國　立　屏　東　大　學

資 訊 管 理 學 系

資訊學院專題實務成果報告書

113學年度專題報告書

反詐圖文掃描機

指導教授： 林家樑

學生姓名： 李睿凱

林逸倫

王紹丞

林幸縈

中　華　民　國　113　年　12　月

目錄

圖目錄

壹、緒論

一、摘要

隨著網路詐騙手法日益多樣化，識別和預防詐騙行為變得愈加重要。本研究旨在開發一款針對詐騙訊息的預測應用，該應用能在 iOS（使用 Swift）、Android（使用 Kotlin）和 Web 平台（使用 Flask）上運行，提供用戶全方位的防詐騙服務。系統的主要功能包括：利用光學字符識別（OCR）技術進行圖片中的文字辨識，將識別出的文本進行詐騙性質的分類，以及提供查詢功能以驗證 Line ID 和 URL 是否為詐騙資訊。

在模型的構建上，我們選擇了 BERT（Bidirectional Encoder Representations from Transformers）作為文本分類的核心算法，並對其進行微調以適應我們的數據集。訓練過程中，我們對數據進行了詳細的預處理，使用 CKIP Transformers 進行斷句和斷詞，以提高模型對中文文本的理解能力。

在實驗中，我們進行了多輪交叉驗證，最終模型達到了平均準確度 99.04%。這顯示出模型在識別潛在詐騙訊息方面具有極高的效能。此外，系統的設計也考慮了用戶體驗，確保界面友好且操作簡便，使用戶能快速獲得所需資訊。

本研究的成果不僅展示了 BERT 模型在詐騙識別中的有效性，也為未來在詐騙防範方面的應用提供了重要的技術基礎。希望本系統能為用戶提供更安全的網路環境，減少詐騙行為對社會的影響。

關鍵詞：OCR、BERT、自然語言處理、訓練數據、文本識別、詐騙查詢

二、研究動機

在當前社會中，詐騙已成為一個嚴重的社會問題，對個人和經濟造成了巨大的損失。根據台灣刑事警察局的統計，2023年詐騙案件數量超過35,000件，詐騙金額達到79億元，創下歷史新高。從2020年開始，詐騙案件和金額持續上升，並在2023年達到頂峰。詐騙手法層出不窮，從電話詐騙到網路和社交媒體平台（如 LINE）進行的詐騙行為尤為猖獗，尤其是投資詐騙、網購詐騙和解除分期付款詐騙等手法，對廣大民眾造成了心理和經濟上的雙重壓力。

傳統的防範手段，如法律措施和事後補救，面對詐騙集團不斷演變的手法顯得無力。詐騙者常利用偽造的 LINE ID 或釣魚網站 URL 誤導用戶，而手動檢查和黑名單系統無法即時、準確地應對新型詐騙。這樣的現狀使得許多潛在的受害者在未察覺的情況下陷入詐騙陷阱，進一步凸顯出社會對詐騙防範工具的迫切需求。

隨著人工智慧（AI）和自然語言處理（NLP）技術的發展，特別是語言模型的應用，我們可以更加有效地分析與識別詐騙信息。OCR 技術的進步也為圖片中文字的自動識別提供了更高效的解決方案，這使得我們能夠從各種來源快速提取信息，並進行深入分析。因此，開發一個能即時且準確識別詐騙行為的智能工具，不僅是技術進步的體現，更是對社會責任的履行。透過這項研究，我們希望能為用戶提供一個有效的防範機制，減少詐騙事件的發生，並提升公眾的防詐騙意識。

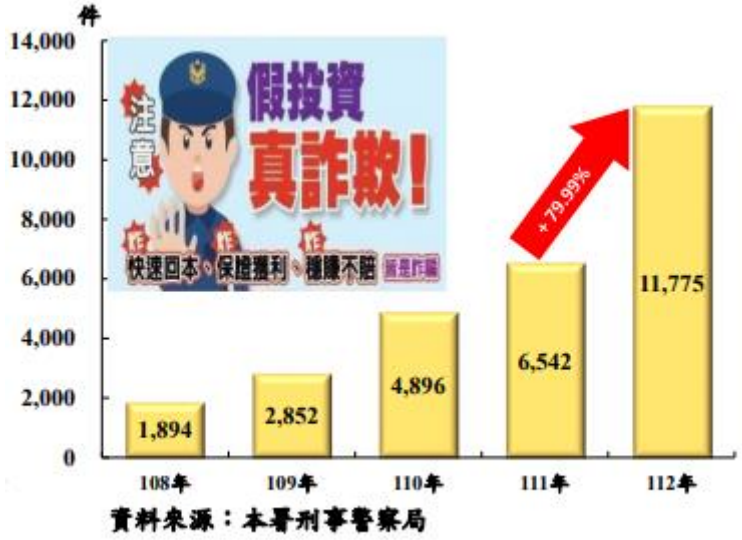

圖 1：詐騙統計圖

三、研究目的

本研究的目的是開發一款智能詐騙預測應用程式，結合 OCR 技術和 BERT 模型，提供三大核心功能：

（一）透過 OCR 辨識圖片中的文字，並用語言模型即時分類是否涉及詐騙

詐騙訊息不僅局限於網頁和訊息，許多詐騙集團會使用圖片來傳達詐騙內容，如偽造的發票、身份證明、或帶有詐騙性質的廣告圖像。因此，本系統的首個核心功能是通過光學字符識別技術（OCR，Optical Character Recognition）來自動化解析圖片中的文字內容，將圖片中的文字數據轉換為可分析的文字信息。這一過程使得用戶無需手動輸入圖片中的內容，提供了更高的效率與準確度。

