2024 總統盃黑客松

比賽企劃書

金融詐欺守門員：生成仿真資料集深度學習建模之即時來電詐騙偵測 APP

組長	賴冠儒
組員	許兆豐
組員	邵以懷
組員	李肇恆
組員	蔡安德
組員	簡子策
組員	李淳皓
組員	蘇胤翔
組員	Apoorv Saxena
指導教授	AI 中心 陳昭伶

目錄

壹、研究背景與動機

生成式人工智慧技術的發展，使得一人變成有眾多分身，導致全球商業與金融詐欺案件暴增，例如：生成式人工智慧撰寫大量詐騙電子郵件；根據國際反詐騙聯盟 2023 年數據統計，詐騙金額前三名的國家分別為新加坡、瑞士及奧地利(台灣排名第 23 名)；GDP 損失前三名的國家分別為肯亞、越南及巴西(台灣排名第 14 名)。生成式人工智慧結合深度學習方法導入金融產業的應用，於金融詐欺檢測方面，針對具備隱匿性高及犯罪手法多樣的電話詐騙進行分析，其演算法可快速分析來電詐騙風險及異常情況，增強詐欺預防以確保社會個人財務安全，可實際應用於真實生活場域。

貳、研究問題

目前社會詐騙案件層出不窮存在以下問題：

- 執法機關蒐證困難：由於電話詐騙具備隱匿性，執法機關往往難以取得蒐證，導致追緝困難。
- 民眾報案意願不高：若詐騙集團沒有成功詐騙，則民眾的報案意願不高，因此許多潛在的詐騙行為無法取得。

在本研究中，透過網路爬蟲由網路真實詐騙文本資料，及自然語言處理取得詐騙關鍵字建立詐騙真實資料集，並由此真實資料集生成仿真資料集；採用深度學習方法進行建模與分析，提供準確的詐騙分析；並開發用於使用者端手機應用程式，用於幫助一般民眾預防日益猖獗的電話詐騙，透過通話內容語音轉文字，及連結後端詐騙關鍵字判斷，即時偵測電話詐騙行為。本研究可有效防範電話詐騙犯罪，保護個人財產安全，並可透過電話錄音保存詐騙犯案證據，對於打擊電話詐騙犯罪具有實質貢獻。

參、可行性及研究步驟

● 自然語言處理真實詐騙案件分析及生成仿真資料集

在本研究中，未來媒體實驗室金融詐欺守門員研究團隊之研究步驟，包含網路爬蟲，自然語言處理文本分析，詞頻統計與視覺化，建立關鍵字資料集，資料及缺失值處理及生成仿真資料集。首先透過爬蟲技術(Selenium 及 BeautifulSoup)爬取國內外論壇和社團上之真實詐騙資料(如圖一所示)，內容為民眾分享電話詐騙的經驗，爬蟲資料來源包含 Quora 論壇、臉書社團、Scam Phone Calls 及 YT 頻道 Kitboga；接著採用自然語言處理的(Natural Language Tool Kit, NLTK)模組對所有爬蟲後取得的資料進行文本分析，由於名詞最具有顯著性，採用詞性標註方式取出文本的名詞關鍵字以建立詞袋，由文本中擷取出關鍵字(如圖二所示)；再透過 Counter 函數進行詞頻統計，統計前 50 個出現次數最多之關鍵字(如圖三所示)，以長條圖(如圖四所示)和文字雲視覺化呈現(如圖五所示)，方便人工判斷新增停用詞。

最後，由關鍵字建立資料集，詞頻統計結果依高詞頻關鍵字主要包含五類：金錢相關(Money related)，個資相關(Personal information related)，誘導性詞彙(Inducement words related)，推銷字眼(Promotional content related)及親屬關係(Kinship related)。其中金錢相關類別於金融詐欺檢測最為重要，其權重高於其他類別，且此類別大部分關鍵字詞頻皆高於其他類別，因此將此類別關鍵字個別取出建立金錢相關類別一(Money related 1)至金錢相關類別九(Money related 9)，資料表最後欄位標籤值皆預設為 1(如圖七所示)。接著由關鍵字頻率進行金錢相關類別一(Money related 1)至金錢相關類別五(Money related 9)的文本搜尋，依詞頻高低依序由關鍵字(money, card, credit, bank, insurance, taxes, bill, dollars, warrant)查找文本，當遍歷每個關鍵字時，使用 str.contains 方法選擇取得包含該關鍵字的文本資料值，並取得其對應的 ID 值，將文本資料值及 ID 值分別儲存於資料表對應的金錢相關類別欄位及 ID 欄位。依此步驟由金錢相關類別一(Money related 1)至金錢相關類別九(Money related 9)依序取得資料值填入資料表中，若取得關鍵字文本資料值對應的 ID 欄位已存在於資料表，表示資料已存在則忽略，未找到關鍵字的資料欄則補 0；個資相關(Personal information related)，誘導性詞彙(Inducement words related)，推銷字眼(Promotional content related)及親屬關係(Kinship related)欄位，亦依詞頻高低依序查找填入關鍵字資料值至資料表中(如圖六所示)。依

