

中原大學資訊工程學系

108 學年度專題實驗報告

法規資料搜尋引擎

專題成員：

10520136 黃少麒

10520104 莊東翰

10520121 杜欣洋

指導教授：吳宜鴻 教授

目錄

第一章、專題摘要

1.1 前言	1
--------	---

第二章、研究動機與目的

2.1 研究動機	1
2.2 研究目的	1
2.2.1 口語化搜尋	1
2.2.2 文字分析	1

第三章、專題實作方法

3.1 專題架構	2
3.2 資料架構	3
3.3 網路爬蟲	6
3.4 網頁設計	7
3.5 伺服器架設	9
3.6 資料庫系統	11
3.7 資料庫管理	12
3.8 文字分析	13
3.8.1 去除停詞	13
3.8.2 Jieba 斷詞	13
3.9 句子數據化方法	14
3.10 口語化模型訓練：Word2Vec 訓練模型	15
3.11 二分類訓練模型：SVM 訓練模型	16
3.12 結果分數排序	18

第四章、未來方向

4.1 結論	19
4.2 未來方向	22
4.2.1 系統改善	22
4.2.2 系統擴展	22

第五章、參考資料

5.1 使用資料	23
5.2 官方工具	23
5.3 使用套件	23

第六章、附錄

6.1 系統展示影片的初步問卷調查	24
6.2 問卷評論結語	25
6.3 分工與成員心得	26
6.3.1 分工表	26
6.3.2 收穫與貢獻	26

第一章、專題摘要

1.1 前言

本專題實驗著重於將法規資料整理成法條形式讓使用者可以不用在搜尋過後還需要去整個法規資料中尋找相對應的法條，並將新增口語化搜尋的模式讓使用者就算不知道法規中的專有名詞還是可以找到相對應的法條進而達到便民的效果。

主要的問題為口語化的搜尋，要如何解決將使用者輸入的字句與法規中的專業字句做連接，讓使用者可以不會因為不知道專業化的用詞而無法找到自己想要的結果。

第二章、研究動機與目的

2.1 研究動機

在現有的法規資料中有一套“全國法規資料庫”，但經過我們多次的測試後發現其有些許的缺點，這讓我們想要將其改進並且未來有機會可將兩套系統合併之後讓一般民眾也可以輕易的找到自己想要的法規資料，讓所有百姓可以增加法律的知識並且不再遇到事情時再吃悶虧。

2.2 研究目的

2.2.1 口語化搜尋

在現有的全國法規資料庫中並沒有支援口語化搜尋的方式，若輸入較為口語化的用詞則會顯示「搜尋不到結果」，此法規資料庫是使用全文檢索的方式去搜尋的，所以需要一些較為專業的字詞才可以找到結果，這也大大降低了一般民眾使用的意願。

什麼是口語化搜尋？

顧名思義就是一般民眾在日常生活中比較會提到的字詞，不像是法律中提到的專有字詞在一般日常生活中比較不會提到。

口語化字詞->EX：違停、肇逃、連續殺人犯判什麼刑……等

專業字詞->EX:業務過失、違規停車、肇事逃逸……等

2.2.2 文字分析

將使用者輸入的關鍵字句透過 Jieba 斷詞與停詞表抽取出關鍵字句中重要的字詞然後將其透過事先透過 SVM 與 Word2Vec 訓練好的資料進行交叉比對，取得關鍵字句的分類就可以獲得所要的資料。

第三章、 專題實作方法

3.1 專題架構

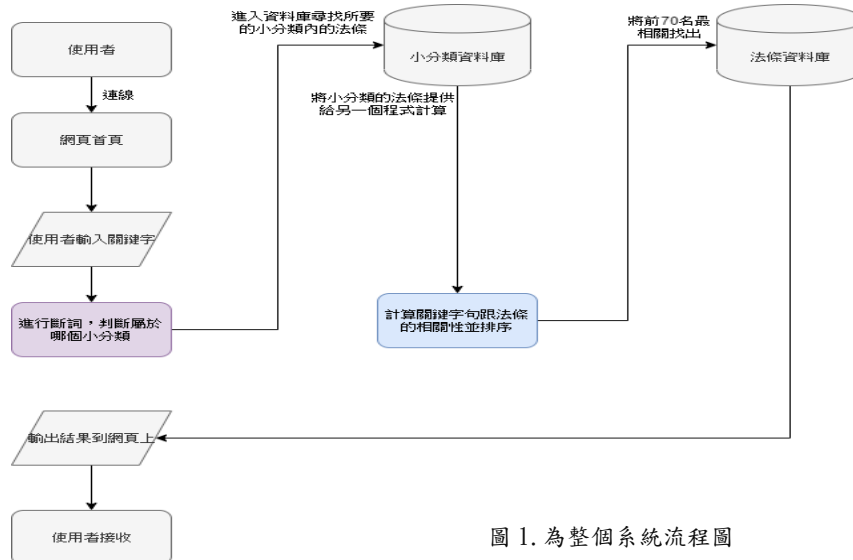


圖 1. 為整個系統流程圖

整個系統的流程圖如圖 1，圖 1 的左半部為使用者從連線到我們網頁再到我們分析使用者之關鍵字句，右半部兩個資料庫分別為存放所有法條內容之資料庫和我們事先進行分類的法條內容。

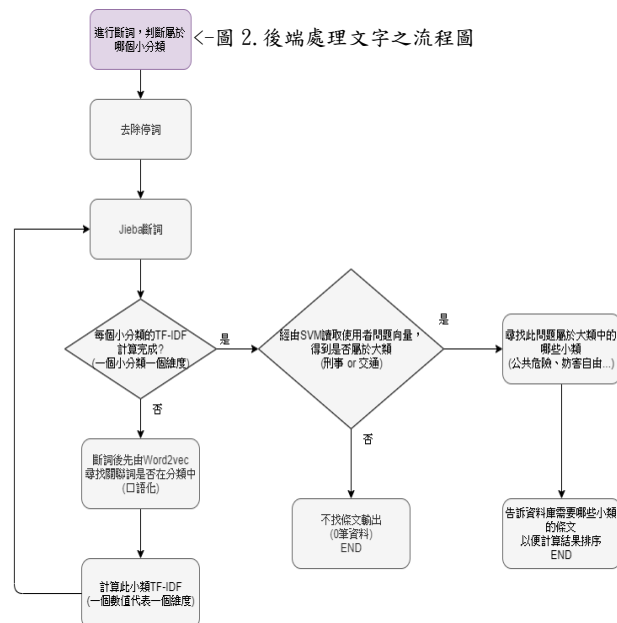


圖 2. 後端處理文字之流程圖

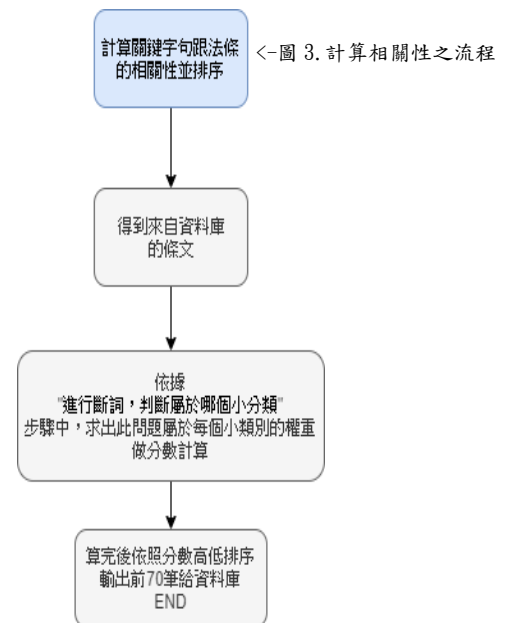


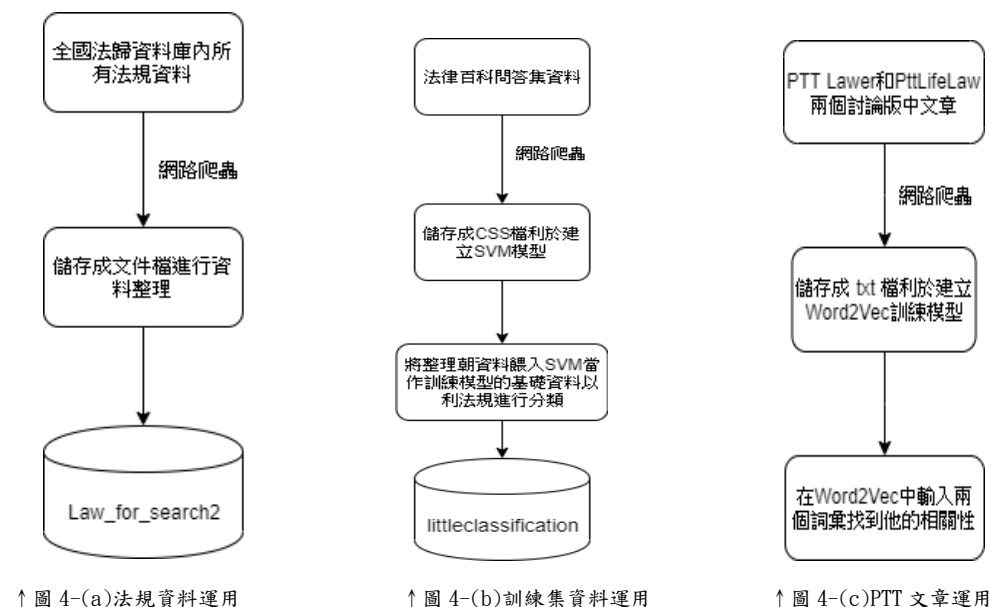
圖 3. 計算相關性之流程

獲取使用者輸入之關鍵字句之後，將關鍵字句透過停詞表將一些不重要的字詞刪除後再進行 Jieba 斷詞，將斷詞完成的結果進行 Word2Vec 找尋最相關的專業化字詞後再透過 SVM 將這些專業化字詞進行分析判斷是哪個分類如圖 2。