在完成文字辨識後，系統會使用預先訓練的語言模型進行即時的詐騙分類。語言模型是一類基於自然語言處理（Natural Language Processing, NLP）的工具，它們透過學習大量的文本數據，來理解語言的語法和語意結構。語言模型能夠根據句子的上下文來分析其語意，並進行推理判斷。在詐騙偵測的場景中，語言模型可以根據已知的詐騙模式來判斷新文本中是否存在類似的詐騙語言特徵。這種基於深度學習的模型擅長處理包含潛在欺詐信息的句子，並能夠即時做出分類決策。語言模型的強大之處在於它們能夠捕捉到一些隱藏的語意信息，甚至是微妙的語言變化，這些往往是詐騙行為的關鍵線索。

透過這個功能，使用者只需上傳含有可疑信息的圖片，系統即可自動檢測該圖片中的文字並判斷是否可能涉及詐騙。這不僅大幅減少了手動檢查的工作量，還能提供即時且高準確率的詐騙預警。語言模型的引入讓詐騙偵測更具智慧性，能適應不斷變化的詐騙模式和手法，從而提供更可靠的保護。

（二）提供 LINE ID查詢功能，判斷該 ID 是否曾被報告為詐騙帳號

LINE 作為一個廣泛使用的即時通訊應用，常常成為詐騙者進行欺詐活動的平台。他們利用偽造的 LINE ID 冒充可信任的身份，誘騙用戶進行交易或提供個人信息。為了解決這一問題，我們的應用程式提供了一個 LINE ID 查詢功能，讓使用者能夠查詢某個 LINE ID 是否曾被報告為詐騙帳號。當用戶輸入一個 LINE ID 時，系統會即時比對該 ID 是否存在於資料庫中。如果該 ID 曾經被報告為詐騙帳號，系統將發出警告，提醒用戶避免與此 ID 進行進一步的互動。

（三）提供 URL 查詢功能，檢查該網址是否被標記為詐騙網站

釣魚網站（Phishing Websites）是詐騙行為中的常見手法，詐騙者常會創建與合法網站極其相似的假網站，誘導用戶輸入個人敏感信息或進行不當的金錢交易。為應對這種威脅，應用程式設計了 URL 查詢功能，讓使用者能夠快速檢查一個網址是否曾被標記為詐騙網站。此功能依賴於我們收集的已知詐騙網站數據庫，該數據庫包括來自各個信任機構和社群回報的詐騙 URL 清單。當使用者輸入一個 URL 時，系統會即時比對該網址是否存在於詐騙數據庫中。如果該網址被列為詐騙網站，系統將發出警示，提醒用戶避免訪問該網站，從而保護用戶的數據安全。

（四）跨平台實現

系統支援 iOS、Android 和 Web 三個平台，確保使用者無論是在行動裝置還是電腦上，都能便捷地使用系統功能。跨平台的技術實現提供了廣泛的使用場景，滿足了現代使用者隨時隨地查詢與辨識詐騙的需求。

貳、文獻探討

文獻探討主要涵蓋以下幾個部分：詐騙偵測技術的現狀、OCR 與 NLP 技術在詐騙檢測中的應用、LINE ID及URL查詢的研究現況、以及跨平台應用的趨勢與挑戰。

一、詐騙偵測技術的現狀

近年來，隨著數位科技的發展，詐騙行為的手段越發多樣化且複雜。傳統的詐騙防範方法，如黑名單機制和規則檢測，面對不斷演變的詐騙手法顯得力不從心。例如，黑名單系統依賴於已知的詐騙數據，難以應對新型態的詐騙行為。此外，這些系統通常無法即時更新，導致誤報與漏報的情況頻繁發生。

隨著人工智慧（AI）技術和自然語言處理（NLP）的快速發展，語言模型在詐騙偵測中展現了巨大的潛力。這些模型能夠深入分析文本中的語言特徵與模式，並且能夠辨識出潛在的詐騙線索。例如，詐騙者經常使用一些特定的語言或結構來誘導受害者，而傳統的防範方法無法充分捕捉這些細微的語言變化。透過應用語言模型，我們能更精確地捕捉詐騙語言中的模式，並有效降低誤報與漏報的情況。

此外，語言模型能夠不斷學習和適應新的詐騙手法，這使得它們在面對多變的詐騙行為時，仍能保持高效的偵測能力。隨著語言模型的不斷優化，它們將在詐騙偵測的精準度和即時性方面發揮更大的作用，為使用者提供更加智能化的詐騙預防解決方案。