此步驟可確保資料表依詞頻高低順序取得資料值，擷取與金融詐欺最相關的關鍵字，建立來自真實金融詐欺文本分析資料的真實資料集(如圖七所示)，後續步驟包含對真實資料集進行缺失值處理。

由於深度學習需要龐大的資料量進行建模，最後將真實資料集生成仿真資料集，採用 Python 的 CTGAN 模組進行資料生成，解決訓練資料量不足的問題，用於監督式學習的人工神經網路(Artificial neural network, ANN)建模。建模測試階段，除了蒐集來自網路的真實詐騙文本資料，亦將與電信業者及金融業者合作，取得真實金融詐騙資料，進一步增進訓練模型的準確度。由於合成資料集已確定無法用於判斷金融詐欺事件，本研究所建立之生成仿真資料集，內容源自詐騙文本所建立之真實資料集，可實際應用於真實金融場域。

```python
# define initial page height
last_height = driver.execute_script("return document.body.scrollHeight")

while True:
    # simulate scrolling down the page
    driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")

    # wait until the new content load
    time.sleep(2)

    # get the current page height
    new_height = driver.execute_script("return document.body.scrollHeight")

    # check if new content has been loaded
    if new_height == last_height:
        break

    # update page height for next iteration
    last_height = new_height

# scrap the page text
results = driver.find_elements(By.CLASS_NAME, "q-box.qu-cursor--pointer.QTextTruncated___StyledBox-sc-1pev100-0.gCXnis")

# create csv file to store results
with open('scam_records_1.csv', 'w', newline='', encoding='utf-8') as csvfile:
    fieldnames = ['Scam Record', 'Content']
    writer = csv.DictWriter(csvfile, fieldnames=fieldnames)

    writer.writeheader()
    for i, result in enumerate(results, start=1):
        # replace "Continue Reading" with space in each content
        result_text = re.sub(r"Continue\s*Reading", "", result.text, flags=re.IGNORECASE)
        result_text = result_text.strip()

        print(f"<Scam record #{i}>")
        print(result_text)
        print("-" * 30)

        # store the results
        writer.writerow({'Scam Record': f"{i}", 'Content': result_text})

driver.quit()
```

```
<Scam record #1>
My favorite is the “Grandma I’m in trouble call.” I have neither spouse nor children, so it is difficult to imagine how I would have acquired a grandchild.
“Grandma, How are you?”
“OK.”
“Do you know who this is?”
“Of course, I know my own grandson.”
“Grandma, how are you doing?”
“Not so good, honey. Actually, I am in real trouble. I was on the way home from church, and the police officer was really very nice when he stopped me. He just did not understand t
------------------------------
<Scam record #2>
I might be a little biased, but as an Asana employee, I believe Asana is the best project management tool. With Asana, my entire team has a central source of truth so they know exa
Every team is different, so your team might not use Asana exactly like my team does. But every team can benefit from more clarity, coordination, and collaboration at work. Here are
------------------------------
<Scam record #3>
I had the “I’m calling to fix your Windows” call. I played the dumb as a rock woman.
Me: “Finally! I’ve been waiting all day for you guys to show up!”
Scammer: “No, we don’t come to your house. I fix your Windows.”
Me: “Well of course you do, and it has a big crack in it. Your company said you would be here today, and I’ve been waiting all day.”
Scammer: “Lady, I’m fixing the Windows on your computer.”
Me: “Oh my goodness! You mean you can fix that too? Great! But I need the living room window fixed first, the rain is getting in, and we’re having a problem with birds flying in. O
------------------------------
<Scam record #4>
One time, I was hanging out at my grandmother’s house, and she got a phone call. This was a few years ago, when she only had two grandkids - me and my brother.
When my grandma looked at the caller ID, she didn’t recognize the number, so she winked at me, and answered the phone, putting it on speakerphone.
Scammer: Hey, Grandma, I just wanted to ask for a little bit of help. I’m stuck in New York. I went there, but I didn’t have enough money to get back.
Now, this voice was distinctly female, maybe in their twenties. I’m only fifteen, and I was ten or eleven, maybe twelve, when this call was takin
------------------------------
<Scam record #5>
A young woman, perhaps in her early twenties by the sound of her voice, called me pretending to be from Microsoft. She had an American accent and was very pleasant. I acted like a
“Sir, what kind of computer do you have? We detected suspicious activity and we need to resolve this now to protect you and our servers.”
```