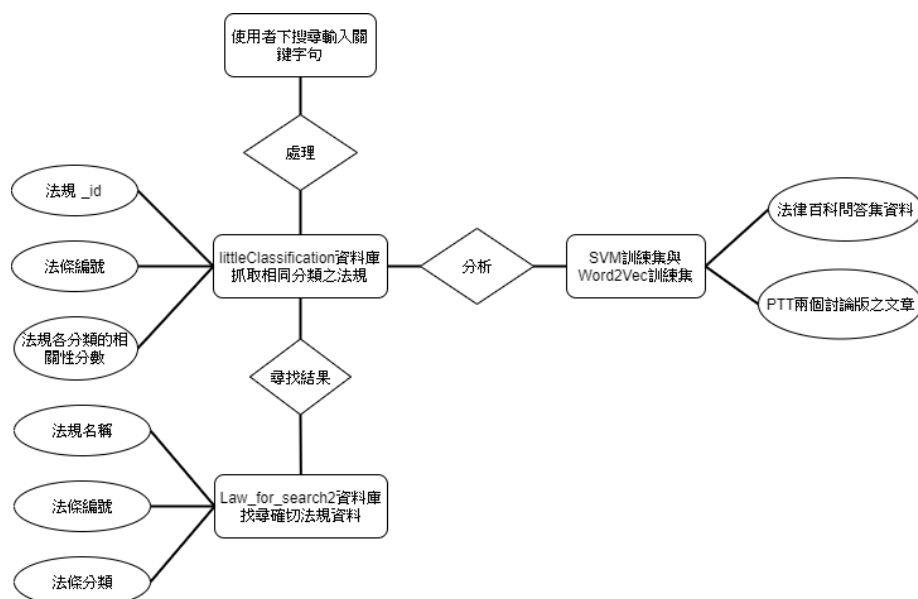
最後找到分類中的法條內容後再透過分數計算模型計算出每個條文與使用者輸入之關鍵字的相關性如圖 3，最後再將最相關的 70 筆法條輸出給使用者。

3.2 資料架構

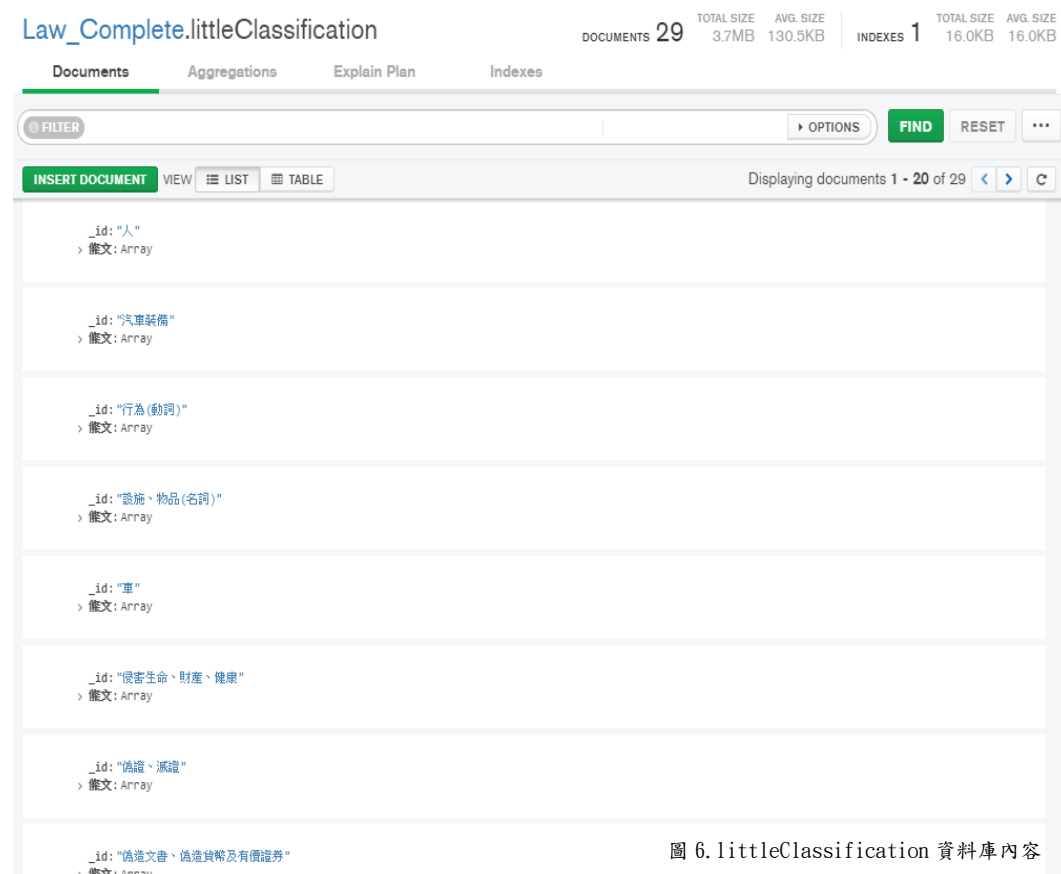
法規中的資料將進行整理與分析然後再放入 Law_for_search2 資料庫中如圖 4-(a)所示，法律百科問答集中的資料是用來使用 SVM 訓練集的如圖 4-(b)所示，而 PTT 討論版中的資料則是用來讓 Word2Vec 獲取裡面的文章讓我們輸入兩個字詞去計算相關性如圖 4-(c)所示。



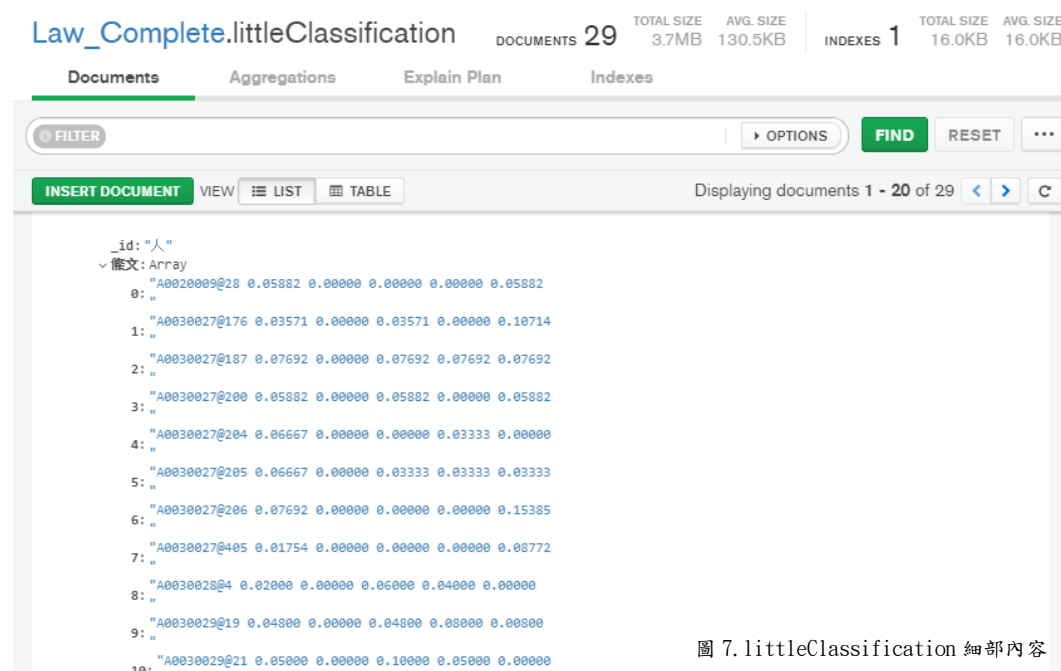
而實際上使用這些資料之資料關聯圖如圖 5 所示，littleClassification 資料中存放著的資料是小分類中有哪些法條紀錄的方式會如圖 6 所示，Law_for_search2 資料庫中所存方的資料為最後要輸出給使用者的完整法規資料存放的內容如圖 7 所示。



