賭博案件起訴書文句分類之研究

Toward Classification of Sentences in Indictment Documents for Offenses of Gambling

廖鼎銘 劉昭麟
國立政治大學 資訊科學研究所
{g1753026,chaolin}@nccu.edu.tw

摘 要

複雜的刑事案件常有多位涉案人，因此刑事案件的起訴書常會描述涉案人如何共同完成其犯罪行為。為了建立刑事案件的判決輔助系統，我們以賭博案件為例，讓程式從標記過的訓練語料中學習如何判別跟刑事判決相關的文字訊息的類別。本論文提出將起訴書中的文句作分類的方式及程式如何能夠透過訓練語料習得分類規則的演算法。以實際的案例檢驗，結果顯示我們的演算法對於特定句型的抽取可以達到不錯的效果，但是仍有相當的改進空間。

1. 概論

對法官來說，經由法學資訊系統的幫助，他們可以省下不少處理繁雜工作的時間，把精神花費在電腦無法取代的工作上，這對於降低工作壓力、增加工作效率甚至避免人為疏失都有正面的影響。

法學資訊系統的研究已經進行了許多年，而且有相當的成果[6,10]。例如 Branting 等人在 1998 年提出一個自動產生法律文件草稿的系統[3]，此系統可以根據人工產生的法律文件結構的文法，自動合成類似的文件，以節省人工撰寫法律文件草稿的時間。該文法包括兩個主要部分，第一個部分描述該法律文件目的(譬如：撤回訴訟)與一些必要詞彙的關係；第二個部分對該文件類型所能使用的用語作出限制。Rissland 和 Ashley 發展出一個應用 case-based reasoning 的系統[9]，它可以根據使用者輸入的案件，找出資料庫裡相關的案例，藉以分析出使用者輸入的案件裡面，被告與原告哪方面比較有利。

在法學資訊和 Web Mining 的領域裡，有許多藉由把文件結構化，以提供系統某些資訊的技術與概念，值得我們參考。例如 Boella 等人在 2001 年提出利用 ontological basis 去表達 legal relation，以達到 agent 能夠處理該文件的可能性，legal relation 指的是法律規定“在何種情況下有何種行為，則必須負起何種責任”的規則[2]。Allen 等人曾提出一個近似自然語言的知識表達法，Aide[1]，它的特色在於，其 knowledgebase 可以與 web 上的系統互動；它可以根據 objects 和 attributes 來作推論；它的 rule 的語法比較近似法律用語的語法。Cooley 等人利用 multilevel databases 和 web query systems，把半結構化文件變得更為結構化；利用資料的關係、特徵找出 sequential patterns[4]。Craven 等人，用人工建立 ontology 並給予一個由一些被標記過的網頁所構成的集合，稱為 seed knowledge，以從網路中學習 knowledge-base objects 和 relations，其中，他們用 text classification 找出網頁中感興趣的類別；用 relational learning 找出 ontology 中定義的 relation 的 instances，並從已有的結構中，利用超連結找新的 relations[5]。Kolodner 與 Leake 在 Case-Based Reasoning(CBR) 的介紹裡[7]，提到 CBR 利用 library of cases 和合併過的 cases 去對一個新問題做出解答，這些概念也值得我們引用。

過去，我們從司法院法學資訊全文檢索系統[11]及其他相關地方蒐集資料，建立了一套輔助法官判決的系統[8]，該系統可以從簡易法庭的起訴書中，根據其中事實欄位的資料來做出判決，但是只能判斷一個人或多個人犯了同一種罪刑的案件。我們稱此為**「一人一罪」**與**「多人一罪」**，**「多人多罪」**的意義可依此類推。我們想把系統擴展成可以處理刑事法庭中涉及多人多罪的起訴書。其中，首要工作是**分配行為**，也就是把起訴書所述的犯罪行為分配到被告的人名之下，以便將來判斷適用的法條。由於刑事法庭起訴書的事實欄位資料較簡易法庭要來得複雜許多，若想要直接分配起訴書所述的犯罪行為到被告的人名之下有相當的難度。因此，在本論文中，我們提出將資料半結構化的方法，以人工從事實欄位的內容中找出四種格式，訓練程式自動找出屬於這四種格式的語句出來，以幫助將來分配行為時的分析與處理。我們拿 178 篇起訴書做為訓練用案件，142 篇起訴書做為測試用案件，實驗結果顯示，我們提出的方法精確率大約在 0.85 左右。

本論文的章節架構如下：我們在第 2 節提出設計動機。第 3 節介紹實驗的前置步驟。第 4 節介紹並討論實驗使用的兩種斷詞原則，以及找特徵關鍵詞集合的方法。第 5 節描述建立 case database 的過程，以及如何利用 case database 切割出句組。第 6 節說明並討論實驗結果。第 7 節為結論與未來發展方向。

