基於人工智慧協助用戶諮詢民法系統開發之系統文書

[1.摘要 1](#_Toc201243182)

[2.緒論 1](#_Toc201243183)

[3.系統概要設計 5](#_Toc201243184)

[4.系統開發工具與使用環境 8](#_Toc201243185)

## 1.摘要

隨著人工智慧技術的進步與法律資源數位化的普及，越來越多民眾期望能透過科技手段獲得即時、初步的法律諮詢。然而，目前一般民眾在面臨法律問題時，常因資訊不足、用語艱澀或流程複雜而無從下手。因此，本專題旨在開發一套結合自然語言處理技術（NLP）與互動式導引機制的「法律案件導引 AI 系統」，協助使用者釐清案情，快速辨識其可能涉及的法律領域，並提供後續的行動建議與相關法規資訊。

本系統採用對話式介面，透過問題導向流程與語意理解模型，與使用者互動式收集案件內容，並以關鍵字抽取與語意比對技術，推斷可能涉及之法律類別（如勞資糾紛、家庭關係、契約爭議等）。系統進一步串接預建法律知識庫與法條摘要模組，提供相對應的法規內容與白話解釋，讓使用者能更容易理解其法律處境與應對選項。此外，本專題亦導入案件分類模型與行動建議模組，給予使用者「是否建議尋求法律扶助」、「可聯繫之單位」、「建議保存之證據」等實用資訊。

本系統設計強調使用者導向、語意理解與資訊可信度，期望能降低法律資訊的進入門檻，作為未來法律資訊平台或法律輔助系統的雛型，提升一般大眾對法律知識的可近性與應用性。

## 2.緒論

1. 背景介紹

隨著法律意識抬頭，民眾面對法律問題的需求日益增加，從勞資糾紛、租屋爭議、家庭問題到網路糾紛，日常生活中與法律相關的事件層出不窮。然而，對於沒有法律背景的一般人而言，往往難以理解法律條文、無法判斷是否違法，甚至不知道該向誰尋求協助。雖然目前已有部分法律機構或網站提供線上諮詢服務，但多數屬於靜態 FAQ 或需等待真人回覆，缺乏即時性與個人化互動。

1. 專題的目的及重要性

本專題旨在建構一套互動式的「法律案件導引 AI 系統」，協助使用者透過自然語言對話，快速釐清案件輪廓、辨識可能涉及的法律問題，並提供相關法條與初步建議方向。透過 AI 對話引導，降低法律資訊的理解門檻，提高民眾接觸法律知識的機會與效率。本系統將結合 NLP 技術與法律知識庫，實現具備「引導、判別、解釋、建議」功能的智慧輔助平台。

此專題對於推動智慧法律服務、促進法律平權與未來數位法律輔助系統的設計具有實質意義。

1. 主要研究問題或目標

 **如何讓 AI 理解使用者輸入的自然語言案情？**

 **如何辨識案件所涉及的法律領域（如民事、勞動、家庭等）？**

 **如何提供使用者易於理解的法條摘要與建議行動？**

 **如何設計一個互動流程，讓使用者能快速找到問題的法律定位？**

1. **系統功能簡介**

| **模組名稱** |  | **功能說明** |
| --- | --- | --- |
| 案件輸入與語意理解模組 |  | 分析使用者以自然語言描述的法律問題，進行語意向量化與關鍵詞抽取。 |
| 案件類型判別模組 |  | 利用分類模型或規則比對，推斷所涉及的法律領域（如：勞基法、民法、刑法）。 |
| 法條檢索與白話解釋模組 |  | 從法律知識庫中找出相關法條，並以 AI 模型生成白話摘要。 |
| 行動建議與導引模組 |  | 根據案情提供可能後續行動（如：調解、報警、申訴）、建議保存資料或聯繫單位。 |
| 對話式互動介面 |  | 使用類似 Chatbot 的方式與使用者互動，收集資訊並即時回應。 |

1. 系統使用對象

本系統設計目標族群如下：

* **一般民眾**：遇到生活法律問題但無法律背景者。
* **學生族群**：如高中、大學生遇到租屋、網路糾紛等。
* **社會新鮮人**：進入職場後遇到勞資爭議、解雇、合約問題。
* **法律扶助前導詢者**：可作為向法律機構諮詢前的初步資訊工具。