<-圖 5 資料關聯圖

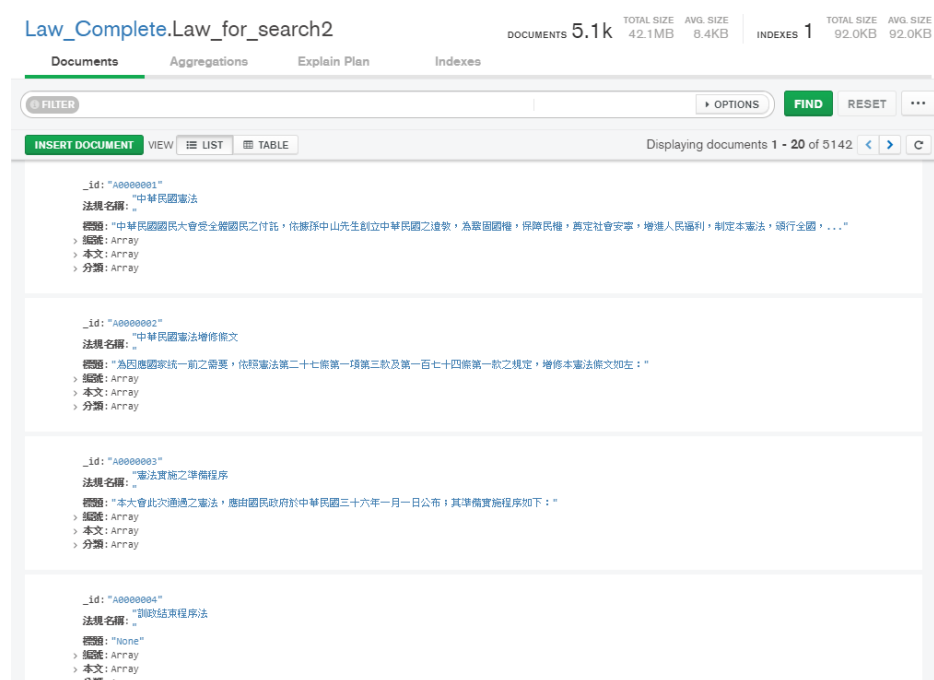


每個小分類的內容如圖 7 所示，每一筆都存放著該法規的_id、法條編號以及這個法規對於每個小分類的相關分數，以第 0 筆為例第一個空白前的資料為“_id@法條編號”中間的符號只是讓我們在資料傳述過程中有一個可判斷的依據，後面的其他資料均為對於每個小分類的相關性分數。



Law_for_search2 資料庫的內容共有 6 項分別為“_id”、“法規名稱”、“標題”、“編號”、“本文”、“分類”。以上這 6 點為我們整理完法規資料後認為對於我們系統最有幫助的 6 種資料

1. _id: 資料在 MongoDB 中必備的項目為每一筆資料的獨立編號，且各個資料的_id 不得重複。
2. 法規名稱: 法規的名字在結果中需要顯示給使用者。
3. 標題: 有些法規的最上端會有如大綱的資料如圖 7 紅框所示。
4. 編號: 法規中每個法條的編號整理成一個列表放入資料庫中。
5. 本文: 法規中每個法條的資料整理成一個列表放入資料庫中。
6. 分類: 經過 SVM 訓練集確認該條法規為哪個分類後放入資料庫中。



↑ 圖 8. Law_for_search2 資料庫內容



↑ 圖 9. 法規資料中對於整個法規的大綱描述

3.3 網路爬蟲

主要爬 3 個網站：

- 1、全國法規資料庫：所有法規資料的來源，爬下來後放入資料庫。
- 2、法律百科：用來訓練「法律專有名詞」與「口語化字詞」之間的關聯。
- 3、PTT 的 PttLifeLaw 和 Lawer：訓練「詞」與「詞」之間的關聯性。

爬蟲使用的 Python 套件如圖 10: BeautifulSoup4

圖 10-> `soup = BeautifulSoup(c, "html.parser")`

利用 BeautifulSoup4 對要爬蟲的網站的 HTML 進行分析，分析後找到我們要的內容的存放位置後，抓取下來存成 txt 檔。

```
<a href="#content" id="gotocenter" class="sr-only sr-only-focusable" title="跳  
至主要內容">跳至主要內容</a>  
<header>...</header>  
<!-- 主要內容 -->  
<div id="content" class="clearfix content">  
  ::before  
  <!-- 整合查詢 行動版 -->  
  <nav class="clearfix search-box-m visible-sm visible-xs">  
    <div class="container">...</div>  
  </nav>  
  <div class="container">...</div>  
  ::after  
</div>  
<!-- 主要內容結束 -->  
<!-- 頁尾 --> == $0  
<footer>...</footer>  
<script src="/js/bootstrap.min.js" type="text/javascript"></script>  
<script src="/js/jquery.cookie.js" type="text/javascript"></script>  
<script src="/js/app.js" type="text/javascript"></script>  
<script>...</script>  
<script src="/js/jquery.fancybox.min.js" type="text/javascript"></script>
```

↑ 圖 11. 網站的 HTML 碼

若有多個連續頁面需要進行爬蟲，則用迴圈來進行，以法規資料庫為例，以下為其中一頁的網址：

<https://law.moj.gov.tw/LawClass/LawAll.aspx?pcode=A0000001>

經過觀察，主要變動的為最後面的那串 A0000001，其中又分成了三段，分別為：

- 1、前面的英文字母
- 2、前 3 位數字
- 3、後 4 位數字

因此使用了 3 層迴圈成功的將資料完全爬下來。

3.4 網頁設計

主要使用到 HTML(網頁主架構)、CSS(網頁美化)

下圖 12 為首頁：



圖 12. 首頁介面

最中間提供欄位讓使用者輸入欲查詢的內容如圖 13 所示。

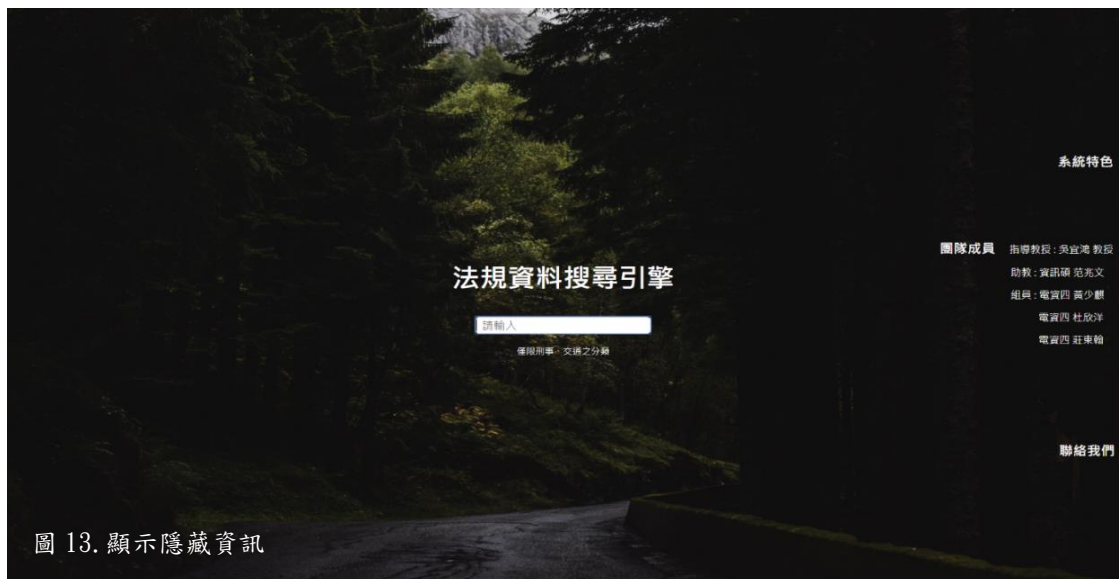


圖 13. 顯示隱藏資訊

若將滑鼠放到右邊的文字上，會顯示更多網站的資訊如圖 13 所示。

以下這個部分主要是使用 CSS 的 hover 功能

```
#contact:hover{  
    right: 10px;  
}
```

結果頁面如下圖 14 所示：



圖 14. 結果頁面之顯示

使用者下完查詢後會來來到搜尋結果頁面，我們將「相符的結果」以表格的方式呈現，其中淡橘色內的文字為「法規名稱」，下面為該法規中的：

1. 第幾條
2. 我們系統給他的分類
3. 法條內容。



圖 15. 上方搜尋欄位

若看了這些答案，衍生出了新的疑問，則可以使用上面的搜尋欄進行新的搜尋如圖 15 紅框所示。

3.5 伺服器架設

最一開始的做法是自己使用 python 架一個 server 如圖 16 所示，其中做了一些必備的功能如 Head、Get、delete。

```
22 class MyHandler(BaseHTTPRequestHandler):
23     def do_HEAD(self):...
37     #do_HEAD()
38
39     def do_GET(self):...
137     # do_GET()
138
139     def do_DELETE(self):...
154     #do_DELETE
155
```