2. 設計動機

在一份多人多罪的起訴書中，分配起訴書所述的犯罪行為到被告的人名之下是困難的。我們用如下的實際例子來說明：「李○○以月薪新臺幣二萬多元，僱用陳○○，負責內外場服務，而共同與不特定之人賭博財物。」我們的工作是要讓程式知道「李○○僱用陳○○。李○○與不特定之人賭博財物」以及「陳○○負責內外場服務。陳○○與不特定之人賭博財物」。人類可以分辨「僱用」這個行為是「李○○」做的，「負責內外場服務」是「陳○○」做的，「而共同與不特定之人賭博財物」是兩個人都有做的；但程式不容易分辨，程式不知道哪些行為是單獨行為，哪些是共同行為，甚至不知道「負責內外場服務」這些字是不是一個行為，該不該拿去歸類給某一個人(假設程式已經能夠分辨人名)。

在上述例子中，如果程式能預先知道「僱用陳○○，負責內外場服務」這段話的結構是一個人被僱用到賭博場所去做一個行為，換句話說，如果我們

能提供譬如在賭博場所被僱用的人會有哪些行為，程式就可以根據這些資訊去做搜尋，找出符合的文字並歸類給「陳○○」，而不用再去辨認哪些字是行為哪些字不是。也就是說，如果能分析出事實欄位的結構，並給予每個結構欄位相關的標記，那我們將可以降低程式分析處理事實欄位內容時的難度。此即為本研究的構想與目的：我們針對事實欄位，觀察它的內容，挑出有較固定格式的語句，訓練出該種格式的規則。再藉由這些格式去找出事實欄位的結構，最後讓程式根據結構與每種格式的規則，去分類一篇事實欄位的內容。分類後，程式可以從這個語句所屬的格式類型，得到許多我們已經定義在該格式裡的資訊，有助於將來想分配行為時的分析與處理。簡單地說，本論文提出的方法，主要目的在於過濾事實欄位資料，使被過濾後的資料，分析處理變簡單。

我們把事實欄位的內容分類成四種，**格式 A、B、C 和 D**。格式 A 為賭博方式的描述，格式 B 為賭客被查獲過程的描述，格式 C 為被查獲物品的描述，格式 D 為偵查起訴單位的相關描述；但有某些語句不屬於這四種格式，在本論文中，就不會對它們加以處理。這四種格式，是由我們閱讀訓練用案件裡事實欄位的內容之後決定的，相較於事實欄位其他部分的內容顯得較為規律，所謂的**規律**是指該種格式幾乎在每篇起訴書事實欄位中都會出現，並且同種格式的每個句組之間有比較類似的起始句和結尾句，句組裡面若有出現人名和行為，也全屬於多人一罪的範圍，結構簡單，有利於將來分配行為時的分析與處理。因此，我們將事實欄位分成如圖 1 的結構。

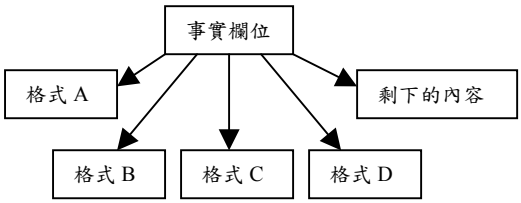

圖 1　事實欄位內容的結構

以下是一篇起訴書事實欄位以及四種格式的範例，擺設及扣得的賭博器具以“…”代替。

一、 緣王○○（已審結）明知未依電子遊戲場業管理條例規定辦理營利事業登記者，不得經營電子遊戲場業，竟基於經營電子遊戲場及常業賭博之犯意，自民國○○年○○月○○日起，在○○縣○○市○○路○○巷○○號旁停車場內鐵皮屋之公眾得出入之場所內，未經登記擅自擺設電動賭博機具「拉霸」十九台…供不特定人把玩，並先後於○○年○○月下旬至○○月間，以每月薪資新台幣（下同）二萬七千元至三萬元不等之薪資，僱用與其有共同以賭博為常業之犯意聯絡之洪○○及周○○、蘇○○為員工，由洪○○負責櫃檯、兌換現金等工作，周○○擔任開分員，蘇○○負責維修機台，共同與不特定之人賭博財物。其賭博方法為賭客先向店員兌換代幣開啟電子賭博機具之分數，由賭客自行下注，如未押中，分數沒入歸王○○所有，如有押中，則可得倍數不等之分數，最後再以機具上顯示之分數以相同比例換取同額之現金，洪○○與王○○、周○○、蘇○○等人均賴此維生而以之為常業。嗣於○○年○○月○○日晚間○○時○○分許，適有賭客詹○○、許○○、王○○等人在上址把玩前述電動賭博機具時，為警當場查獲，並扣得前揭電動賭博機具七十四台…中獎錄影帶四卷。

