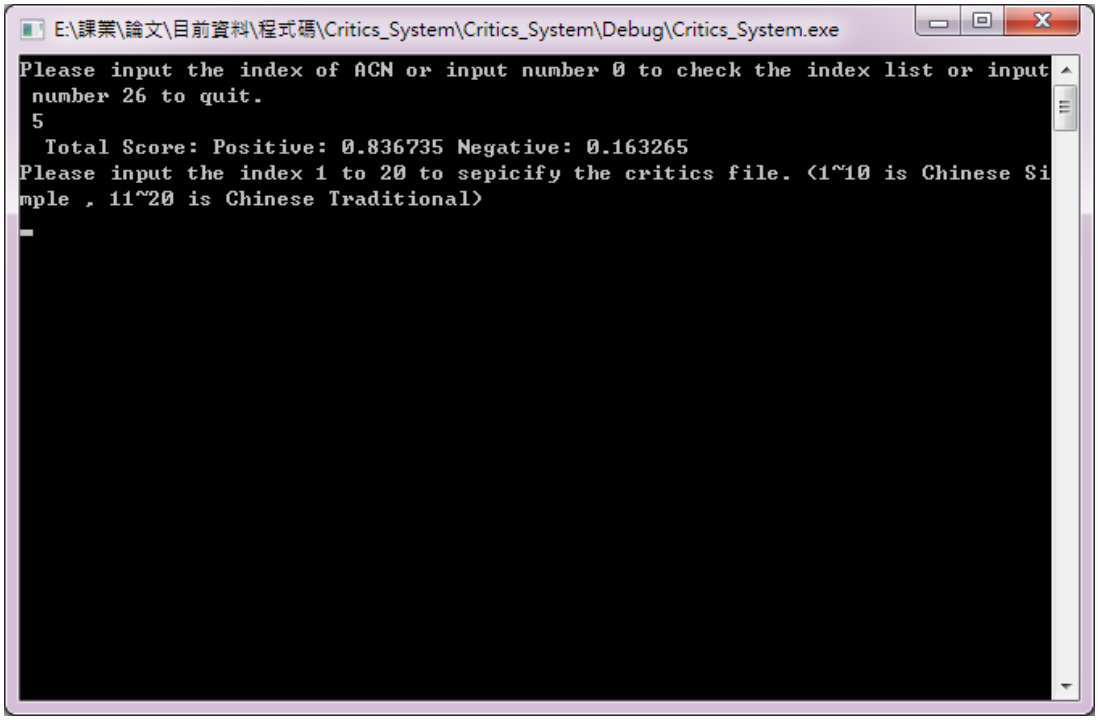

圖 5- 11 顯示作品總體評論分數

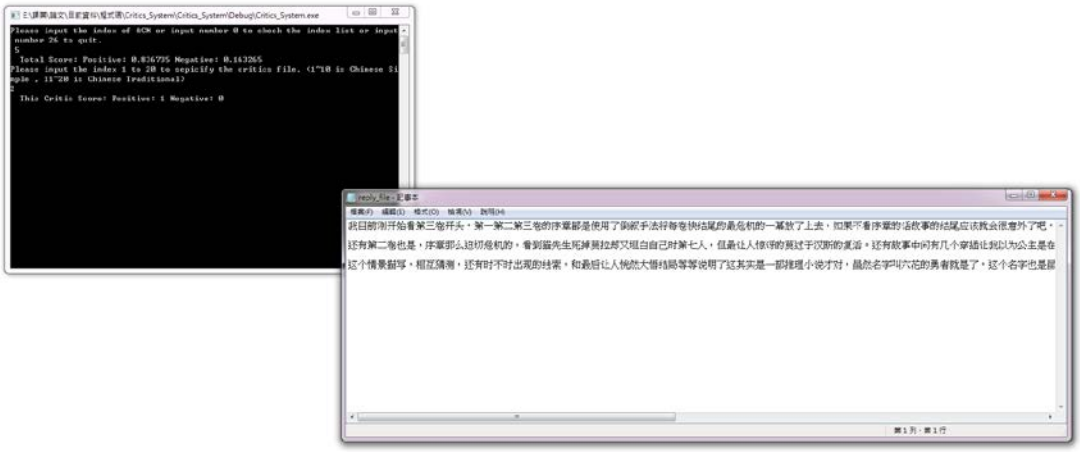

圖 5-12 顯示作品單一評論

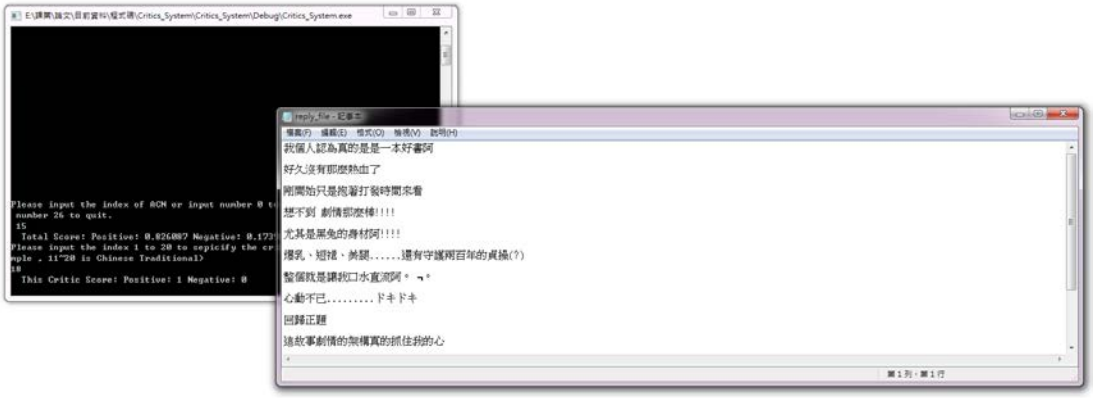

圖 5- 13 顯示作品單一評論

第六章 結論

本研究針對劇情特徵詞對意見判斷的干擾，提出了一套解決方法，並基於 NTUSD 與 E-HowNet，建立了對於作品評論使用的特徵詞詞典，實現利用系統判斷意見的簡易動漫輕小說作品評論系統。本論文主要使用規則與詞典為判斷依據。這個方法在資料庫本身的選用上受到不少限制，而斷詞或詞性的標註錯誤，也會影響意見的判斷。未來的研究可以透過特徵抽取方式，配合監督式學習，實現意見判斷的系統。語料庫部分，也需收集更多的評論，或直接使用作品內的文句，加強特徵詞的抽取效果。

本研究的訓練與測試語料主要是針對輕小說與動漫的評論，而類似的方法亦可用於電影或電視劇的評論。未來希望能將此系統發展到不限於動漫輕小說的範疇。

Reference:

[1] M. Hu and B. Liu, “Mining and summarizing customer reviews,” *Proc. 2004 ACM SIGKDD Int. Conf. Knowl. Discov. data Min. KDD 04*, vol. 04, p. 168, 2004.

[2] X. Ding, X. Ding, B. Liu, B. Liu, P. S. Yu, and P. S. Yu, “A holistic lexicon-based approach to opinion mining,” *Proc. Int. Conf. Web search web data Min. - WSDM ’08*, p. 231, 2008.

[3] S. S. Sohail, J. Siddiqui, and R. Ali, “Book recommendation system using opinion mining technique,” *2013 Int. Conf. Adv. Comput. Commun. Informatics*, pp. 1609–1614, 2013.

[4] 謝鎮宇(國立交通大學),“國 立 交 通 大 學 資訊學院 資訊學程 碩 士 論 文.”

[5] 邱鴻達(國立交通大學),“國 立 交 通 大 學 資訊科學與工程研究所 碩 士 論 文.”

[6] C. L. Liu, W. H. Hsaio, C. H. Lee, G. C. Lu, and E. Jou, “Movie Rating and Review Summarization in Mobile Environment,” *Ieee Trans. Syst. Man Cybern. Part C-Applications Rev.*, vol. 42, no. 3, pp. 397–407, 2012.

[7] C. Wang, “Opinion Mining Research on Chinese Micro-blog,” no. 2013, pp. 115–120, 2015.

[8] C. D. Manning and H. Schutze, “Foundations of Statistical Natural Language Processing,” 1999.

[9] K. M. G. A. Miller, R. Beckwith, C. D. Fellbaum, D. Gross, “WordNet: An online lexical database. Int. J. Lexicograph. 3, 4, pp. 235–244.,” pp. 1–9, 2011.