圖 16. 使用 python 建立之 server

完成伺服器的架設後，其他使用者是有辦法連進我們的網站的，但之後遇到一個問題：我們做的是一個搜尋網站，因此我們的網站需要使用表單讀入使用者輸入的資料，這個部分需要使用到 PHP，而我們自己寫的伺服器無法支援 PHP。

因此我們決定使用 python 的套件來架設我們的伺服器，映入眼簾的為兩個套件：

- 1、Django
- 2、Flask

經過比較後，我們的理解是：Django 為一個很完整的架構，幾乎所有的功能都有模板可以使用，但使用起來較不靈活。相較之下，Flask 較為輕量，僅提供一些簡單的服務，但使用起來很靈活。

最後我們選擇了 Flask，其中有幾個原因：

1. 確實 Django 很強大，模板齊全，但我們需要用到的功能其實不多。
2. Flask 的高靈活度比較適合我們這次專題伺服器的需求。

Flask 中我們最先接觸到的為 route 指令如圖 17 所示

```
@app.route('/home', methods=['GET'])
def homepage():
    return render_template('index.html')
```

↑ 圖 17. route 指令實作

route 這個指令的用途主要對應到 url，使用者在網址中輸入東西後，我們的伺服器要做什麼相對應的動作，或是回傳什麼網頁回去給他。上圖的範例就是回傳我們的首頁給使用者。

接著介紹我們改用 flask 的最主要原因：讀入使用者所輸入的值

這裡分為兩個部分，分別為：

1. 前端(網頁 html) - 作一個表單，提供欄位讓使用者輸入
2. 後端(伺服器) - 接收使用者輸入的內容並作後續的處理

網頁端：

```
<form action="/" method="post">
<center>
  <input type="text" name="searchfor" style="fo
  <br>
  <br>
  <font size ="3px" color ="#FFFFFF">僅限刑事、
</center>
<BR>
</center>
</form>
```

↑ 圖 18. 我們首頁(index.html)中搜尋欄位的程式碼

其中「form」表示這是一個表單，可以將資料傳到我們伺服器中如圖 18 所示。「action」用來選擇對應的 route，最後我們再給這個輸入(input)欄位取一個名字(name)叫「searchfor」，讓伺服器有辦法取值。

伺服器端：

圖 19->

```
search = request.form.get('searchfor')
```

利用 request.form.get() 這個 function 從表單中取出使用者輸入的內容，並進行後續的處理如圖 19 所示。

3.6 資料庫系統

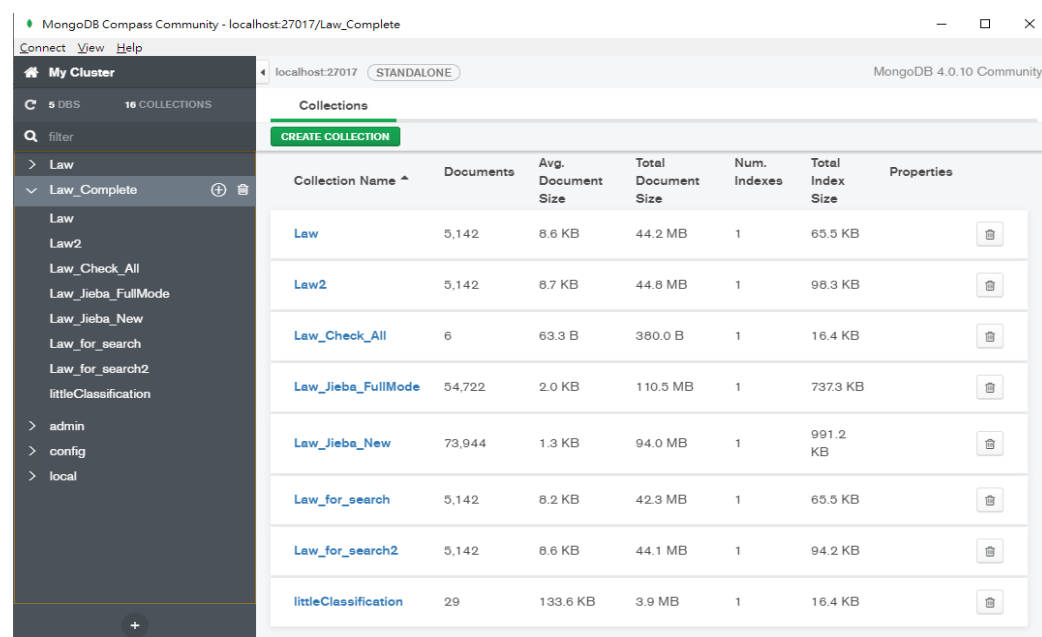
現在有許多的資料庫系統可以使用像是:SQLite、MySQL、MongoDB、Oracle...等，實驗中所選擇的是 MongoDB 這個資料庫系統，MongoDB 為一種文件導向的資料庫系統，對於存放法規這種全文字資料是有優點的，我們可以直接把法規中的資料全部放入。

```
from pymongo import MongoClient
import os

client = MongoClient()
db = client.Law_Complete
collection = db.Law2
```

圖 20. 與 MongoDB 之連接

圖 20 為透過 python 內建的套件 pymongo 用來與 MongoDB 的 Server 進行連接，db 變數為連結到 MongoDB 本機端的資料庫名稱，collection 變數則為該資料庫中的哪一個資料集內的資料。



The screenshot shows the MongoDB Compass Community interface. On the left, a sidebar lists the database 'Law_Complete' and its collections: Law, Law2, Law_Check_All, Law_Jieba_FullMode, Law_Jieba_New, Law_for_search, Law_for_search2, and littleClassification. The main panel displays a table of these collections with their respective statistics.

Collection Name	Documents	Avg. Document Size	Total Document Size	Num. Indexes	Total Index Size	Properties
Law	5,142	8.6 KB	44.2 MB	1	65.5 KB	[icon]
Law2	5,142	8.7 KB	44.8 MB	1	98.3 KB	[icon]
Law_Check_All	6	63.3 B	380.0 B	1	16.4 KB	[icon]
Law_Jieba_FullMode	54,722	2.0 KB	110.5 MB	1	737.3 KB	[icon]
Law_Jieba_New	73,944	1.3 KB	94.0 MB	1	991.2 KB	[icon]
Law_for_search	5,142	8.2 KB	42.3 MB	1	65.5 KB	[icon]
Law_for_search2	5,142	8.6 KB	44.1 MB	1	94.2 KB	[icon]
littleClassification	29	133.6 KB	3.9 MB	1	16.4 KB	[icon]

↑ 圖 21. MongoDB Compass 介面

使用 MongoDB Compass Community 來對資料進行可視化的管理如圖 21 所示，這個 MongoDB 官方的 GUI 工具可以讓我們更迅速清楚的知道資料擺放的位置和擺放的方式是否有符合我們自己預設的需求。

3.7 資料庫管理

原本的資料形式儲存在資料庫時有許多的分段依序是編、章、節、款、目最後才會到條文存放的地方如圖 22 所示，而是否有這麼多的分段也會根據法規而有不同的分段，因為我們的目的是將資料以法條的形式輸出給使用者所以我們將原始資料整理成單純條文的內容以便使用如圖 23 所示。內容僅存放法規的名稱以及該條文的編號與內容，圖 22 中之_id 為我們引用法規資料庫之網址所設計的資料 id。



圖 22. 經過網路爬蟲後所抓取下來的法規原始資料



圖 23. 修正後法規資料

3.9 句子數據化之方法

將文字轉換成數字數據的一種方法，利於分析如圖 29 所示。(TF:詞-單一句子, IDF:詞-整體性, TF-IDF:詞-兩者皆考慮)

句子1:每天最期待的就是吃飯 → “每天”，“期待”，“吃飯”
 句子2:記得要多吃飯菜 → “記得”，“吃飯”