二、 案經○○縣警察局○○分局報請臺灣○○地方法院檢察署檢察官偵查起訴。

其中，「其賭博方法為賭客先…換取同額之現金，」為格式 A，「嗣於○○年○○月…為警當場查獲」為格式 B，「並扣得前揭…中獎錄影帶四卷。」為格式 C，「案經○○縣…偵查起訴。」為格式 D。

我們利用程式，分別替每種格式建立一個 case database。case 記錄尋找該種格式起始句的資訊和結尾句的資訊，**起始句**指的是可以做為開頭的句子，**結尾句**指的是可以做為結尾的句子。程式根據 case 抓到起始句和結尾句後，把這兩句話之間的句子全部抓出來，註明這一個**句組**(為了方便，我們稱這幾些連續的句子為一個句組)屬於該種格式。一個 case 即代表屬於該種格式的句組的資訊，我們稱它為一種句組的輪廓，我們利用這些輪廓去抓出一篇測試用案件裡，所有屬於某種格式的句組。譬如上面提到的格式 A 的範例中，「其賭博方法為賭客先向店員兌換代幣開啟電子賭博機具之分數」即為一個起始句，「最後再以機具上顯示之分數以相同比例換取同額之現金」即為一個結尾句，這兩句話與它們之間所有的內容，即為一組句組。

3. 前置步驟

我們從司法院以及其他相關地方的網頁上收集刑事法庭裡屬於賭博罪的起訴書，對這些資料進行處理。前處理部分包括篩選下載的起訴書，捨棄掉資料有缺漏或者錯誤的不用，最後得到 178 篇訓練用案件，142 篇測試用案件。接著用程式把人名、地址、時間標上標籤。因為人名辨識不是本實驗的研究目標，所以我們用人工把所有案件中出現的人名都先挑出來收錄進資料庫裡，以便程式辨識。至於把地址和時間標上標籤的目的，是希望這些標籤也可以是切割句組時的線索。

4. 特徵關鍵詞集合之取得

4.1 斷詞方法

本論文採取使用中文詞典的斷詞方法，並分別進行了長詞優先與短詞優先兩種方式，最後拿兩種實驗結果來互相比較。使用的詞典會影響斷詞後的結果。我們除了使用 HowNet 之外，因為考慮到 HowNet 缺乏法律這個領域的專用詞彙，所以我們利用統計訓練用案件中、字串的出現頻率的方法，自己造出一個專業詞典，用來輔助 HowNet。

使用中文詞典的斷詞方法通常採用長詞優先的經驗法則。一般來說，長詞優先有較高的機率把句子裡正確的詞斷出來，譬如說：「電子遊戲機具」這個詞，用長詞優先被斷成「電子遊戲機具」，用短詞優先則斷成「電子」、「遊戲」、「機具」。從上述例子我們可以看到，長詞優先斷出來的詞彙較正確，所謂的正確是指它的語意比較完整，比較能反應出我們原本想表達的意思。

我們斷詞的目的是想拿這些詞彙作為關鍵詞，利用關鍵詞去找出四種格式各自的起始句與

結尾句的特徵，程式決定起始句和結尾句的特徵關鍵詞後，就可以從一篇起訴書的事實欄位裡切割出該格式的句組出來。我們利用程式輔助人工建立特徵關鍵詞庫，以節省人力。

特徵關鍵詞是用來抓出一種格式的起始句或結尾句，而這個特徵關鍵詞在語意上完不完整甚至具不具備語意，會否影響其功用的正確性？一個語意完整的詞彙較容易被人從一個句子中發現，譬如我們讀到「電子遊戲場」時馬上可以知道裡面有「電子」這個有意義的詞，但我們可能得想一下才會發現裡面有「戲場」這個無意義的詞。所以語意有利於以人工檢索來建立特徵關鍵詞詞庫，但當我們是用程式而非人工去做這個工作時，語意的完整性或這是不是一個有意義的詞就變的不是那麼重要了，因為程式明不明白那個詞彙的意義，並不影響它能否發現該詞彙的存在的可能性。因此，一般採用長詞優先的考量，語意完整性，在本論文的方法中可能不是必要的，具備完整語意的詞彙，它的功能仍然存在，但不具備完整語意的詞彙可能也有用處。使用短詞優先時，相較於長詞優先，我們會得到較多語意不完整的詞彙與較少語意完整的詞彙。

針對這個問題，藉由觀察下面提出來的實際情況來討論。在實驗中，格式 C 的起始句用短詞優先的原則可以得到一個特徵關鍵詞：「機賭」，這個詞彙本身不具有意義，但確實可以根據這個詞彙去抓出該格式的其中一種起始句，其他還有許多類似的例子，所以無意義的詞彙在本論文的方法中，有可能可以發揮作用。看另一種情況，實驗中格式 B 的結尾句用長詞優先的原則可以得到如下：「經警在上址當場查獲」、「為警當場查獲」、「經警當場查獲」三個相似的特徵關鍵詞，用短詞優先的原則可以得到「查獲」這個特徵關鍵詞，而前面三個長詞可以抓到的結尾句用後面的短詞也都可以抓到，我們觀察實驗裡類似的例子，可以發現，短詞優先得到的特徵關鍵詞相較於長詞優先，通常包含較多的起始句，可能可以找到較多的句組。這些句組有可能不是正確答案，但在本論文中，我們會用其他方法來提高答案的正確性。因此，我們認為短詞優先是值得嘗試的。