[10] 黃居仁陳克健,“中央研究院漢語料庫.”

[11] P. D. Turney, “Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews,” *Proc. 40th Annu. Meet. Assoc. Comput. Linguist.*, no. July, pp. 417–424, 2002.

[12] Q. Dong, “HowNet,” 1988.

[13] 梅家駒、竺一鳴、高蘊琦與殷鴻翔 1983,“同義詞詞林,” p. 1983, 1983.

[14] 朱嫣岚, 闵锦, 周雅倩, 黄萱菁, and 吴立德,“基于HowNet的词汇语义倾向计算,” 中文信息学报, vol. 1. pp. 14–20, 2006.

[15] M.-C. de Marneffe, C. D. Manning, and C. Potts, “‘Was it good? It was

provocative.’ Learning the meaning of scalar adjectives,” *ACL ’10 Proc. 48th Annu. Meet. Assoc. Comput. Linguist.*, pp. 167–176, 2010.

[16] K. Yessenov and S. Misailovi, “Sentiment Analysis of Movie Review Comments 6.863,” *Methodology*, pp. 1–17, 2009.

[17] H. P. Luhn, “A Statistical Approach to Mechanized Encoding and Searching of Literary Information,” *IBM J. Res. Dev.*, vol. 1, no. 4, pp. 309–317, 1957.

[18] K. S. Jones, “A statistical interpretation of term specificity and its application in retrieval,” *J. Doc.*, vol. 60, no. 5, pp. 493–502, 2004.

[19] 楊遠 ， 香港天健出版社1962., “標點符號研究,” p. 1962, 1962.

[20] 陳克健、黃淑齡、施悅音、陳怡君, “多層次概念定義與複雜關係表達－繁體字知網的新增架構,” no. 2, pp. 1–13, 2001.

[21] “結巴斷詞 https://github.com/fxsjy/jieba.” .

[22] T. X. Jingbo Zhu, Muhua Zhu, _ Qiang Wang, “NiuParser : A Chinese Syntactic and Semantic Parsing Toolkit,” *Proc. 53rd Annu. Meet. Assoc. Comput. Linguist. 7th Int. Jt. Conf. Nat. Lang. Process. (Volume 1 Long Pap.*, pp. 145–150, 2015.

[23] L.-W. K. andHsin-H. Chen, “Mining opinions from the Web: Beyond relevance retrieval,” *Int. Rev. Res. Open Distance Learn.*, vol. 14, no. 4, pp. 90–103, 2013.