TF(“吃飯”，句子1) = 1(出現“吃飯”次數)/3(斷詞數) = 0.333
 IDF(“吃飯”) = $\log(\text{總共幾個句子}/\text{幾個句子出現“吃飯”}) = \log(2/2) = 0$
 (表示“吃飯”在這兩句子中沒有指標性)

TF-IDF (單位：一個句子) = TF(單位：一個句子) x IDF (單位：一個數字，表整體數值)
 (同時考慮一個字在一個句子裡的占比 & 這個字在所有句子中是否具指標性)

←圖 29

接著看看實際的例子假設有兩個 句子已製作成表格如圖 30、31 所示：

Document 1	
Term	Term Count
this	1
is	1
a	2
sample	1

←圖 30

Document 2	
Term	Term Count
this	1
is	1
another	2
example	3

←圖 31

先計算出 this 的機率在兩個句子個別的機率，此為 tf

$$\text{tf}(\text{"this"}, d_1) = \frac{1}{5} = 0.2$$

$$\text{tf}(\text{"this"}, d_2) = \frac{1}{7} \approx 0.14$$

再來則是計算 idf

$$\text{idf}(\text{"this"}, D) = \log\left(\frac{2}{2}\right) = 0$$

最後再計算出總共的 tfidf 為 tf*idf

$$\text{tfidf}(\text{"this"}, d_1, D) = 0.2 \times 0 = 0$$

$$\text{tfidf}(\text{"this"}, d_2, D) = 0.14 \times 0 = 0$$

結果可知 this 在這兩個句子中沒有指標性，其他的字詞則依此類推。

3.10 口語化模型訓練：Word2Vec 訓練模型

Word2vec 為 Python 的套件，用來尋找詞的相關性，是口語化的詞轉換為專業用語的步驟。從 ptt 看板: PttLifeLaw 版、lawer 版爬蟲下來的文章經過斷詞、去除停詞後，剩下的詞彙(通常為名詞、動詞、形容詞)，形成一個陣列(表單)給予 Word2vec 訓練。依據給予的成千上萬篇文章，紀錄詞之間的關聯性，最後產生一個模型方便讀取。

讀取訓練好的模型如圖 32 所示：

```
modell = gensim.models.Word2Vec.load(PATH + "\\word2vec_model\\word2vec.model")
```

<-圖 32

找出“違停”最相關的關聯詞(十筆)，數值在 0 到 1 之間如圖 33、34 所示：

```
print("違停的相關詞")
array = modell.wv.most_similar("違停")
for i in range(len(array)):
    print("排名", i+1, " ", array[i])
```

<-圖 33

違停的相關詞

圖 34. 與違停最相關字詞

排名 1	('臨停', 0.9507654309272766)
排名 2	('黃線', 0.9269193410873413)
排名 3	('停靠', 0.9211543202400208)
排名 4	('路邊', 0.9199017286300659)
排名 5	('紅線', 0.9194145202636719)
排名 6	('車在', 0.9020031690597534)
排名 7	('警示燈', 0.8993890881538391)
排名 8	('駛出', 0.8963791131973267)
排名 9	('停好', 0.8948168754577637)
排名 10	('兩輛', 0.8933905363082886)

也能指定找出“詞”與“詞”之間的相關性(數值在 0 到 1 之間) 如圖 35、36：

```
num = modell.wv.similarity("違停", "停車")
print("違停與停車相關性: ", num)
```

<-圖 35

違停與停車相關性： 0.7047812125586803

↑ 圖 36. 兩個字詞之相關性

3.11 二分類訓練模型：SVM 訓練模型

SVM 為 Python 的套件，SVM 訓練完的模型可分辨一個條文是否屬於某個類別。訓練集資料來源為全國法規資料庫和法律百科，將這些文章中的句子進行去除停詞與斷詞後，剩下來的詞彙做 TF-IDF，一個小分類(中有許多屬於此小分類的詞)，每個小分類做一次 TF-IDF 得到一個向量(一個維度一個小分類)。並且在訓練時還要告知 SVM 套件這個分類是否屬於某種分類(事先由人判定)，若由人判斷過 100 篇文章，就能以這 100 篇文章訓練出來的基礎讓 SVM 自己判斷新來的更多文章(可能數千筆不等)，以達到自動化效果。

SVM 訓練的向量(文章 or 句子已數據化)如圖 37 所示：

```
向量: [0.0, 0.0, 0.0, 0.0, 0.0] , 是與否: 非交通類
向量: [0.0, 0.0, 0.0, 0.0, 0.0] , 是與否: 非交通類
向量: [0.0, 0.0, 0.1216259793, 0.036915665, 0.0530421273] , 是與否: 交通類
向量: [0.0, 0.0, 0.0, 0.0649442254, 0.0] , 是與否: 交通類
向量: [0.0, 0.0, 0.0, 0.0, 0.0] , 是與否: 非交通類
向量: [0.0, 0.0, 0.0, 0.0, 0.0] , 是與否: 非交通類
向量: [0.0447728209, 0.0, 0.0262601547, 0.0199260691, 0.0687136649] , 是與否: 交通類
向量: [0.0, 0.0, 0.0350135395, 0.0318817106, 0.0763485165] , 是與否: 交通類
向量: [0.0, 0.0, 0.0, 0.0, 0.0] , 是與否: 非交通類
向量: [0.0, 0.0, 0.0, 0.0, 0.0] , 是與否: 非交通類
向量: [0.0, 0.0, 0.0, 0.0, 0.0] , 是與否: 非交通類
向量: [0.0, 0.0, 0.0, 0.0, 0.0] , 是與否: 非交通類
向量: [0.0, 0.0, 0.0, 0.0, 0.0] , 是與否: 非交通類
向量: [0.0, 0.0, 0.0, 0.0, 0.0] , 是與否: 非交通類
向量: [0.02345243, 0.0, 0.0206329786, 0.0125249577, 0.0] , 是與否: 交通類
向量: [0.0273611683, 0.0, 0.0, 0.0584498029, 0.0] , 是與否: 交通類
向量: [0.0, 0.0, 0.0, 0.0350698817, 0.0377925157] , 是與否: 交通類
```

圖 37

將這些向量經過訓練形成模型如圖 37，與使用者問題向量一併做預測，得到結果如圖 39 所示。

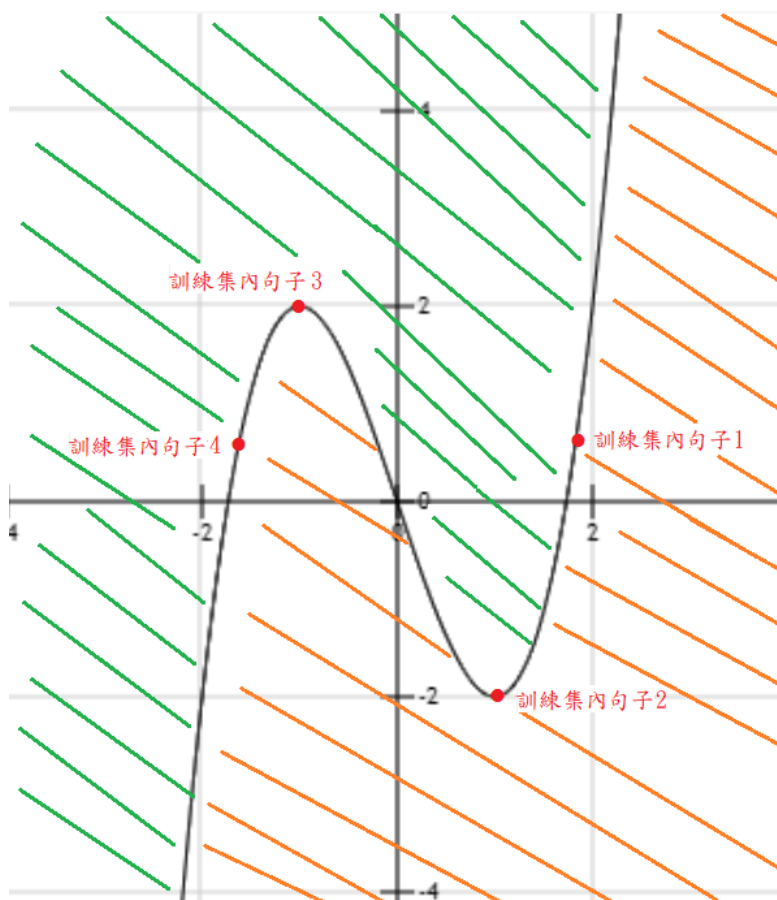
```
print("使用者問題轉換成向量：", YourQ)
pred = model.predict>YourQ)
print("結果：", pred)
```

圖 38

```
使用者問題轉換成向量： [[0.3283340198, 0.2855973665, 0.1444308505, 0.1753494085, 0]]
結果： ['交通類']
```