我們分別用長詞優先和短詞優先的原則去對相同的資料斷詞後得到兩組資料，然後拿這兩組資料去進行相同的實驗。在第 6 節中我們可以藉由實驗數據去觀察出這兩種斷詞原則在本論文方法裡的適用性。

4.2 取得特徵關鍵詞集合

本研究的目的在過濾一份起訴書的事實欄位，希望把屬於四種格式的句組先切割出來，將來就可以分別針對這些句組的格式類別，用簡單的方法進行分析和分配行為，對剩下的內容則用其他較複雜的方法分析和分配行為。

我們對訓練用案件做兩種斷詞，得到兩份輸出檔案，然後各自從中取出四種格式的句組之後，每份檔案的資料即分成兩個部分。第一個部分是每篇起訴書的事實欄位，第二部分是用人工從事實欄位切割出來的句組，這些句組分屬於四種格式。由於我們會對不同檔案的四種不同格式進行相同的實驗步驟，所以我們只藉由格式 A 來說明，且不特別標明目前的資料屬於哪種斷詞方式。

我們的工作相當於將起訴書事實欄位結構化，然後從裡面找出一些特定的樣式(pattern)，如果我們能知道格式 A 曾經在起始句和結尾句中用過哪些詞，並把這些資訊存入 case database 中，就可以去對手上的測試用案件做全文搜尋，找出那些詞的位置，以此推論格式 A 的起始句和結尾句，然後抓出屬於格式 A 的句組。所以接下來，我們要建立格式 A 的 case database。首先，我們必須取得一些需要用到的**特徵關鍵詞集合**。我們用圖 2 說明事實欄位中，用來取出各種特徵關鍵詞集合的各個部分。下文中，“一句話”是以逗號、分號與句號來彼此區隔。

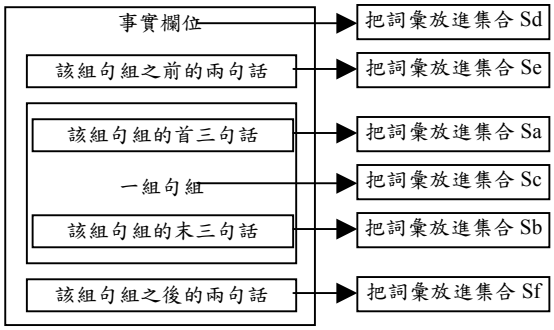

圖 2　訓練語料(起訴書的事實欄位)的處理

我們先把所有訓練語料裡，屬於格式 A 的句組取出來，程式會把每組句組裡已經斷過詞的首三句話的詞彙挑出來，把這些詞彙放進集合 **Sa** 裡，把末三句話的詞彙也挑出來，放進集合 **Sb** 裡，整個句組的所有詞彙放進集合 **Sc** 裡。接著處理句組之外的句子。程式把所有訓練語料非出現在句組裡句子的詞彙放進集合 **Sd**，把所有出現在句組之前的兩句話裡面的詞彙放進集合 **Se**，出現在句組之後的兩句話裡面的詞彙放進集合 **Sf**。最後拿集合 Sa 和集合 Sd 去做差集得到集合 **Sg**(Sg=Sa-Sd)，Sg 即為格式 A 起始句的特徵關鍵詞集合。同理，格式 A 結尾句的特徵關鍵詞集合為 **Sh**=Sb-Sd。

為了找出起始句和結尾句，我們建立了集合 Sa 和 Sb。但是出現在格式 A 起始句和結尾句的詞也有可能出現在別的位置，這會導致我們抓錯起始句和結尾句。抓錯的狀況有兩種，一種是抓到格式 A 句組裡面的句子，也就是這個句子仍然屬於格式 A，但不是起始句或結尾句；另一種是抓到格式 A 以外的句子，也就是這個句子不屬於格式 A 的句組。前一種錯誤只會造成沒有把屬於該格式的所有句子抓乾淨，但是不會造成抓到不屬於該格式的句子；後一種錯誤正好相反。

如果一組句組裡混雜了不屬於該格式的句子，將會影響將來分析句組的分析結果。譬如說在格式 B 裡，只要抓到人名，就可以確定這個人有在公共場所裡賭博的行為，如果混到了非 B 格式的句子，譬如第 2 節的起訴書事實欄位範例裡，「周○○擔任開分員」這一句，那歸類「周○○」的行為時就會發生錯誤。如果是沒抓乾淨的狀況，譬