二、OCR 與 NLP 技術在詐騙檢測中的應用

OCR 技術： 光學字符識別（Optical Character Recognition, OCR）是一種能夠將圖片中的文字轉換為可編輯數據的技術，廣泛應用於票據識別、文件數位化等領域。在詐騙檢測方面，OCR 技術可用於將偽造文件、詐騙訊息等內容數位化，為後續的語義分析提供基礎數據。然而，OCR 技術面臨的挑戰包括文字辨識的準確度問題，特別是在面對手寫文字和低質量圖片時。

BERT 模型與 NLP 技術： BERT（Bidirectional Encoder Representations from Transformers）是由 Google 開發的深度學習模型，能夠有效捕捉語句上下文關係，被廣泛應用於文本分類、情感分析等 NLP 任務中。在詐騙偵測方面，BERT 模型可以透過學習大量詐騙案例數據，精準識別出詐騙語句中的關鍵字和語義特徵。研究顯示，基於 BERT 的詐騙分類模型在準確性和檢測速度上均優於傳統的詞袋模型。

三、LINE ID 及 URL 查詢的研究現況

LINE ID 和 URL 查詢： LINE ID 查詢功能主要基於政府公開資料庫，透過匯入詐騙帳號的名單和已知的詐騙網址來進行判斷。政府資料的公開性和透明度提供了可靠的參考依據，能有效輔助用戶在日常通訊中避免接觸到可疑的聯絡人與保護用戶免於釣魚網站的侵害。

參、研究架構

一、研究流程

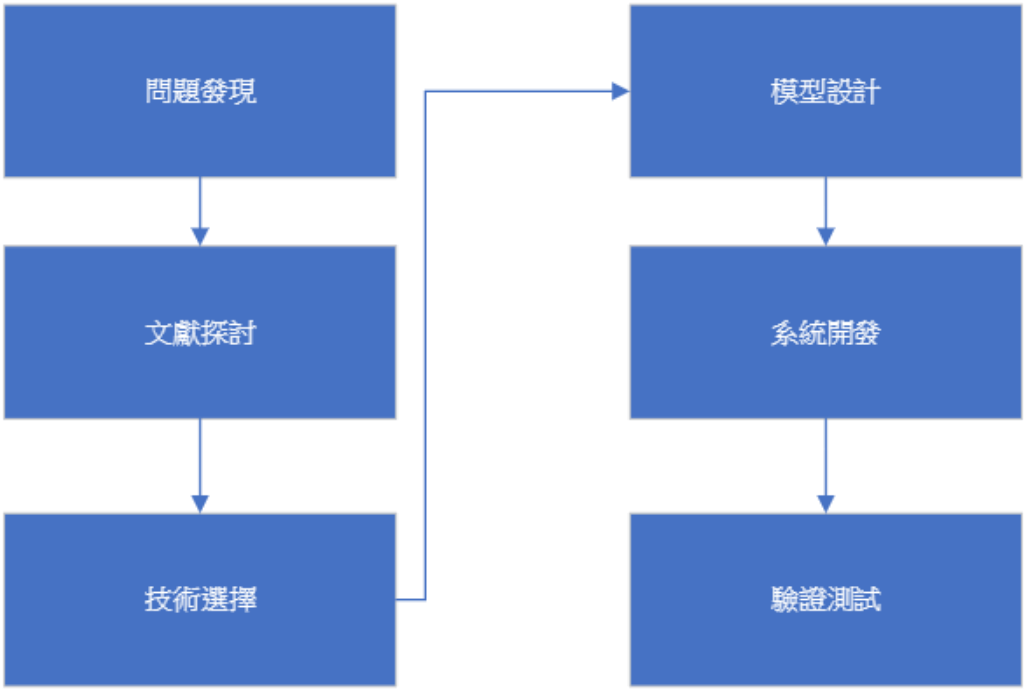

圖 2：研究流程圖

二、研究方法

（一）OCR 影像文字辨識

- iOS（Vision Framework）

  在 iOS 平台上，選用了 Apple 的 Vision Framework 來進行圖片中的文字辨識。Vision Framework 是一個高度優化的影像處理框架，具備強大的 OCR 能力，並能夠與 iOS 原生系統無縫集成。

- Android（Google ML Kit）

在 Android 平台上，選用了 Google 的 ML Kit。ML Kit 是一個強大的移動端機器學習工具包，提供多種 API，其中包括高效的 OCR 功能，能夠直接從圖片中提取文字。

- Web（Tesseract OCR）

  在 Web 平台上，選用了開源的 Tesseract OCR 工具。Tesseract 是一款經過多年發展的強大 OCR 引擎，適用於各種平台，特別是網頁端的應用。

（二）詐騙文本分類模型訓練

我們選用 BERT（Bidirectional Encoder Representations from Transformers）模型進行詐騙文本的分類微調。BERT 是一種基於 Transformer 的深度學習模型，能夠同時考慮上下文語義，對文本進行精確的語意理解與分類。