1. 系統特色

1.**語意理解導向的互動流程**：不同於傳統 FAQ，本系統可根據使用者輸入案情進行語意分析與互動導引。

2.**多樣案件類型支援**：涵蓋民事、勞動、家庭等常見法律領域，靈活擴充。

3.**法條白話摘要**：透過 AI 模型將正式條文轉譯為易懂語句，提升可讀性。

4.**行動建議提供**：不僅指出問題，還給出可行行動方案（如聯繫勞工局、保存證據）。

5.**前後端分離架構**：易於部署與未來擴充功能，如加入地區法律資源串接、語音輸入等。

3.相關技術應用與重要文獻

為了實現本專題所需的互動式法律導引系統，我們參考了以下幾種技術與既有應用成果：

**1. 法律問答系統（Legal QA Systems）**

**代表研究/應用：**

**台灣法務部法律諮詢系統**：提供靜態問答與法條查詢功能。

**LegalQA 中文法律問答資料集（Tian et al., 2021）**：提供大量法律問答對，為語意理解與匹配任務提供資料基礎。

優點：

專注於法律領域，具法條與實務案例基礎

問答資料結構明確，適合模型訓練與匹配

缺點：

缺乏互動式導引，無法針對個案情境做推理或引導提問

部分系統仍仰賴關鍵字而非語意理解，導致準確率受限

**2. 法律知識檢索與白話翻譯技術**

**代表研究/應用：**

**Legal-BERT (Chalkidis et al., 2020)**：專為法律語言訓練的 BERT 模型，可用於法條語意分析、文件分類

**司法院法規資料庫轉換與摘要任務（台灣研究團隊）**：針對條文進行語意摘要與簡化翻譯

優點：

可處理大量法條文本，支援關鍵字與語意相似度檢索

BERT 模型能進行條文語句的白話翻譯與語意近似分析

缺點：

中文法律語料訓練資源較少，效果可能不如英文系統

模型推論速度相對慢，不易在低資源設備即時部署

**3. Chatbot 對話系統與決策導引流程**

**代表研究/應用：**

**對話式決策支援系統（Decision Support Systems with Dialog Management）**

**Rule-based chatbot with decision tree**：常見於醫療、心理諮詢等初步導引領域

優點：

可引導使用者逐步澄清問題情境，適用於判斷流程式問題

介面親民、易於操作，降低使用門檻

缺點：

規則太多時易變得繁瑣、僵硬

無法應對太自由、不規則的語句，對話靈活性受限

**4. RAG 技術（Retrieval-Augmented Generation）**

**技術來源：**

由 Facebook AI 提出的結合知識檢索與文本生成的架構

在法條回答任務中已被應用於生成「依據法條回答」的回應

優點：

可以即時從資料庫中拉出法條作為模型依據，再由語言模型產出答案

結合法規資訊與自然語言生成，回應更準確且具可讀性

缺點：

開發成本較高，需要資料庫建設與模型協同運作

回應品質受限於檢索品質與語言模型理解力

**綜合比較與本專題選擇理由**

| **技術/應用** | **與本專題相關性** | **優點** | **缺點** | **對本專題的採用情形** |
| --- | --- | --- | --- | --- |
| 法律問答系統 | 高 | 可作為知識來源 | 缺乏互動性 | 作為資料基礎 |
| Legal-BERT | 高 | 語意理解佳 | 中文模型效果較低 | 可用於語意比對與條文摘要 |
| Chatbot + 決策導引 | 非常高 | 使用者導向、引導性強 | 回答靈活度有限 | 作為核心導引架構 |
| RAG 架構 | 中高 | 回應具條文依據 | 開發難度較高 | 若時程允許可加入生成功能 |

**小結:**

本專題綜合採用 NLP 語意比對（如 Legal-BERT）、問答資料檢索、對話流程引導與法條白話摘要等技術，並結合 Chatbot 框架與案件分類模組，設計一套具備「理解 → 導引 → 解釋 → 建議」完整流程的互動式法律輔助系統。相較於傳統靜態系統，本系統更強調使用者參與與實務導向，有望作為未來法律導引系統的原型平台。

## 3.系統概要設計

3-1 本專題所採用的研究或開發方法及技術

本專題採用結合自然語言處理（NLP）、分類演算法與對話式導引的開發方法，目的是打造一個能理解案情並回應法律建議的互動式系統。

開發流程採用漸進式模組設計（Incremental Module Development），先從語意比對與分類模組建立原型，再結合法條摘要與回應模組進行整合測試。

3-2 工具和技術的選擇理由

| 技術 / 工具 | 用途 | 選擇理由 |
| --- | --- | --- |
| Python（Flask） | 後端伺服器與 API | 輕量、易於串接 NLP 模型 |
| HTML/CSS + Vue.js 或 React | 前端使用者介面 | 動態互動佳，適合製作對話式界面 |
| MongoDB 或 SQLite | 儲存問答記錄與法條資料 | 彈性大、資料結構自由 |
| Hugging Face Transformers (BERT) | 處理自然語言與語意比對 | 可使用中文預訓練模型（如 Legal-BERT） |
| Node-RED（可選） | 用於設計導引流程圖 | 可視化流程設計，降低開發門檻 |

3-3 處理流程（文字說明）

使用者輸入案情描述（自然語言）

系統進行語意處理（BERT embedding）並進行分類預測（案件類別）

對應分類，呼叫相關法律知識庫內容（如：法條、案例）

法條摘要模組將法律條文轉換成白話易懂語句

對話導引模組詢問更多案情細節（如時間點、證據狀況等）

系統綜合分析後回傳初步法律方向建議（如：可能違法、建議聯繫單位）

儲存本次紀錄並提供使用者下載、列印或轉寄建議內容

3-4 檔案關聯與模組結構

| 檔案/模組名稱 | 功能說明 |
| --- | --- |
| app.py | Flask 主伺服器與 API 控制中心 |
| bert\_model.py | 處理語意嵌入與分類模型 |
| dialog\_flow.json | 導引對話流程設定（規則或條件分支） |
| laws.db | 法條與案例資料庫（可為 SQLite） |
| frontend/ | 前端介面（使用者輸入、訊息互動） |
| log.json | 儲存使用者問題與系統回應紀錄 |