如上述範例中，格式 B 裡的「適有賭客詹○○、許○○、王○○等人在上址把玩前述電動賭博機具時」這句話沒抓到，那這句話就會被留到剩下的內容用較複雜但非特定的方法分析，並不會因此就導致分析錯誤；至於被抓到的句子依然適用於專屬格式 B 的分析方法。因此我們比較不在乎上述沒抓乾淨的錯誤，而必須避免抓錯句子的錯誤。

針對這個需求，我們去做 Sg=Sa-Sd 的動作，因為 Sd 蒐集了所有非格式 A 句子裡的詞彙，所以集合 Sg 裡面的詞彙不曾出現在訓練用案件裡格式 A 之外的句子中，因此，根據 Sg 取得的句子就比較不會發生上述想避免的錯誤。同理，取得集合 Sh 的目的也在於避免此一錯誤。如果不用差集去取得集合 Sg，而是用比較常見的利用門檻值的做法，也就是當集合 Sa 裡面某個詞出現在集合 Sd 裡面時不要直接刪掉，只當這個詞在集合 Sd 裡面的紀錄的出現頻率高於門檻值時才刪掉，這樣雖然可以增加集合 Sg 裡面的詞彙，被抓到的句組有可能因此而增加，但這些同時屬於 Sa 和 Sd 的詞彙卻會造成抓錯句子的錯誤，因為我們希望能藉由差集來提高本方法的正確性，所以決定不採取利用門檻值的做法。

集合 Sa 收集了每組格式 A 的句組首三句話的詞彙，這是因為在實作中，我們曾經只收集第一句話和前兩句話裡面的詞彙，但是經過差集後 Sg 裡面的詞彙剩下太少，導致我們用訓練用案件先進行的一場實驗，其實驗結果不盡理想，所以最後設定為三句話。集合 Sb 取末三句話也是同樣道理。

集合 Sa 取了首三句話的詞彙，則根據集合 Sg 裡面的某個詞彙去抓起始句時，若該詞彙實際上是出現在訓練用案件裡面的第二句或第三句而不是第一句，就有可能導致起始句抓成測試用案件裡面的第二句或第三句，而遺漏了第一句或前兩句話。為了解決這個問題，我們從第二句抓到起始句時，會記錄第一個句子的資訊，從第三句抓到起始句時，則記錄第一和第二個句子的資訊，當從測試用案件切割句組時，若我們抓到起始句，則會檢查起始句的前兩個句子是否包含了上述被記錄的資訊，若有，我們會往前挪動起始句的位置。如果某組句組之前、不屬於這組句組的兩句話裡，恰好有包含上述被記錄的資訊，為了避免可能抓錯句子的錯誤，所以我們建立集合 **Si**=Sa-Se。建立 **Sj**=Sb-Sf 也是基於同樣的道理。

在為每種格式建立 case database 之前，我們都會利用該格式所有訓練用案件，計算出該格式的 Sg、Sh、Si 和 Sj 四個集合。我們再利用個別格式的個別訓練語料建立個別格式的 case。詳細步驟描述於下面小節。

5. Case Database

5.1 建立 case database

Case databases 中的每一個 case 都有三個欄位，第一個欄位稱為 **Hcase**，記錄該 case 的特徵關鍵詞，這個特徵關鍵詞屬於集合 Sg，記錄句組起始句可能包含的詞。因為一個起始句的特徵關鍵詞只會對應到一個 case，所以我們用 Hcase 作為 case 的索引。當檢查到某個起始句特徵關鍵詞，發現它沒有對應的 case 時，就為它建立一個相對應的 case；若已經有了，就根據索引抓出該 case，以增加該 case 第二和第三個欄位的相關資訊。第二個欄位稱為 **pre-Keywords**，記錄許多屬於集合 Si 的詞彙，我們利用這個欄位的資訊，以第 4.2 節所提到的方法，去決定是否往前挪動測試案件中一組句組的候選起始句位置。第三個欄位記錄 Hcase 對應的 **Tcases**。與 case 的第一和第二個欄位類似，每個 Tcase 都有兩個欄位，第一個欄位記錄句組結尾句可能包含的詞，屬於集合 Sh，可作為 Tcase 的索引；第二個欄位為 **post-Keywords**，裡面記錄的所有詞彙皆屬於集合 Sj，用來決定是否往後挪動測試用案件中，一組句組的候選結尾句位置。我們利用第 4.2 節所描述的各個集合一次處理一組句組，建立 Tcase 的演算法如下。

Input: 一篇格式 A 經過斷詞處理的句組={SE_1,SE_2…SE_m}。
Output：All-T-set。

令 SE_x 為句組中的第 x 句話(m-2≤x≤m)。令 W 為 SE_x 中的一個詞彙。Tcase(W)表示以 W 為索引的 Tcase。All-T-set 為用來存放 Tcases 的集合。