1. 數據收集

資料主要從公開的詐騙報告、社交平台上的用戶舉報資料以及網絡上公開的詐騙數據庫中收集文本數據，這些數據涵蓋常見的詐騙信息、廣告以及正常的非詐騙文本。

2. 數據預處理

在數據清理前，我們使用 CKIP Transformers 進行中文文本的段句與斷詞處理。這步驟能夠有效地將中文文本切分成句子和詞彙，便於後續的文本分析與處理。CKIP 的預訓練模型在中文自然語言處理方面表現出色，能夠提升斷詞的準確性。

清理：清理過程包括去除標點符號、去除空白符號以及移除不必要的雜訊。

3. 訓練模型

先把一開始處理完的文本轉換為 BERT 可處理的 tokens、注意力 masks 等格式。接下來將數據集分為多組訓練集和驗證集，並針對每組數據進行交叉驗證訓練。每次訓練時，使用預訓練的 BERT 模型進行微調，這過程中會根據詐騙和非詐騙訊息的數據進行學習。訓練使用交叉熵損失函數來計算誤差，並使用 Adam 優化器來更新模型參數。

訓練過程中，我們會在每一個 epoch 結束後，對模型在驗證集上的表現進行評估，並通過混淆矩陣來分析分類效果。在多次訓練後，選擇表現最佳的模型，並將其保存下來與 tokenizer 一同使用於後續的詐騙訊息分類系統中。

三、系統架構

本系統架構分為前端和後端兩部分，前端包括 iOS、Android 和 Web 平台，後端則由 Python 的 http.server 和 Flask 提供 API 支援。三個平台各自運行，通過 HTTP 請求與後端進行通信。

- **前端設計**
  - **iOS 前端**：使用 Swift 開發，實現圖片上傳、文字辨識功能，並通過 HTTP 請求與 iOS 平台上的 Python http.server 進行通信。
  - **Android 前端**：使用 Kotlin 開發，與 iOS 功能類似，主要負責圖片上傳與文字辨識，並通過 HTTP 請求與 Android 端的 Python http.server 進行數據交換。
  - **Web 前端**：使用 HTML、CSS、JavaScript，並由 Flask 框架支援後端。Web 平台負責提供用戶界面，用於圖片上傳、文字識別和查詢詐騙信息。
- **後端設計**
  - **iOS 和 Android 的後端**：使用 Python 的內建 http.server 架設簡單的 HTTP 伺服器，接收來自前端的 HTTP 請求，進行圖片文字的分類或查詢詐騙資料。
  - **Web 的後端**：使用 Flask 框架提供的 Server 處理圖片文字識別的分類結果，並進行 Line ID 和 URL 查詢。
  - **資料庫設計**：後端資料儲存在 CSV 文件中，這些 CSV 檔案用於儲存已知的詐騙 Line ID 和 URL。後端系統會讀取 CSV 文件進行數據查詢，然後將結果返回前端。

四、系統環境

（一）系統開發工具

1. 開發語言

- 前端：iOS (Swift)， Android (Kotlin)， Web (JavaScript, HTML, CSS)
- 後端：Flask (Python)， http.server(Python)

2. 開發環境

Xcode， Android Studio， VSCode

（二）系統執行環境

1. 軟體

Android端版本7.0以上，並針對Android 13進行最佳化

iOS端需版本16.6以上

2. 硬體

系統對硬體的需求較低，大多數移動裝置和桌面電腦皆能流暢運行。

肆、專題成果

一、BERT 模型訓練與評估

透過對大量訓練數據的處理與訓練，我們獲得了以下成果：

1. **模型訓練**：
   - ○ 訓練過程共進行了 5 個 epoch，最終的損失值（Loss）從初始的 0.1516 降至 0.0089，顯示出模型逐漸改善的學習能力。
   - ○ 每一輪訓練結束後，我們使用交叉驗證來評估模型的性能，並記錄了每次訓練與驗證的準確度。
2. **交叉驗證結果**：
   - ○ 在 10 次的交叉驗證中，模型的準確度均高於 98%，最高達到 100%。最終計算出的平均準確度為 0.9904，顯示模型在詐騙訊息分類上的優異表現。
3. **訓練時間**：
   - ○ 整體訓練時間為 845.22 秒，這表明模型訓練過程相對穩定且高效。
4. **模型評估**：
   - ○ 我們在每次驗證後使用混淆矩陣分析模型的分類效果，發現模型對詐騙和非詐騙訊息的識別能力強，幫助我們更好地理解其性能。

```
Epoch 1/5, Loss: 0.15161800497469116
Epoch 2/5, Loss: 0.046525115094392014
Epoch 3/5, Loss: 0.02907952981016843
Epoch 4/5, Loss: 0.013028701163782137
Epoch 5/5, Loss: 0.008885813907122385
Training time: 845.22 seconds
100%|██████████| 10/10 [2:12:19<00:00, 793.97s/it]
0.9821428571428571
  train_period val_period       acc
0      第 1 次訓練    第 1 次驗證  0.992126
1      第 2 次訓練    第 2 次驗證  0.992908
2      第 3 次訓練    第 3 次驗證  1.000000
3      第 4 次訓練    第 4 次驗證  0.984496
4      第 5 次訓練    第 5 次驗證  1.000000
5      第 6 次訓練    第 6 次驗證  0.985294
6      第 7 次訓練    第 7 次驗證  1.000000
7      第 8 次訓練    第 8 次驗證  0.982456
8      第 9 次訓練    第 9 次驗證  0.984733
9     第 10 次訓練   第 10 次驗證  0.982143
平均準確度 0.990415584926969
```