圖一 真實詐騙文本 Selenium 爬蟲

```
import re

import nltk
nltk.download("stopwords")
nltk.download('averaged_perceptron_tagger')

from nltk.corpus import stopwords
from nltk import pos_tag

# create an empty list to store the modified texts
corpus = []
for i in range(0, len(texts)):
  review = re.sub("[^a-zA-Z]", " ", texts[i]) # replace all punctuations with space
  review = review.lower() # turn all words into lowercase ones
  review = review.split()
  tagged_review = pos_tag(review)

  all_stopwords = stopwords.words("english")
  # add redundancy words
  new_stopwords = ["call", "scammer", "phone", "time", "name", "year", "caller",
                   "scam", "day", "guy", "hello", "minutes", "windows", "accent", "someone", "voice",
                   "person", "people", "question", "thing", "car", "calls", "conversation", "scammers",
                   "years", "something", "work", "telephone", "man", "home", "house", "line", "rings",
                   "point", "friend", "office", "yes", "sir", "lot", "couple", "questions", "way", "months",
                   "bit", "fun", "things", "end", "today", "computer", "company", "life", "woman", "okay",
                   "days", "lady", "anything", "sorry", "morning", "times", "india", "one", "hi", "fraud",
                   "kind", "night", "department", "nothing", "job", "cell", "rang", "friends", "everything",
                   "hours", "month", "list", "weeks", "story", "answer", "part", "hour", "moment", "etc",
                   "team", "family", "week", "state", "number", "please", "numbers", "yeah", "right", "thank"]
  all_stopwords.extend(new_stopwords)

  review = [word for word, pos in tagged_review if pos.startswith("NN") and word not in set(all_stopwords)] # remove stopwords and keep meaningful noun roots
  review = " ".join(review) # turn each word lists into full strings(in one list)
  corpus.append(review)
print(corpus)
print(len(corpus))

[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     /root/nltk_data...
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
['favorite grandma trouble spouse children grandchild grandma course grandson grandma honey trouble church police officer grandpa marijuana possession intent', 'employee asana project management tool source truth tasks goals priorities challen
1037
```

圖二 詐騙關鍵字詞袋模型

```
from collections import Counter

# calculate word frequency(decending sorted)
word_counter = Counter(" ".join(corpus).split())

# choose the number of keywords to display
num_top_words = 50
top_words = [word for word, _ in word_counter.most_common(num_top_words)]
top_frequencies = [word_counter[word] for word in top_words]

for word, freq in zip(top_words, top_frequencies):
    print(f"{word}: {freq}")
```

```
money: 120
account: 103
card: 84
security: 79
credit: 61
problem: 58
support: 51
bank: 47
virus: 45
address: 43
```