Step1：建立一個空集合，令為 All-T-set。

Step2：我們依序處理這篇句組的最後三句話。讀入 SE_x，檢查所有屬於集合 Sh 的 W，如果已經幫它建立過 Tcase(W)，則找出該 Tcase；如果沒有，則幫它建立。把得到的 Tcases 放入 All-T-set 中。

Step3：檢查 SE_{x+1} 和 SE_{x+2} 中的詞彙 W，x+1≤m，x+2≤m。把所有也屬於集合 Sj 的 W 存入上個步驟中得到的 Tcases 的第二個欄位。

接著以類似的方法處理這組句組的前三句話，以為每一個屬於首三句中不同的詞建立一個 Hcase 和相對的 pre-Keywords 以完成一個 case。這一個過程跟上述演算法相似，主要的差別在於：第一，case 的 Hcase 是從首三句所得 W∈SE_i，i←1 to 3，並且必須是 W∈Sg；第二，pre-Keywords 欄位裡記錄的所有 W 屬於集合 Si。然後我們會把 All-T-set 裡的 Tcases 寫入由這組句組得到的所有 cases 的第三個欄位裡，亦即建立 Hcase 與 Tcase 的連結。

我們從一組句組裡，根據一個屬於集合 Sg 的詞彙去建立 Hcase 及尋找其他欄位的資訊，以完成一個 case。一個起始句特徵關鍵詞可能出現在不同組句組的起始句中，但是我們只會為一個起始句關鍵詞建立一個唯一的 case，相關配套的欄位如 pre-Keywords 和 Tcases 則會合併在同一個 case 中。因此，每一個 case 包含了一個曾經出現在句組首三句的特徵關鍵詞，即 Hcase，以及它的 pre-Keywords 的資訊，並且連結到數個曾經與該 Hcase 一起出現的 Tcases。一個 Tcase 則包含一個曾經出現在句組末三句的特徵關鍵詞，以及它的 post-Keywords 的資訊。在切割句組時，我們會根據特徵關鍵詞去抓出該詞彙對應的 case 來使用。

5.2 切割句組

我們要利用上一節所描述的方法建立 case databases 去分析測試語料中的句子，用 cases 找出起始句與結尾句，把屬於所尋找的四種格式的句

組切割出來。切割方法如下。

Input: 一篇測試用案件={SE_1,SE_2…SE_m}。
Output: 數個號碼組合 Num(H-tag,T-tag)，這些組合被儲存在集合 N-set 之內。
令 SE_x 為測試用案件中的第 x 句話(1≤x≤m)。令 W 為 SE_x 中的一個詞彙。Hcase(W)表示以 W 為索引的 Hcase。Tcase(W)表示以 W 為索引的 Tcase。H-tag 表示一篇句組起始句的句子號碼，亦即一篇句組的起始句在本篇測試用案件中是第 H-tag 句話；T-tag 為結尾句號碼。
Step1：讀入 SE_x，檢查裡面的 W，如果沒有對應的 Hcase(W)，則令 x=x+1 並且重複此步驟；如果有，則取出該 case，並記錄該 case 的 H-tag=x。在 SE_x 可能可以找到數個 cases，我們會把它們全部記錄下來。
Step2：我們分開並依序處理被記錄下來的 cases。根據每個 case 的 pre-Keywords 欄位，往前檢查 SE_{x-2}，x-2≥1。如果有詞彙出現在正在處理的 case 的 pre-Keywords 內，則調整該 case 的 H-tag=x-2；如果沒有，則以相同方法檢查及處理 SE_{x-1}，x-1≥1。
Step3：我們仍然分開並依序處理被記錄下來的 cases。根據每個 case 的 Tcases，往後一句句檢查 SE_y，x<y≤m，遇到句號時也會停止檢查。檢查的目的是為了找出 T-tag，方法跟步驟 1 和步驟 2 找 H-tag 的方法類似，主要的差別在於 T-tag 的值是根據 post-Keywords 往後檢查句子來做出調整。一個 case 可能找到好幾個 T-tags，我們只取值最大的一個，與該 case 的 H-tag 配對。

我們重複步驟 1 到 3，直到處理完所有句子。每個 case 都會找出一組起始句與結尾句的號碼組合，也就是最後會得到數組號碼，因為這些號碼組合可能會有交錯的現象，所以我們用如下的演算法合併它們。

```
for ∀ Num(H-tag,T-tag)，比較任意兩組，令為 Num(a,b)、
        Num(c,d)，a≤c
  if c≤b
    if d≤b 把它們合併成 Num(a,b)
    else 把它們合併成 Num(a,d)
  else do nothing   //兩段落沒有交集
```