圖 3：BERT訓練結果

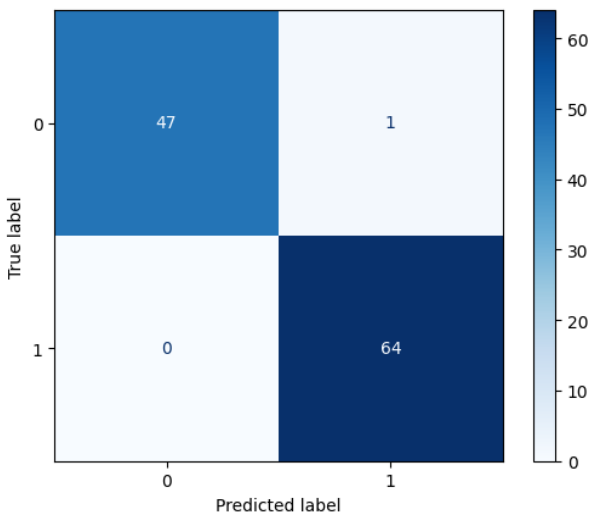

圖 4：混淆矩陣

二、系統功能

（一）iOS端

1. 圖片分享辨識

此功能是在程式介面外，可以快速辨識圖片是否為詐騙，先選擇想要辨識的圖片，選擇分享對象選擇"Anti-fraud"，之後彈出視窗會顯示預測結果。

圖 5：iOS圖片分享1

圖 6：iOS圖片分享2

2. 拍照掃描辨識

在"拍照掃描"除了可以自己手動拍照外，也可以讓相機自己掃描。拍完照後還可以讓使用者手動拖曳選取文字範圍的框，點擊"儲存"後並點選"擷取的文字"後可以看到相機擷取到的文字顯示於此，且這裡還可讓使用者編輯文字，按下"發送"後就會顯示預測結果。