↑ 圖 39. 分析出哪個類

SVM 訓練好的模型觀念類似找到許多點連成的方程式(點為訓練集資料)如圖 40 所示，當今天來了一個新的問題時將問題數據化，類似變成一個座標。若在方程式上方則“屬於”此分類，反之則“不屬於”。



<-圖 40. SVM 之方程式圖

3.12 結果分數排序

1. 使用者的問題:發生車禍、除了向肇事者求償，還可以向誰求償?
2. 去除停詞、完成斷詞後: [“發生”，“車禍”，“肇事者”，“求償”，“求償”]。
3. 每個小分類(ex:行為、人、車)都有屬於此類別的詞彙，算出每個斷出來的詞彙與每個小分類中的每個詞彙的關係分數(利用 Word2Vec，詳情請看 P12)，算出每個小分類分別加總後，就能得知使用者問題中那些詞與那些小類較相關(較為重要)。

以上述為例：若求： [“發生”，“車禍”，“肇事者”，“求償”，“求償”] 中的“車禍”在“行為小類中”的分數如圖 41 所示。

```
車禍與 肇事 的關係分數: 0.5924488511753188
車禍與 超速 的關係分數: 0.466242470905638
車禍與 吊銷 的關係分數: 0.3346499908470576
車禍與 闖紅燈 的關係分數: 0.5217761824886984
車禍與 逆向 的關係分數: 0.513804089884639

車禍 在此小類中的分數總和: 2.4289215853013513
```

↑ 圖 41. 兩詞之相關性分數

行為小分類
肇事
超速
吊銷
闖紅燈
逆向

↑ 圖 42. 小分類中的字詞

```
車禍 向量: [2.4289215853013513, 0.6456764753188, 3.4641255705638, 0.51380437855, 0.133464999]
```

↑ 圖 43. 車禍與各種詞之相關分數所製成之向量

找出每個詞的向量後如圖 43 所示，將每個詞向量的同一小分類分數相加(直行)，即可得到每個小分類的分數如圖 42 所示，由此就能看出使用者問題較偏向哪一個小分類，並從資料庫中拿出占比較高小類中的法律條文(事先存於資料庫，每個條文的 TF-IDF 向量)。藉由使用者問題的向量，可得知分數分布如圖 44 所示(ex: 與分類 1 無關，與分類四最相關)。

```
使用者問題的分數分布: [7.530597113554334, 58.52479979289112, 68.6245072133791, 77.4154712271962, 41.946776640325524]
```

↑ 圖 44. 使用者問題在各種分類的分數

尋找占比方法為常態分布中的原則。計算出平均數(μ or \bar{X})後求得標準差(σ)，只要大於等於第一個標準差($\mu - \sigma$)的類別就會加入計算的範疇中。

資料來源: https://zh.wikipedia.org/wiki/68-95-99.7_原則
<https://zh.wikipedia.org/wiki/標準差>

第四章、 結論與未來方向

4.1 結論

將兩個系統比較後發現，我們在進行搜尋時的精確度還是較為低的，與全國法規資料庫相比，若是搜尋專業化用詞就可以明顯的發現我們的系統與全國法規資料庫所顯示的結果截然不同，會出現這樣的原因有一部分是因為兩者使用的是截然不同的搜尋方式，另一部分是分析使用者關鍵字句的方法不同導致分析出來的語意不同，就會發生這樣結果不同的問題，以下將搜尋一些關鍵字做對比。

關鍵字 1：賣國



<-圖 45. 全國法規資料庫之搜尋結果



<-圖 46. 全國法規資料庫點細項結果



<-圖 47. 我們系統之搜尋結果

我們在描述賣國，可能是指「出賣國家」與「背叛國家」的意思，應該是與刑法中的內亂與外患罪相關聯，應在結果中顯示刑法但在全國法規資料庫中卻因全文檢索的搜尋方式就只找到了證券相關的法規如圖 45、46 所示，因條文內有提到賣國兩個字就判斷為符合的結果。但我們的系統卻可成功搜尋到如圖 47。

關鍵字 2：闖紅燈



The screenshot shows the National Legal Information System (全國法規資料庫) search results for the keyword "闖紅燈". The search results are displayed in a table with columns for "法規名稱" (Legal Name) and "法規日期" (Legal Date). The results include:

法規名稱	法規日期
公立學校教職員退休資遣撫卹條例施行細則	(民國 107 年 05 月 14 日)
公務人員退休資遣撫卹法施行細則	(民國 107 年 03 月 21 日)
警察人員退休資遣撫卹法施行細則	(民國 107 年 03 月 21 日)
勞工保險被保險人因執行職務而致傷病審定準則	(民國 105 年 03 月 21 日)
農民職業災害保險被保險人因實際從事農業工作而致傷害審定辦法	(民國 107 年 10 月 17 日)
道路交通管理處罰條例	(民國 108 年 06 月 19 日)

↑ 圖 48. 全國法規資料庫搜尋結果



The screenshot shows the National Legal Information System (全國法規資料庫) search results for the keyword "闖紅燈". The search results are displayed in a table with columns for "法規名稱" (Legal Name) and "法規日期" (Legal Date). The results include:

法規名稱	法規日期
公立學校教職員退休資遣撫卹條例施行細則	(民國 107 年 05 月 14 日)
公務人員退休資遣撫卹法施行細則	(民國 107 年 03 月 21 日)
警察人員退休資遣撫卹法施行細則	(民國 107 年 03 月 21 日)
勞工保險被保險人因執行職務而致傷病審定準則	(民國 105 年 03 月 21 日)
農民職業災害保險被保險人因實際從事農業工作而致傷害審定辦法	(民國 107 年 10 月 17 日)
道路交通管理處罰條例	(民國 108 年 06 月 19 日)

↑ 圖 49. 細項結果



The screenshot shows the National Legal Information System (全國法規資料庫) search results for the keyword "闖紅燈". The search results are displayed in a table with columns for "法規名稱" (Legal Name) and "法規日期" (Legal Date). The results include:

法規名稱	法規日期
公立學校教職員退休資遣撫卹條例施行細則	(民國 107 年 05 月 14 日)
公務人員退休資遣撫卹法施行細則	(民國 107 年 03 月 21 日)
警察人員退休資遣撫卹法施行細則	(民國 107 年 03 月 21 日)
勞工保險被保險人因執行職務而致傷病審定準則	(民國 105 年 03 月 21 日)
農民職業災害保險被保險人因實際從事農業工作而致傷害審定辦法	(民國 107 年 10 月 17 日)
道路交通管理處罰條例	(民國 108 年 06 月 19 日)

↑ 圖 50. 我們的法規搜尋引擎之結果

如圖 48、49 為全國法規資料庫之搜尋結果，接下來則為我們的法規搜尋引擎之搜尋結果如圖 50 所示，可發現到結果有明顯的差異，在全國法規資料庫中除了道路交通處罰條例之外其他的幾條法規依名稱來看會覺得與交通類別的闖紅燈無關，但在我們的搜尋引擎中出現的結果均是與交通有關的。

關鍵字 3：連續殺人犯判什麼刑

	全國法規資料庫	查詢	輔助說明
	Law & Regulations Database of The Republic of China		

熱門詞彙：刑法、勞基法、憲法、公然侮辱、留職停薪

最新訊息	中央法規	司法判解	條約協定	兩岸協議	綜合查詢	跨機關檢索
------	------	------	------	------	------	-------

現在位置：[首頁](#) > [整合查詢查詢結果](#)

整合查詢查詢結果

推薦字詞：

護理人員什麼時候納入勞基法
連續請普通病假
違反兒童及少年性交易防治條例會受到什麼處分
為什麼要廢除兒少29條
十字杆
兒童與少年性交易防制條例
連續駕車超過八小時經查屬實，或患病足以影響安全駕駛

» 中央法規

- [法規名稱](#) 0
- [法條內容](#) 0

» 最新訊息

- [法律](#) 0
- [法規命令](#) 0
- [行政規則](#) 0
- [地方法規](#) 0
- [法規草案](#) 0
- [大法官解釋](#) 0