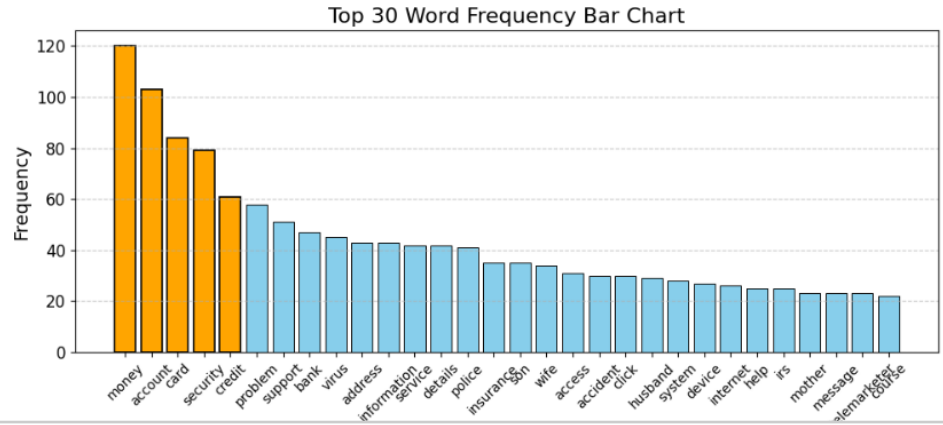
圖三 詐騙關鍵字詞頻統計

```python
import matplotlib.pyplot as plt

# choose the number of keywords to display
num_top_words = 30
top_words = [word for word, _ in word_counter.most_common(num_top_words)]
top_frequencies = [word_counter[word] for word in top_words]

plt.figure(figsize=(10, 5))
bars = plt.bar(top_words, top_frequencies, color="skyblue", edgecolor="black", linewidth=0.7)
# Highlight the top 5 keywords
for bar in bars[:5]:
    bar.set_color("orange")
    bar.set_edgecolor("black")
    bar.set_linewidth(1.2)

plt.title("Top {} Word Frequency Bar Chart".format(num_top_words), fontsize=16)
plt.xlabel("Words", fontsize=14)
plt.ylabel("Frequency", fontsize=14)
plt.xticks(rotation=45, fontsize=10)
plt.yticks(fontsize=12)
plt.grid(axis="y", linestyle="--", alpha=0.7)
plt.tight_layout()
plt.show()
```

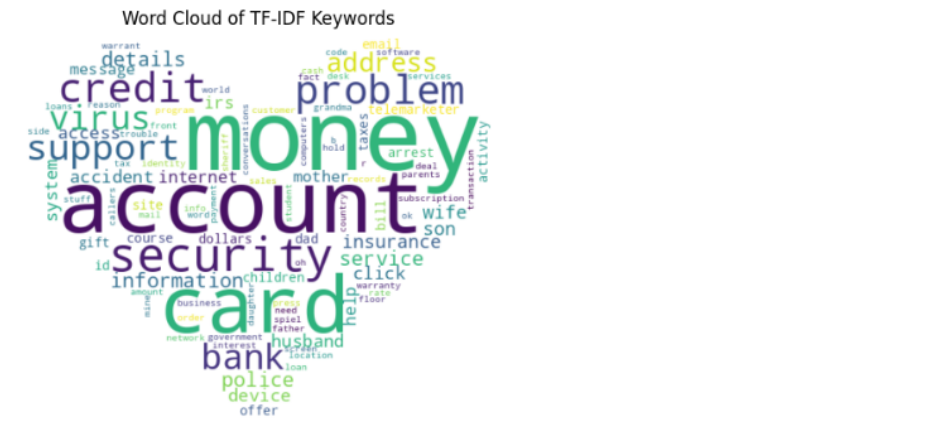


圖四 詐騙關鍵字高詞頻統計圖表

```python
from wordcloud import WordCloud
from PIL import Image
import numpy as np

# generate word cloud
mask = np.array(Image.open("/content/drive/MyDrive/人工智慧實務專題/正式專案/NLP/heart.png"))
wordcloud = WordCloud(width=800, height=400, background_color="white", colormap="viridis", max_words=100, mask=mask)
wordcloud.generate_from_frequencies(frequencies=word_counter)

plt.figure(figsize=(10, 5))
plt.imshow(wordcloud, interpolation="bilinear")
plt.title("Word Cloud of TF-IDF Keywords")
plt.axis("off")
plt.show()
```