圖 7：iOS拍照掃描1

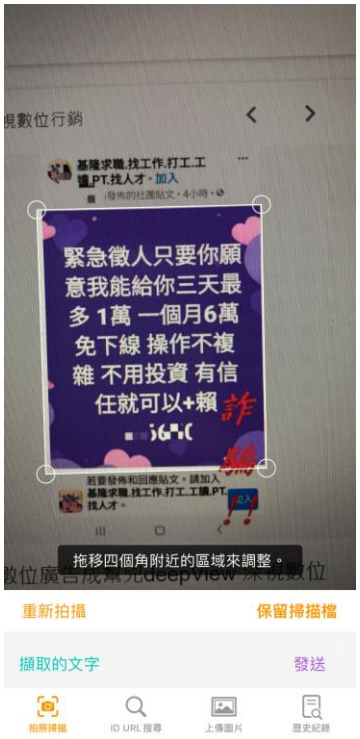

圖 8：iOS拍照掃描2

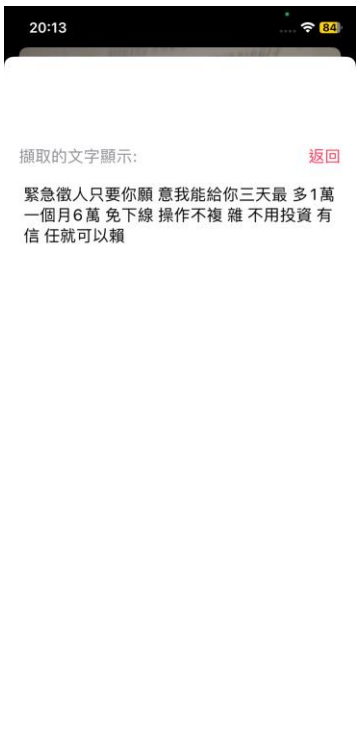

圖 9：iOS拍照掃描3

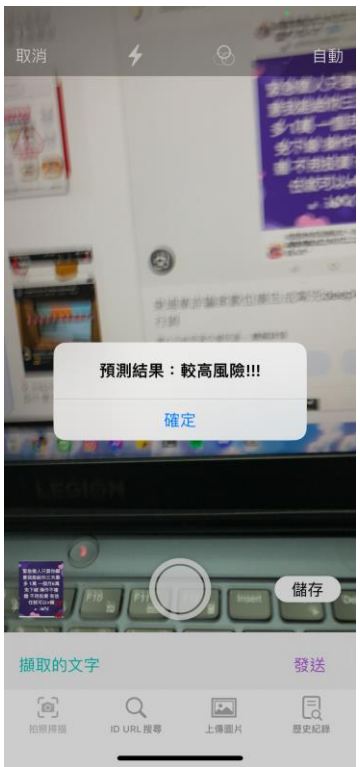

圖 10：iOS拍照掃描4

3. LINE ID/URL 搜尋

切換按鈕可以切換成LINE ID/URL 模式，當接收到詐騙 ID 或 URL 時除了會彈出訊息框及紅色的文字外，也會用醒目的動畫提醒使用者。

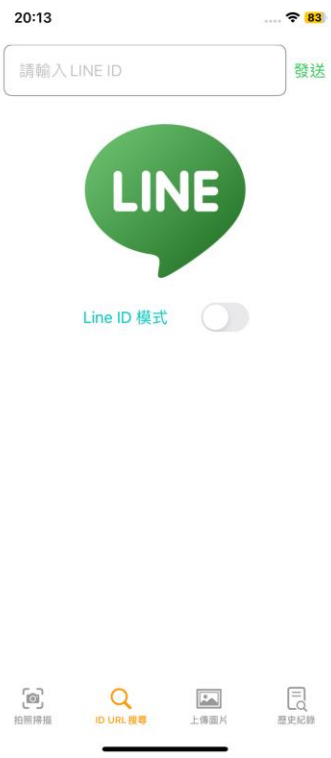

圖 11：iOS LINE ID/URL 搜尋1

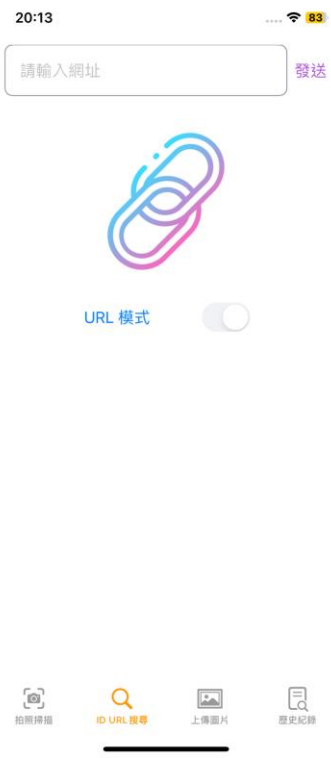

圖 12：iOS LINE ID/URL 搜尋2

圖 13：iOS LINE ID/URL 搜尋3

圖 14：iOS LINE ID/URL 搜尋4

4. 圖片上傳辨識

只要上傳圖片就會自動執行文字掃描，底部就會顯示一個可供編輯的文字框，按下發送後就會顯示預測結果。

圖 15：iOS 上傳辨識1

圖 16：iOS 上傳辨識2

圖 17：iOS 上傳辨識3

5. 歷史紀錄

可以讓使用者看到先前掃描的時間、文字及結果。

圖 18：iOS 歷史紀錄

（二）Android端

1. 圖片辨識

在此功能下，使用者可以選擇拍照上傳圖片或是相簿上傳圖片，上傳後按下辨識文字可以掃描圖片中所有的文字，文字會顯示在文字辨識結果欄並且文字可以編輯，按下字串辨識後就會顯示預測結果。按下查詢紀錄可以查以前辨識的時間和結果。