» 司法判解

- [大法官解釋](#) 0
- [最高法院民事判例](#) 0
- [最高法院刑事判例](#) 0
- [最高行政法院判例](#) 0

» 條約協定

- [條約協定名稱](#) 0
- [條約協定內容](#) 0

📍 中央法規 > 法規名稱

您所輸入的查詢條件查無資料，請重新設定查詢條件！

↑圖 51. 全國法規資料庫中輸入較為口語化之字句的結果

	<div data-bbox="306 1097 962 1142">  法規搜尋網站 <input data-bbox="505 1120 962 1140" type="text" value="關鍵字"/> </div> <div data-bbox="545 1160 679 1205"> 共70筆符合資料 第1頁/共7頁 </div> <div data-bbox="470 1240 1208 1657"> <table border="1"> <thead> <tr> <th colspan="2">中華民國刑法</th> </tr> </thead> <tbody> <tr> <td>第64條</td> <td>分條·刑罰原則(刑罰)、刑罰(刑罰)。</td> </tr> <tr> <td colspan="2">死刑不得加重，死刑減輕者，為無期徒刑。</td> </tr> <tr> <td>第65條</td> <td>分條·刑罰原則(刑罰)、刑罰(刑罰)。</td> </tr> <tr> <td colspan="2">無期徒刑不得加重，無期徒刑減輕者，為二十年以下十五年以上有期徒刑。</td> </tr> <tr> <td>第66條</td> <td>分條·刑罰原則(刑罰)、刑罰(刑罰)。</td> </tr> <tr> <td colspan="2">未滿十八歲人或滿八十歲人犯罪者，不得處死刑或無期徒刑，本刑為死刑或無期徒刑者，減輕其刑。</td> </tr> <tr> <td>第271條</td> <td>分條·侵害生命、財產、健康(刑罰)、刑法原則(刑罰)、刑罰(刑罰)。</td> </tr> <tr> <td colspan="2">殺人者，處死刑、無期徒刑或十年以上有期徒刑。前項之未遂犯罰之，預備犯第一項之罪者，處二年以下有期徒刑。</td> </tr> <tr> <td>第101條</td> <td>分條·刑法原則(刑罰)、刑罰(刑罰)。</td> </tr> <tr> <td colspan="2">以暴動犯前條第一項之罪者，處無期徒刑或七年以上有期徒刑。竊匪者，處死刑或無期徒刑，預備或強盜犯前條之罪者，處一年以上七年以下有期徒刑。</td> </tr> <tr> <td>第334條</td> <td>分條·侵害生命、財產、健康(刑罰)、刑法原則(刑罰)、刑罰(刑罰)。妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)。</td> </tr> <tr> <td colspan="2">犯海盜罪而致殺殺人者，處死刑或無期徒刑。犯海盜罪而有下列行為之一者，處死刑、無期徒刑或二十年以上有期徒刑：一、放火者，二、強制性交者，三、擄人勒贖者，四、使人墮落者。</td> </tr> <tr> <td>第332條</td> <td>分條·侵害生命、財產、健康(刑罰)、公共危險(刑罰)、刑法原則(刑罰)、刑罰(刑罰)。妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)。</td> </tr> <tr> <td colspan="2">犯海盜罪而致殺殺人者，處死刑或無期徒刑。犯海盜罪而有下列行為之一者，處死刑、無期徒刑或二十年以上有期徒刑：一、放火者，二、強制性交者，三、擄人勒贖者，四、使人墮落者。</td> </tr> </tbody> </table> </div>	中華民國刑法		第64條	分條·刑罰原則(刑罰)、刑罰(刑罰)。	死刑不得加重，死刑減輕者，為無期徒刑。		第65條	分條·刑罰原則(刑罰)、刑罰(刑罰)。	無期徒刑不得加重，無期徒刑減輕者，為二十年以下十五年以上有期徒刑。		第66條	分條·刑罰原則(刑罰)、刑罰(刑罰)。	未滿十八歲人或滿八十歲人犯罪者，不得處死刑或無期徒刑，本刑為死刑或無期徒刑者，減輕其刑。		第271條	分條·侵害生命、財產、健康(刑罰)、刑法原則(刑罰)、刑罰(刑罰)。	殺人者，處死刑、無期徒刑或十年以上有期徒刑。前項之未遂犯罰之，預備犯第一項之罪者，處二年以下有期徒刑。		第101條	分條·刑法原則(刑罰)、刑罰(刑罰)。	以暴動犯前條第一項之罪者，處無期徒刑或七年以上有期徒刑。竊匪者，處死刑或無期徒刑，預備或強盜犯前條之罪者，處一年以上七年以下有期徒刑。		第334條	分條·侵害生命、財產、健康(刑罰)、刑法原則(刑罰)、刑罰(刑罰)。妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)。	犯海盜罪而致殺殺人者，處死刑或無期徒刑。犯海盜罪而有下列行為之一者，處死刑、無期徒刑或二十年以上有期徒刑：一、放火者，二、強制性交者，三、擄人勒贖者，四、使人墮落者。		第332條	分條·侵害生命、財產、健康(刑罰)、公共危險(刑罰)、刑法原則(刑罰)、刑罰(刑罰)。妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)。	犯海盜罪而致殺殺人者，處死刑或無期徒刑。犯海盜罪而有下列行為之一者，處死刑、無期徒刑或二十年以上有期徒刑：一、放火者，二、強制性交者，三、擄人勒贖者，四、使人墮落者。	
中華民國刑法																															
第64條	分條·刑罰原則(刑罰)、刑罰(刑罰)。																														
死刑不得加重，死刑減輕者，為無期徒刑。																															
第65條	分條·刑罰原則(刑罰)、刑罰(刑罰)。																														
無期徒刑不得加重，無期徒刑減輕者，為二十年以下十五年以上有期徒刑。																															
第66條	分條·刑罰原則(刑罰)、刑罰(刑罰)。																														
未滿十八歲人或滿八十歲人犯罪者，不得處死刑或無期徒刑，本刑為死刑或無期徒刑者，減輕其刑。																															
第271條	分條·侵害生命、財產、健康(刑罰)、刑法原則(刑罰)、刑罰(刑罰)。																														
殺人者，處死刑、無期徒刑或十年以上有期徒刑。前項之未遂犯罰之，預備犯第一項之罪者，處二年以下有期徒刑。																															
第101條	分條·刑法原則(刑罰)、刑罰(刑罰)。																														
以暴動犯前條第一項之罪者，處無期徒刑或七年以上有期徒刑。竊匪者，處死刑或無期徒刑，預備或強盜犯前條之罪者，處一年以上七年以下有期徒刑。																															
第334條	分條·侵害生命、財產、健康(刑罰)、刑法原則(刑罰)、刑罰(刑罰)。妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)。																														
犯海盜罪而致殺殺人者，處死刑或無期徒刑。犯海盜罪而有下列行為之一者，處死刑、無期徒刑或二十年以上有期徒刑：一、放火者，二、強制性交者，三、擄人勒贖者，四、使人墮落者。																															
第332條	分條·侵害生命、財產、健康(刑罰)、公共危險(刑罰)、刑法原則(刑罰)、刑罰(刑罰)。妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)、妨害自由(刑罰)。																														
犯海盜罪而致殺殺人者，處死刑或無期徒刑。犯海盜罪而有下列行為之一者，處死刑、無期徒刑或二十年以上有期徒刑：一、放火者，二、強制性交者，三、擄人勒贖者，四、使人墮落者。																															

↑圖 52. 我們的系統輸入較為口語化之字句的結果

在全國法規資料庫中如果搜尋口語化的字詞，像是連續殺人犯判什麼刑、我看到殺人案、違停、撞到人賠多少錢．．．等，較為日常的用詞能會顯示無查詢條件相符的結果如圖 51 所示，而在我們的法規資料庫中則會依據你所輸入的關鍵字句去判斷是哪一個分類再將結果顯示出來如圖 52 所示。

4.2 未來方向

4.2.1 系統改善

1. 在上述的結果與多張圖示可以發現，我們的系統在一些關鍵字的搜尋下會出現一些極為相同的結果，會出現這樣的結果是因為為了節省在眾多法條中尋找相關性讓使用者可以迅速的拿到想要的結果，所以在系統後端有一份預先計算好各個法規在這個分類的相關性分數，只要找出關鍵字句屬於哪一個分類就可以快速的找到結果回傳給使用者。

2. 在最後的計算相關性分數的部分，在經過多次的搜尋與測試之後我們發現了都會有固定的幾條特別短的法規會顯示在結果頁面的最前段，這很有可能是因為句子的長度較短(Jieba 出來的詞彙少於5)但在這個短句中卻有出現1到2個斷詞是屬於這個分類的所以導致他的相關分數就會特別高。反之若為長句子(Jieba 出來的詞彙多於10)則會因為屬於該分類的詞彙較少導致使用者可能需要的結果被排到較為後段的地方。

4.2.2 系統擴展

1. 目前的系統僅限於交通與刑事類別的查詢，未來希望可以將所有法規擴充至我們的系統中，讓使用者可以方便的查詢法律相關知識。

2. 未來可以與一些線上的法律事務所做結合，然後透過製作一個對話機器人讓使用者可以在真的走投無路的情況下還有一個可以解決問題的出路，不會因為到處都沒有可以解決問題的方法然後就要花大筆的律師費去請律師解決問題。