合併時，我們的起始句會取較小的號碼而結尾句會取較大的號碼，以抓較多的句子出來。若起始句取較大的號碼而結尾句取較小的號碼，則可能比較不容易抓到該組句組之外的句子，但是我們在第 4.2 節提到，本論文已經用 Sg=Sa-Sd、Sh=Sb-Sd、Si=Sa-Se 以及 Sj=Sb-Sf 的方法避免抓錯句子的錯誤，所以，合併時我們的起始句會取較小的號碼而結尾句會取較大的號碼。最後，我們根據這些 Num(H-tag,T-tag)切割出該篇事實欄位中，所有屬於格式 A 的句組。

6. 實驗結果

目前我們的研究以刑事法庭中賭博罪的起訴書為主。實驗流程如下：首先我們先從司法院以及其他相關地方的網頁上收集起訴書，經過前處理(preprocessing)後，用人工從每篇訓練用案件中切割出符合四種格式的內容，再拿切割出來的結果，用兩種斷詞原則斷詞以得到兩份訓練用案件，然後分別去找出這兩份資料裡，四種格式種類的一些特徵關鍵詞集合，我們用這些特徵關鍵詞集合去建立各種格式的 case database 裡面句組的輪廓，最後讓程式根據各個 case database，一篇篇從測試用案件中切割出屬於這四種格式的句組出來，如圖 3 所示。

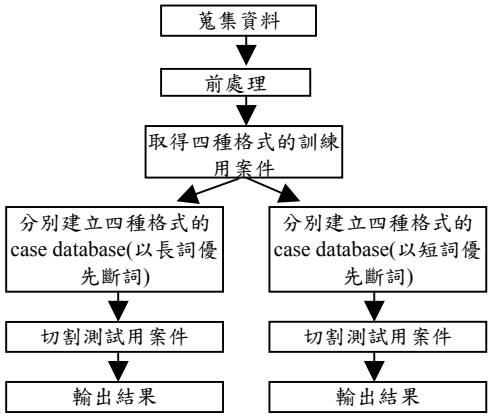

圖 3　實驗流程

我們的實驗資料經過前處理的篩選之後，取得 178 篇起訴書的事實欄位，用人工從中抽取出 A 格式句組 184 組、B 格式句組 205 組、C 格式句組 196 組、D 格式句組 185 組作為訓練用案件。在測試方面，篩選後我們取得 142 篇起訴書的事實欄位作為測試用案件。用人工從中抽取得到格式 A 的句組 134 組、格式 B 的句組 155 組、格式 C 的句組 153 組、格式 D 的句組 140 組，我們把人工得到的答案當作正確答案，以拿來和程式從測試用案件中切割出來的句組做比較，計算精確率(precision)和召回率(recall)。

表 1 和表 2 中，程式抓出一個句組即為一組答案，**「完全正確」**表示該組句組與正確答案一模一樣；**「沒抓乾淨」**表示正確答案裡的某些句子，程式在這組句組裡沒有抓到；**「錯誤」**表示程式在這組句組裡有抓到不在正確答案裡的句子。在一篇測試用案件中，譬如格式 A 的句組的正確答案只有 1 組，但程式可能抓成 3 組沒抓乾淨的句組，每組都只有正確答案的一部份句子，此時我們會把它算成抓到 3 組「沒抓乾淨」的答案。

公式(1)和(2)定義本實驗的精確率和召回率。**ans** 表示正確答案的組數；**Q-ans** 表示程式抓出來的答案的總組數；**M-ans** 表示 Q-ans 中「完全正確」的答案的組數；**C-ans** 表示 Q-ans 中「完全正確」加上「沒抓乾淨」的答案的組數。在第 4.2 節中我們提到，一組「沒抓乾淨」的句組在將來分配行為的工作中仍然可以作用，因此我們在計算精確率時，會把「沒抓乾淨」的句組算進 C-ans 裡。精確率與召回率的式子如下。實驗數據列在下面數個表格。

$$Precision = C\text{-}ans \;/\; Q\text{-}ans \qquad (1)$$

$$Recall = M\text{-}ans \;/\; ans \qquad (2)$$

	完全正確	沒抓乾淨	錯誤
格式 A 的句組	41	285	67
格式 B 的句組	91	62	68
格式 C 的句組	142	6	100
格式 D 的句組	131	11	7

表 1　以長詞優先斷詞的實驗數據

	完全正確	沒抓乾淨	錯誤
格式 A 的句組	25	242	45
格式 B 的句組	79	75	43
格式 C 的句組	128	15	44
格式 D 的句組	127	6	7

表 2　以短詞優先斷詞的實驗數據

	精確率	召回率
格式 A 的句組	326/393=83%	41/134=31%
格式 B 的句組	153/221=69%	91/155=59%
格式 C 的句組	148/248=60%	142/153=93%
格式 D 的句組	142/149=95%	131/140=94%