圖 19：Android 上傳辨識1

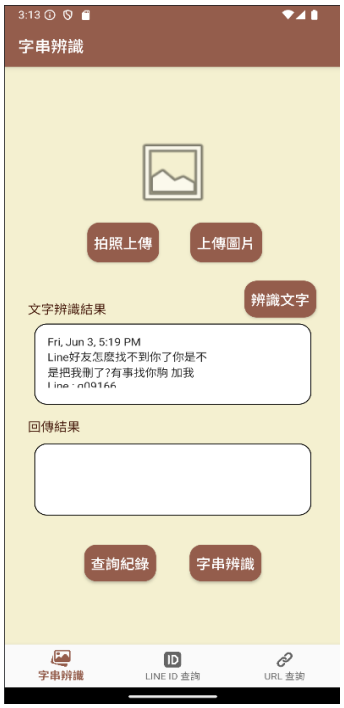

圖 20：Android 上傳辨識2

圖 21：Android 上傳辨識3

圖 22：Android 上傳辨識4

圖 23：Android 上傳辨識查詢紀錄

2. LINE ID 與 URL 搜尋

按下查詢即可搜尋，回傳結果會顯示查詢結果。按下查詢紀錄可以查以前辨識的時間和結果。

圖 24：LINE ID/URL 搜尋1

圖 25：LINE ID/URL 搜尋2

圖 26：LINE ID/URL 搜尋3

圖 27：LINE ID/URL 搜尋4

圖 28：LINE ID/URL 搜尋5

圖 29：LINE ID/URL 搜尋6

（三）Web端

1. 圖片上傳辨識

只要上傳圖片後就會出現可以拖曳紅色框來選取需要辨識的文字，底部就會顯示一個可供編輯的文字框，按下發送後就會顯示預測結果。

圖 30：Web 上傳辨識1

圖 31：Web 上傳辨識2

圖 32：Web 上傳辨識3

圖 33：Web 上傳辨識4

2. LINE ID/URL 搜尋

切換按鈕可以切換成 LINE ID/URL 模式，輸入後按下搜索即可開始查詢，之後會顯示預測結果。

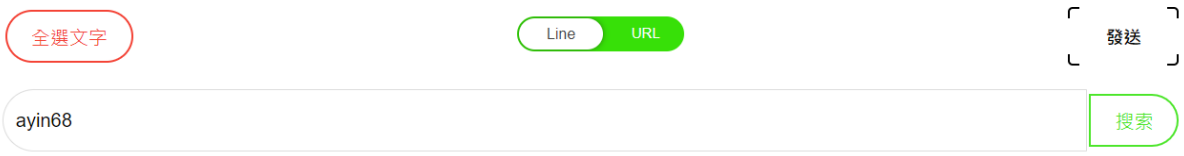

圖 34：Web ID/URL 搜尋1

圖 35：Web ID/URL 搜尋2

（四）Server端

Server 分別寫了三個 URL 路徑，分別為"str_request"、"lineid_request"及"url_request"，個別對應文字預測、LINE ID 搜尋和 URL 的 Function 搜尋。

```python
# 定義請求處理類
class RequestHandler(BaseHTTPRequestHandler):
    def do_POST(self):
        content_length = int(self.headers["Content-Length"])
        post_data = self.rfile.read(content_length)
        post_data_str = post_data.decode("utf-8")

        if self.path == '/str_request':
            response_message = str_request(post_data_str)
        elif self.path == '/lineid_request':
            response_message = lineid_request(post_data_str)
        elif self.path == '/url_request':
            response_message = url_request(post_data_str)
        else:
            response_message = "無效的請求路徑"

        self.send_response(200)
        self.send_header("Content-type", "text/plain; charset=utf-8")
        self.end_headers()
        self.wfile.write(response_message.encode("utf-8"))

server = HTTPServer((HOST, PORT), RequestHandler) # 創建 HTTP 服務器並指定請求處理類

print("server 運行中...")
server.serve_forever()
```

圖 36：HTTP Server

```python
# 處理文字預測
def str_request(post_data_str):
    try:
        response_message = predict(post_data_str, model, tokenizer, ws_driver, DEVICE)
    except Exception as e:
        print(f"錯誤代碼: {str(e)}")
        response_message = "失敗"
    return response_message

# 處理 line id 搜尋
def lineid_request(line_id_str):
    try:
        response_message = select_id(line_id_str)
    except Exception as e:
        print(f"錯誤代碼: {str(e)}")
        response_message = "失敗"
    return response_message

# 處理 url 搜尋
def url_request(url_str):
    try:
        response_message = select_url(url_str)
    except Exception as e:
        print(f"錯誤代碼: {str(e)}")
        response_message = "失敗"
    return response_message
```