圖五 詐騙關鍵字高詞頻文字雲

```python
import pandas as pd

# eliminate texts with NaN
df = df.dropna()
df = df.reset_index(drop=True)

# Initialize the result table
result = pd.DataFrame(columns=["ID", "Money related 1", "Money related 2", "Money related 3", "Money related 4",
                               "Money related 5", "Money related 6", "Money related 7", "Money related 8", "Money related 9",
                               "Personal information related", "Inducement words related", "Promotional content related", "Kinship related"])

# Money related keywords
money_keywords = ["money", "card", "credit", "bank", "insurance", "taxes", "bill", "dollars", "warrant"]

# Iterate through each keyword
for i, keyword in enumerate(money_keywords):
    # Find texts containing the keyword
    keyword_texts = df[df["Texts"].str.contains(keyword, case=False)] # ignore uppercase and lowercase

    # Initialize a list to store IDs for the current keyword
    keyword_ids = []

    # Iterate through each text containing the keyword
    for _, row in keyword_texts.iterrows():
        # Check if the current keyword is the first keyword or if the previous keyword is in the text
        if i == 0 or any(prev_keyword in row["Texts"].lower() for prev_keyword in money_keywords[:i]):
            # Append the ID to the list of IDs for the current keyword
            keyword_ids.append(row["ID"])
            # Update the corresponding column in the result table
            result.loc[result["ID"].isin(keyword_ids), f"Money related {i+1}"] = keyword

    # Store new IDs and keywords in the result table
    new_ids = list(set(keyword_texts["ID"]) - set(result["ID"]))
    new_data = pd.DataFrame({"ID": new_ids, f"Money related {i+1}": keyword})
    result = pd.concat([result, new_data], ignore_index=True)

# Fill missing values with 0
result = result.fillna(0)

# Personal information related keywords
personal_keywords = ["account", "address", "information", "details", "id", "email"]

# Inducement words related keywords
inducement_keywords = ["security", "problem", "support", "police", "virus", "accident", "irs", "message", "help", "arrest", "government", "device", "records", "access"]
```

```python
# Promotional content related keywords
promotional_keywords = ["service", "internet", "system", "telemarketer", "course", "offer", "site", "sales", "gift", "order", "program", "activity", "software", "subscription", "customer"]

# Kinship related keywords
kinship_keywords = ["son", "husband", "wife", "mother", "dad", "children", "parents"]

# Iterate through each ID
for idx, row in result.iterrows():
    # Check for personal information related keywords
    for keyword in personal_keywords:
        if keyword in df.loc[row["ID"], "Texts"].lower():
            result.at[idx, "Personal information related"] = keyword
            break

    # Check for inducement words related keywords
    for keyword in inducement_keywords:
        if keyword in df.loc[row["ID"], "Texts"].lower():
            result.at[idx, "Inducement words related"] = keyword
            break

    # Check for promotional content related keywords
    for keyword in promotional_keywords:
        if keyword in df.loc[row["ID"], "Texts"].lower():
            result.at[idx, "Promotional content related"] = keyword
            break

    # Check for kinship related keywords
    for keyword in kinship_keywords:
        if keyword in df.loc[row["ID"], "Texts"].lower():
            result.at[idx, "Kinship related"] = keyword
            break
```

圖六 依詐騙關鍵字詞頻取得資料值

	ID	Money related 1	Money related 2	Money related 3	Money related 4	Money related 5	Money related 6	Money related 7	Money related 8	Money related 9	Personal information related	Inducement words related	Promotional content related	Kinship related	Label
0	3	money	0	0	0	0	0	0	0	0	id	help	0	mother	1
88	5	0	card	credit	0	0	0	0	0	0	address	irs	0	0	1
224	7	0	0	0	0	0	0	0	0	warrant	0	support	0	0	1
89	9	0	card	credit	0	0	0	0	0	0	account	0	service	husband	1
187	12	0	0	0	0	0	taxes	0	0	0	id	police	0	0	1
4	14	money	card	0	0	0	0	0	0	0	address	0	0	0	1
178	16	0	0	0	0	insurance	0	bill	0	0	0	records	offer	0	1
97	19	0	card	credit	0	0	0	0	0	0	id	0	0	0	1
8	20	money	0	0	bank	0	0	0	0	0	account	0	software	0	1
10	22	money	0	0	0	0	0	0	0	0	id	security	service	0	1
183	23	0	0	0	0	insurance	0	0	0	0	0	0	0	0	1
11	25	money	0	0	bank	0	0	0	0	0	account	irs	0	son	1
102	26	0	card	credit	0	0	0	0	0	0	id	0	service	son	1
103	27	0	card	credit	0	0	0	0	0	0	0	police	gift	children	1
14	31	money	0	0	0	0	0	0	0	0	id	irs	0	0	1
221	34	0	0	0	0	0	0	0	0	warrant	id	support	0	son	1
106	36	0	card	0	0	0	0	0	0	0	address	0	0	0	1
186	38	0	0	0	0	0	taxes	0	0	warrant	0	problem	0	0	1
225	40	0	0	0	0	0	0	0	0	warrant	id	police	0	son	1