表 3　以長詞優先斷詞的實驗結果

	精確率	召回率
格式 A 的句組	267/312=86%	25/134=19%
格式 B 的句組	154/197=78%	79/155=51%
格式 C 的句組	143/187=76%	128/153=84%
格式 D 的句組	133/140=95%	127/140=91%

表 4　以短詞優先斷詞的實驗結果

我們觀察四種格式的句組時發現，格式 A 最不規律，其次是格式 B，格式 C 和格式 D 則較為規律。觀察實驗結果，可以發現當格式種類越不規律時召回率越低，但規律性與精確率並無直接的關係。以下探討之。

針對召回率的部分，測試案件裡，某種格式的句組會出現跟過去案件中完全不一樣的起始句或結尾句，因為不規律的格式的句組之間，原本就比較不容易有相似的起始句和結尾句，所以上述的情形容易發生。此時利用這個格式的 case database 無法完整的抓出這些句組，所以，召回率容易受到格式的規律性影響。

針對精確率的部分，我們在第 4.2 節提到，因為被利用來建立 cases 的特徵關鍵詞曾做過差集，所以，會造成精確率下降的情況，只有當測試案件句組之外的某些句子裡，恰好有符合該格式 cases 的資訊而被誤認為一組句組時，才會抓錯句子而導致精確率下降。發生這種情況的機率與一種格式的規律性無關。會發生這種情況是因為每位法官撰寫起訴書的風格不盡相同，譴詞用句有時會有所差異，如果有某位法官所寫的起訴書，恰好只出現在測試案件中而未曾出現在訓練語料裡，那就容易發生上述原本可以藉由差集去避免掉的錯誤。因為我們不知道起訴書是由哪位法官寫的，所以訓練案件無法涵蓋每位法官的風格。

7. 結論與未來展望

本論文提出刑事法庭賭博案件起訴書的事實資料分類方法，以期簡化將來建立判決輔助功能的難度。因此我們用人工從訓練語料中挑出四種格式的句組作為訓練用案件，然後利用這一些語料為四種格式建立各自的 case database，最後利用這些 case database 從測試用案件中切割出各個格式的句組來，進行過濾的動作。本論文的方法比較注重精確率，以短詞優先斷詞時，精確率約有 0.85；以長詞優先斷詞時，精確率約有 0.77。

未來我們將試著找出其他可能的格式，把事實欄位更為結構化。此外，會同時進行分配起訴書所述的犯罪行為到被告的人名之下的研究，之後會擴展實驗範圍到其他刑事案件的罪型種類。

致謝

本研究承蒙國科會研究計劃 NSC-92-2213-E-004-004 之部份補助。感謝台北地方法院板橋分院何法官君豪對本研究的諮詢和各項協助。

參考文獻

以下我們用 AI & Law 代替 Artificial Intelligence and Law。

[1] R. Allen, P. Chung, A. Mowbray and G. Greenleaf, AustLII's Aide - Natural Language Legislative RuleBases, *Proc. of the 18th Int'l Conf. on AI & Law*, 223-224. 2001.

[2] G. Boella, L. Favali and L. Lesmo, An Action-Based Ontology of Legal Relations, *Proc. of the 18th Int'l Conf. on AI & Law*, 227-228. 2001.

[3] L. K. Branting, J. C. Lester and C. B. Callaway, Automating Judicial Document Drafting：A Discourse-Based Approach, *AI & Law*, 6(2-4), 1998.

[4] R. Cooley, B. Mobasher and J. Srivastava, Web Mining：Information and Pattern Discovery on the World Wide Web, *Proc. of the 9th Int'l Conf. on Tools with Artificial Intelligence*, 558-568, 1997.

[5] M. Craven, D. Freitag, A. McCallum, T. Mitchell, K. Nigam and C. Y. Quek, Learning to Extract Symbolic Knowledge from the World Wide Web, *Proc. of the 15th American Association for Artificial Intelligence*, 509-516, 1998.

[6] *Int'l Conf. on AI & Law* 1987-2003.

[7] J. Kolodner and D. Leake, A Tutorial Introduction to Case-Based Reasoning, D. Leake, Editor, *Case-Based Reasoning：Experiences, Lessons, & Future Directions*, 31-65, MIT Press, 1966.

[8] C.-L. Liu, C.-T. Chang and J.-H. Ho, Case Instance Generation and Refinement for Case-Based Criminal Summary Judgments in Chinese, *Journal of Information Science and Engineering, to appear,* 2004.

[9] E. L. Rissland and K. D. Ashley, A Case-Based System for Trade Secrets Law, *Proc. of the 1st Int'l Conf. on AI & Law*, 60-66, 1987.

[10] 陳起行，"德國法資訊學，"法學叢刊，46(2):79-85, 2001.

[11] 司法院法學資料全文檢索 http://wjirs.judicial.gov.tw/jirs/