圖 37：HTTP Server Fuction

三、工作分配

工作分配表	
姓名	**職責**
李睿凱	程式開發、文件製作
林逸倫	程式開發及美化、文件製作
林幸縈	簡報及文件製作、程式美化
王紹丞	簡報製作、會議記錄統整

圖 38：工作分配表

四、工作進度表

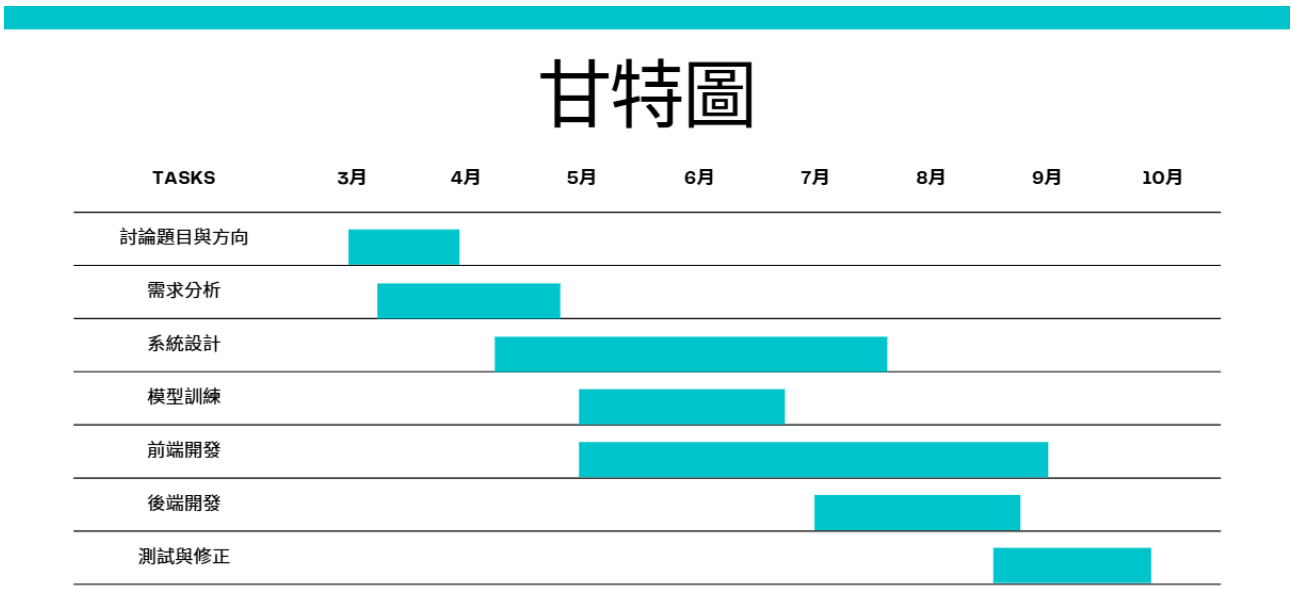

圖 39：甘特圖

伍、總結

一、結論

本專案成功開發了一個多平台的詐騙預測系統，涵蓋 iOS、Android 和 Web 三個主要平台，並整合了多種技術，包括 OCR 文字識別、深度學習文本分類、以及詐騙 Line ID 和 URL 查詢等功能。通過結合各平台特定的 OCR 技術（iOS 使用 Vision Framework、Android 使用 Google ML Kit、Web 使用 Tesseract OCR），系統能夠快速識別圖片中的文字，並使用 BERT 模型進行詐騙判斷。

在詐騙資料查詢方面，系統利用簡單且高效的 CSV 文件作為資料庫，允許使用者查詢已知的詐騙 Line ID 和 URL。此種架構不僅具備輕量級的優勢，也讓開發與部署變得更加靈活。此外，系統的後端通過 Python 的內建 http.server 與 Flask 框架進行實現，為每個平台提供簡單且快速的 API 支援。

專案成果有以下幾點

1. 成功運用 OCR 技術將圖片中的文字提取出來，實現跨平台的詐騙文本預測功能。
2. 使用預訓練的 BERT 模型進行文本分類，能夠有效區分出潛在的詐騙訊息。
3. 實現了對詐騙 Line ID 和 URL 的查詢，幫助使用者識別過去被標記為詐騙的資料。
4. 系統具有簡易擴展性，基於 Python 的輕量伺服器和 CSV 資料庫，具備靈活部署的優勢。

在本專案中，我們不僅學習了多平台開發的技術，也掌握了自然語言處理與深度學習模型應用的實踐經驗，展示了利用 AI 技術防範網路詐騙的可行性。

二、延伸應用

本專題可在以下幾個方面進行延伸與應用：

1. 資料庫擴展：

   現階段的資料庫主要依賴於 CSV 文件，未來可以考慮使用 MySQL、PostgreSQL 或 MongoDB 等更加成熟的資料庫系統，提升查詢性能並支持大規模數據存儲。

2. 功能拓展：

   可以增加更多的查詢功能，例如針對詐騙電話號碼或電郵地址的查詢，進一步提升系統的實用性。

3. 即時詐騙偵測：

   系統可整合即時數據分析功能，通過監控社交媒體上的訊息流，主動識別並提示潛在的詐騙風險。

4. 應用領域擴展：

   除了偵測詐騙訊息，系統也可應用於其他領域，如偽造新聞的檢測，識別虛假信息等，幫助社會進一步防範網絡風險。

5. 模型優化與擴展：

   隨著深度學習技術的進步，未來可以引入更強大的模型或多語言支持，擴展系統對不同語言的詐騙分類能力，甚至結合更多的 NLP 模型來提升預測準確度。

陸、參考資料

1. Lee, M. (2019). 進擊的 BERT：NLP 界的巨人之力與遷移學習.
   https://leemeng.tw/attack_on_bert_transfer_learning_in_nlp.html
2. 李宏毅. (2019). ELMO, BERT, GPT［影片］.
   https://www.youtube.com/watch?v=UYPa347-DdE
3. HuggingFace. (n.d.). BERT-Base, Chinese.
   https://huggingface.co/google-bert/bert-base-chinese
4. BaseHTTPServer (Internet) - Python 中文开发手册 - 腾讯云取自
   https://cloud.tencent.com/developer/section/1368135
5. Uiverse | The Largest Library of Open-Source UI elements
   https://uiverse.io/
6. SwiftUI 初體驗： 建構一個簡單 App 讓你了解 SwiftUI 有多強大！ (appcoda.com.tw)
   https://www.appcoda.com.tw/swiftui-introduction/
7. iOS Share Extension开发 - 简书 (jianshu.com)
   https://www.jianshu.com/p/69f0beacfe17
8. GitHub - tesseract-ocr/tesseract: Tesseract Open Source OCR Engine (main repository)
   https://github.com/tesseract-ocr/tesseract
9. [實用心得] Tesseract-OCR. 因為工作上的關係，接觸到了 Tesseract 由 Google… | by 凱稱研究室 | Medium
   https://kaichenlab.medium.com/%E5%AF%A6%E7%94%A8%E5%BF%83%E5%BE%97-tesseract-ocr-eef4fcd425f0
10. 【Python Flask 入門指南】輕量級網頁框架教學 | 5 行程式碼 x 架設網站 - iT 邦幫忙::一起幫忙解決難題，拯救 IT 人的一天（ithome.com.tw）
    https://ithelp.ithome.com.tw/articles/10258223
11. 警政統計通報-內政部警政署全球資訊網 (npa.gov.tw)
    https://www.npa.gov.tw/ch/app/data/list?module=wg057&id=2218
12. Android Kotlin 基本概念課程 | Android Developers
    https://developer.android.com/courses/android-basics-kotlin/course?hl=zh-tw