3-5 系統設計圖表（示意版）

（1）系統架構圖（System Architecture）

使用者 ↔ 前端（React/Vue） ↔ Flask API

↓

┌──────────────┬───────────────┐

NLP 模型 法條知識庫 案件導引規則

↓ ↓ ↓

語意比對 條文檢索與摘要 決策建議輸出

（2）流程圖（Flow Chart）

[開始]

↓

[輸入案情描述]

↓

[語意分析（BERT）]

↓

[判斷案件類型]

↓

[導引對話蒐集案情細節]

↓

[檢索法條並生成白話摘要]

↓

[提供行動建議與相關資料]

↓

[紀錄問答歷程]

↓

[結束]

（3）使用案例圖（Use Case Diagram）

Actor: 使用者

Use Cases:

輸入法律問題

接收法條摘要

查看初步建議

儲存諮詢紀錄

查詢歷史紀錄

（4）甘特圖（Gantt Chart，階段建議）

| 時間 | 項目 | 說明 |
| --- | --- | --- |
| 第 1~2 週 | 需求分析與技術選型 | 決定使用哪些 NLP 模型、資料庫 |
| 第 3~5 週 | 系統原型開發（語意比對 + 簡易 UI） | 前後端串接 |
| 第 6~8 週 | 案件分類模組 + 法條摘要模組整合 | 加入初步建議生成 |
| 第 9~10 週 | 測試導引流程 + 實例測試 | 測試不同案例 |
| 第 11~12 週 | 介面美化與報告準備 | 撰寫簡報、報告 |

（5）活動圖（Activity Diagram）

使用者輸入案情 →語意分析 →若不明確 → 問細節 →若明確 → 指定案件類型 →檢索法條 → 生成摘要 →提供建議與下一步 →儲存紀錄

## 4.系統開發工具與使用環境

4-1 使用的開發工具

| 類別 | 工具/技術名稱 | 用途說明 |
| --- | --- | --- |
| 編程語言 | Python 3.10 | 建立後端邏輯、NLP 模型處理、法條分析與資料擷取 |
| 網頁前端 | HTML/CSS/JavaScript + React.js 或 Vue.js | 建構使用者介面，支援對話互動與動態回應 |
| 網頁後端 | Flask（Python Web Framework） | 負責處理 API 請求、與 NLP 模型溝通、資料回傳 |
| 資料庫 | MongoDB 或 SQLite3 | 儲存使用者輸入紀錄、法條資料與分類標籤 |
| NLP 模型庫 | Hugging Face Transformers (BERT, Legal-BERT) | 處理句子語意轉換與語意相似度計算 |
| 語言處理工具 | jieba（中文斷詞） | 預處理中文輸入，提升語意準確性 |
| API 開發 | Postman、Swagger（可選） | 用於測試與文件化 API 結構 |
| 版本控制 | Git + GitHub | 團隊協作、版本追蹤與備份 |
| 視覺化設計 | Figma 或 Draw.io（設計流程與 UI） | 畫面規劃、流程圖與操作圖設計 |

4-2 系統運行環境需求

基本硬體需求（開發與執行環境）

| 項目 | 最低需求 | 建議配置 |
| --- | --- | --- |
| CPU | Intel i5 第六代以上 | Intel i7 或 AMD Ryzen |
| 記憶體 | 8 GB RAM | 16 GB 以上（執行 NLP 模型效能更佳） |
| 硬碟空間 | 10 GB 以上 | SSD 儲存（提升模型與資料查詢速度） |
| 顯示卡（選用） | 無需 | 若本地部署 BERT 模型可使用 GPU（如 NVIDIA RTX） |

軟體與作業系統環境

| 項目 | 說明 |
| --- | --- |
| 作業系統 | Windows 10 / 11、Ubuntu 20.04、MacOS 也可（視團隊習慣） |
| Python 環境 | 建議使用 virtualenv 或 Conda 建立虛擬環境管理依賴套件 |
| 套件需求 | transformers, flask, pymongo, sqlite3, jieba, scikit-learn, numpy, pandas 等 |
| 瀏覽器支援 | 支援 Chrome、Edge、Firefox 等主流瀏覽器 |

可選雲端部署平台

| 平台 | 功能 | 說明 |
| --- | --- | --- |
| Render / Heroku | 快速部署 Flask 後端與前端頁面 | 免費方案足以展示 |
| Hugging Face Spaces | 可部署 NLP 模型展示 | 若用 Gradio 可快速完成互動展示 |
| Replit / Vercel | 用於簡單前端介面開發與預覽 | 適合製作小型 Demo |

小結:

本專題採用開源技術與輕量級框架設計，兼顧開發彈性與系統效能，同時保留日後進一步部署與擴充的彈性。透過 NLP 模型的整合與對話式界面設計，建立一套易於實作、可展示、具實用性的智慧法律導引系統。