圖七 詐騙真實資料集

● 即時來電詐騙偵測手機應用程式開發

在本研究中，採用深度學習方法對所建立之生成仿真資料集進行建模，將所產生之詐騙分析結果，透過所開發之即時來電詐騙偵測手機應用程式實現客戶端應用，藉由即時偵測通話內容提醒使用者有詐騙風險的來電。在本研究中，未來媒體實驗室金融詐欺守門員研究團隊之研究步驟，包含手機應用程式系統開發，手機應用程式介面設計與手機應用程式系統改良與測試。在手機應用程式系統開發方面，本研究採用 Swift UI 程式語言進行開發，程式執行環境為 ios；採用 Swift 套件進行手機應用程式功能開發，包含偵測來電(使用 CallKit 的 CXCallObserver 模組)，錄音功能(使用 AVFoundation 的 AVAudioRecorder 模組)及介面設計(使用 ContentView 模組)，並與 OpenAI API 串接，進行語音轉文字辨識。在手機應用程式介面設計部分，採用全域導覽頁面(Tab)呈現手機應用程式架構，以引導式互動設計流暢簡易的操作流程，包含偵測(如圖八(b)所示)、紀錄(如圖八(c)所示)頁面，使用者回饋頁面提供其回報特定的詐騙關鍵字，及資訊頁面顯示開發團隊與程式版本資訊；使用者點擊介面下方按鈕可切換頁面。

使用者開啟手機應用程式後進入首頁動畫(如圖八(a)所示)，按下偵測按鈕變成紅色(如圖八(b)所示)，顯示開啟電話偵測防護機制，應用程式在背景執行來電偵測；當偵測到來電時即開啟錄音功能，將語音內容音訊檔儲存於緩存區，接著透過 OpenAI API 將語音轉文字處理，傳送到後端模型運算，通話內容錄音檔及模型分析結果皆顯示於紀錄頁面以列表顯示(如圖八(c)所示)，此通話錄音檔亦可作為金融犯罪的證據；所有紀錄來電包含通話時間，來電者及其電話號碼(非聯絡人以未知來電顯示) 及詐騙分析結果，介面設計簡明易懂，方便使用者快速了解來電相關資訊，及人工智慧判斷是否為詐騙電話的原因；使用者點擊分析按鈕，顯示此來電的詐騙機率，通話內容所擷取的詐騙判斷關鍵字及建議採取措施(如圖八(d)所示)；並提供刪除選項(如圖八(c)

所示)，使用者可選擇刪除此通話錄音，介面設計防錯機制，如二次確認刪除及刪除警示顏色。在本研究中，所開發之即時來電詐騙偵測手機應用程式，適用於此智慧型手機普及的移動媒體時代，具備高度實用價值。

(a)　　(b)　　(c)　　(d)

圖八 詐騙偵測手機應用程式介面設計

肆、社會影響力及永續發展

警民合作：本研究所開發之深度學習電話詐騙檢測模型，及後端資料庫所保存之詐騙電話錄音檔，亦可與政府部門及檢警機關合作，共同打擊和偵破電話詐騙案件。

- 有效取得數據及資料：透過電話內容即時監測和警示機制，幫助執法機關快速蒐集證據。
- 加速蒐證效率：自動分析通話內容，並將可疑內容標記呈現，方便執法機關調查。
- 有效緝拿與整理潛在加害人資訊：藉由對通話內容的分析，追蹤潛在加害人，並整理相關資訊，為執法機關提供破案線索。

商業價值：在本研究中，所開發之即時來電詐騙偵測手機應用程式，未來亦可與電信業者或手機廠商業者合作，具備商業經濟效益。

SDGs 和平、正義及健全制度：本研究可以預防電話詐騙犯罪，保障一般民眾財產安全，從而促成社會法治公平。

伍、參考資料

- Z. L. Liu*, "Legal system-oriented telecom fraud detection, identification and prevention," 2023. Thesis. School of Law, Hebei Finance University.
- Jian-jia Su and Yun-nung Chen, "Modeling Real-Time Call Behaviors for Fraudulent Phone Call Detection," 2019. Thesis. National Taiwan University.
- Johan H van Heerden, "Detecting Fraud in Cellular Telephone Networks," 2005. Thesis. Operational Analysis University of Stellenbosch.