3. 提拱一些新聞上有出現過的案例，建立成另外一個搜尋模式，讓使用者可以在搜尋法規之餘也可以查找到與自己類似的案例，並看看以其他案例而言法規是怎麼判定疏失等類的。

第五章、 參考資料

5.1 使用資料

法規資料：[全國法規資料庫](#)

口語化訓練資料：[法律百科](#)

停詞表：[停詞表](#)

5.2 官方工具

MongoDB：<https://www.mongodb.com/>

Flask：<https://www.jianshu.com/p/f644330c596a>

MongoDB Compass：<https://www.mongodb.com/products/compass>

5.3 使用套件

SVM：<https://medium.com/jameslearningnote/資料分析-機器學習-第3-4講-支援向量機-support-vector-machine-介紹-9c6c6925856b>

Word2Vec：<https://medium.com/pyladies-taiwan/自然語言處理入門-word2vec小實作-f8832d9677c8>

Jieba：<https://github.com/fxsjy/jieba>

第六章、 附錄

6.1 系統展示影片的初步問卷調查(大二 DS 修課學生)

評價	比例
十分糟糕	0/93
乏善可陳	1/93
普普通通	23/93
感覺良好	44/93
超過 100 分	24/93

↑ 表 1. 修課學生之評價，分 5 個等級

負評

1. 口語化會隨時間產生一定的變化
2. 較無良好介面、較不常用
3. 本身對於專題的興趣感不高，而且感覺實用性比較低
4. 我覺得我在生活上用不到
5. 就沒有甚麼特別的
6. 感覺很普通 而且影片沒字幕
7. 其實與大多搜尋引擎方式雷同
8. 講者背景有雜音
9. 那個背景音真的有點不行，不過我覺得那個透過口語化的搜索還滿不錯的

正評

1. 很多時候，打得太通俗都找不到法規，有了這個系統，明顯的讓較不懂法規的人有查找的地方
2. 讓我可以輕鬆查詢我需要的法律。
3. 貼近生活實際
4. 簡單方便
5. 需要查詢法律的資訊時很方便
6. 是蠻實用 而且很獨特
7. 有著極強的目的性，還有專業性
8. 不常見，有特色
9. 能夠方便的查詢各種法規 即便是疑問句也能夠查到問題的相關法規
10. 雖然目前我用不到，但是以社會人士來說，真的非常有幫助
11. 很有特色且實用，但字詞分類不夠精細有點可惜。

6.2 問卷評論結語

在負評中可以明顯的看到除了針對影片品質(6、8、9)之外的評論大多數都是覺得在生活上用不到或是任為這個系統在日常生活中較為普遍且沒有特色，但我們在蒐集資料階段時卻發現並不如學弟妹所認為的那樣非常普遍，雖然說大部分網路上有很多的系統可以支援法規搜尋但他們並不能用普通的口語化方式去進行搜尋，若輸入太過於通俗的關鍵字很容易找不到結果。

法規類別	<input checked="" type="checkbox"/> 全選 (主管法規) <input checked="" type="checkbox"/> 組織目 <input checked="" type="checkbox"/> 勞動福祉退休目 <input checked="" type="checkbox"/> 勞動檢查目 <input checked="" type="checkbox"/> 其他目	<input checked="" type="checkbox"/> 勞動關係目 <input checked="" type="checkbox"/> 勞動條件及就業平等目 <input checked="" type="checkbox"/> 職業訓練目	<input checked="" type="checkbox"/> 勞動保險目 <input checked="" type="checkbox"/> 職業安全衛生目 <input checked="" type="checkbox"/> 就業服務目
檢索字詞	<input type="text" value="檢索字詞設定方式，敬請參閱 (輔助說明)"/> <input type="button" value="輔助說明"/>		
期 間	自民國 <input type="text" value="YYY"/> 年 <input type="text" value="MM"/> 月 <input type="text" value="DD"/> 日 至 <input type="text" value="YYY"/> 年 <input type="text" value="MM"/> 月 <input type="text" value="DD"/> 日		
發文文號	<input type="text" value="請填入發文文號的號 (半形數字)"/> *本項適用於檢索「實質法規命令」時查詢 *例如：勞動條2字第1060131805號公告，請填「1060131805」查詢		
法規位階	<input checked="" type="checkbox"/> 全選 <input checked="" type="checkbox"/> 法律 <input checked="" type="checkbox"/> 法規命令 <input checked="" type="checkbox"/> 實質法規命令		
法規效力	<input checked="" type="checkbox"/> 現行法規 <input type="checkbox"/> 廢止法規		

↑ 圖 53. 勞動部法令查詢系統之搜尋介面

以[勞動部勞動法令查詢系統](#)為例你需要知道確切法規資訊你才可以找到你想要的法規如圖 53 所示，但以一般民眾對於法規知識的認知較為短缺的情況下很難找到需要的資料，雖然目前我們僅限於刑事與交通類別之法規但未來希望可以拓展至全法規均適用此系統。

在正面評價很感謝學弟妹們的肯定都有看到我們一開始製作此系統的初衷，之後我們會努力朝向全法規均可使用口語化的搜尋，不再僅限於刑事與交通類別，我們也會盡力的將介面設計的更加美觀與實用。

6.3 分工與成員心得

6.3.1 分工表

黃少麒	資料庫建立、結果分析
莊東翰	資料分析、模型訓練
杜欣洋	爬蟲、伺服器架設、網頁架設

↑ 表 2. 成員之分工

6.3.2 收穫與貢獻

1. 黃少麒

這次的專題實驗讓我學習到了許多不僅僅是在程式方面的技巧更多的是團隊合作上的技巧，不僅僅是口頭上的溝通，更多的是在程式碼上的交流，一個系統中分成了好幾個部分大家各自去研究製作最後再將全部合併起來，最困難的部分莫過於在每個不同的項目中運用到同一個資料，但是用法不同時需要不斷地經過討論與研究才能找到最好的方法，而在一個團隊中不是只要熟悉自己的部分而是要每個不分都能了解即便那不是你負責的區域但也必須要了解運作原理這才是團隊合作，我的工作主要是管理資料庫，一開始在拿到資料後有點徬徨，因為在一開始對資料庫的概念就只有把很多資料全部整理好放進去，但怎麼使用資料庫一直都很無解，那時候就不斷查了很多不同的資料，怎麼抓取巢狀結構資料，怎麼擺放巢狀結構資料，法規中的資料有很多不同的形式，有些是巢狀結構有些不是，經過了不斷的嘗試才成功的把資料放入資料庫。

2. 杜欣洋

這次專題的一開始我感到很大的壓力，因為我的工作負責蒐集資料，但當時我連爬蟲的概念都沒有，在上網查了一些資料後，慢慢體會到了python厲害的地方-有很多套件可以使用，但不僅僅是學習如何使用它，我也試著了解他們運作的方法、背後的原理。

在專題中我也學到學會了如何寫一個網站，從完全不認識HTML、CSS，到最後成功寫出一個還算滿意的首頁和結果頁面，在寫網頁的過程中我也體會出團隊合作的好處，在摸不著頭緒的時候有人可以給你想法，大家一起思考怎樣的呈現方式會更好。

而團隊合作也是我認為這次專題中最大的收穫，以往的作業往往是自己做自己的，資料的儲存、傳遞方式也是由自己全權決定。但在這次的專題中就完全不

是這樣了，幾乎所有的東西都需要經過討論，從一開始的爬蟲，儲存檔案的格式要用 txt 或是 excel，到後面寫網頁時哪些資訊需要顯示在頁面上。雖然效率可能不及自己一個人時，但合作的優點還是很多的，有其他人幫你一起分擔工作，每個人負責自己擅長的部份、在不知道下一步該怎麼走時找大家一起討論、遇到問題時集思廣益找出最好的解決辦法……整體來說我很喜歡這次的團隊合作，大家朝同一個目標一起努力的感覺。

3. 莊東翰

這次專題實我主要負責的是文字分析的部分，經過這次專題我學習到了基礎的文字分析方法與流程，從一個句子要如何取出有用的字彙、如何將文字數據化和如何運用訓練集訓練模型。在初期學習使用訓練模型時不斷碰壁，原因是我無法理解訓練模型到底背後是如何幫我做決定的。當我認知到文字訓練模型是一個黑盒子，就像幫你在一個座標空間幫你找連線所有點的方程式，它會在背後幫你做好計算，一切就迎刃而解了。這次的實驗也讓我了解中文字分析的瓶頸，中文可能一個詞會有多種意思，無法輕易辨別是何種意思，就像在“全國法規資料庫”搜尋“賣國”卻跑出“買賣國內公債”。還有就是中文找不太到一個完美的斷詞套件，因為許多套件都詞庫過少，且斷詞的精準度也有待提升。這些文字分析背後許多的工具我認為沒有到很健全，所以這也會間接影響到分析的準確度。