

臺北科技大學資訊工程系

113 學年度實務專題計畫成果報告

結合語言模型與特徵機制 之整合式網路入侵偵測告警系統

專題編號：113-CSIE-SXXX

專題計畫參與人員：110590006 顏睿寬

110590024 許宸瑋

110590029 陳思群

指導教授：孫勤昱 教授

執行期間：112 年 1 學期至 113 年 1 學期

摘要

專題名稱：結合語言模型與特徵機制之整合式網路入侵偵測告警系統

頁數：十九頁

校所別：國立臺北科技大學 資訊工程系

畢業時間：一百一十三學年度第二學期

學位：學士

專題計畫參與人員：顏睿寬、許宸瑋、陳思群

指導教授：孫勤昱 教授

關鍵詞：網路安全、入侵偵測系統、深度學習、語言模型、CICIDS2017

隨著網際網路的急速發展，我們現在能夠透過多種網路搜尋引擎（例如 Google [1]）與最近熱門的生成式 AI 聊天機器人（例如 ChatGPT [2]）來輕鬆地取得所需的資訊或回答，這無疑使我們的生活更加便利。

然而，隨之而來的是各種安全威脅和攻擊，像是釣魚攻擊（Phishing）、分散式阻斷服務（Distributed Denial-of-Service, DDoS）、SQL 注入攻擊（SQL injection）等，這些都是我們在網路上時常會面臨的風險。只要我們使用網際網路，就會時刻面對這些風險與威脅。因此，在享受網路便利的同時，確保個人安全已成為一個極為重要的課題。

為了解決這個問題，本研究主要利用 CICIDS2017 資料集[3]來訓練一個自然語言處理（Natural Language Processing, NLP）模型。這個模型將能夠有效地分辨網路封包是否包含惡意攻擊。同時，本研究結合 TShark [4]來定期監控骨幹網路中的封包，並將擷取到的資料經過一些處理後傳送給我們訓練出來的模型。透過這種方式，我們可以判斷流量中是否存在具惡意行為的封包。一旦檢測到惡意封包，系統將向管理者發送警告，告知其惡意流量的相關資訊，提醒管理者注意封包來源。

本研究成果有望對網路安全防護領域提供更有力的支援。

目錄

摘要.....	i
目錄.....	ii
第一章 緒論.....	1
1.1 研究動機.....	1
1.2 研究目標.....	1
第二章 背景知識.....	2
2.1 研究背景.....	2
2.2 研究方法.....	2
2.2.1 訓練資料.....	2
2.2.2 深度學習框架.....	3
2.2.3 預訓練語言模型.....	3
2.2.4 模型訓練可視化工具.....	4
2.2.5 擷取封包.....	4
2.2.6 提取特徵.....	5
2.2.7 入侵偵測系統 (IDS)	5
2.3 文獻回顧與探討.....	6
第三章 研究步驟.....	7
3.1 建構語言模型.....	7
3.2 擷取封包並提取特徵.....	7
3.3 建構入侵偵測系統.....	7
第四章 實驗與結果.....	8
4.1 流量占比不平衡之預處理.....	8
4.2 建構流量分析語言模型.....	10
4.3 遷移學習 (Transfer Learning)	12
4.4 實際擷取封包與提取特徵.....	13
4.5 實作入侵偵測系統 (IDS).....	14
第五章 結論與未來方向.....	16
5.1 結論.....	16
5.2 未來方向.....	16
5.3 專題心得.....	16
第六章 參考文獻.....	18

第一章 緒論

1.1 研究動機

隨著科技的迅速發展，網路攻擊手法變得更加複雜與隱匿。舉例來說，常見的網路釣魚攻擊，利用了電子郵件或簡訊等方式夾帶偽造的網站鏈結以誘導目標進行操作，試圖竊取敏感資訊或引導目標安裝惡意軟體。另一種常見例子是分散式阻斷服務攻擊，這種攻擊利用大量的殭屍機器向目標網站或伺服器發送大量請求封包，使其無法正常運作。除此之外，還有 SQL 注入攻擊。這是一種常見且具有破壞性的攻擊手法。在 SQL 注入攻擊中，攻擊者通過向應用程式的輸入字串中，試圖串接惡意的 SQL 指令，從而對資料庫進行非法存取或執行惡意操作。這種攻擊不但會導致敏感資料的洩露、資料庫的破壞，甚至是整個系統的控制權被攻擊者所掌握 [5]。

傳統的網路安全防護手段，例如防火牆、入侵偵測系統等，雖然可以對抗一些較簡單的攻擊，但對於新型態的威脅卻顯得力不從心。例如，近年來出現的隱蔽性高且難以檢測的零時差漏洞攻擊（Zero-Day Vulnerability）[6]，這種攻擊利用系統或應用程式中未被發現的漏洞，從而進行未知的攻擊，傳統的防護手段往往難以及時發現和防禦。

本研究的動機在於提高網路入侵偵測的效能與精準度，有效地來保護使用者的網路安全。傳統的 IDS 往往受限於僅能捕捉靜態特徵，對於動態和隱匿性攻擊的偵測能力有所不足。因此，透過深度學習技術結合語言模型，我們能夠更全面地捕捉網路封包中的關鍵特徵，並對其進行有效分析。

1.2 研究目標

透過此研究，我們希望達到以下目的，為用戶提供更安全的網路環境：

- 準確度：開發一個高精度的異常檢測模型，確保在不同情境下都能達到卓越的準確率。
- 模型優化：逐步調整模型訓練參數，最大程度地減少模型收斂時間，進一步加速訓練流程，提升效率。
- 部署與應用：實現定時監控功能，並開發具備即時告警機制的系統，確保在異常情況下能夠快速響應。
- 持續學習與防禦：使模型具備自我再訓練能力，能有效應對新型攻擊手法，提升系統防禦的全面性與持續性。

第二章 背景知識

2.1 研究背景

本研究旨在利用深度學習技術，結合特徵機制與語言模型，開發一個能夠有效偵測網路中惡意流量並傳送警告的系統。

我們的首要目標是建構一個高效的流量分析模型，透過現有資料集進行測試，確保模型能達到預期的準確性。在這個階段，我們使用了現成的工具來評估不同的模型表現，並透過比較找出最適合的解決方案。接下來，我們從網路流量中提取多種特徵，並將這些資料與模型結合進行深入分析。我們定期進行流量監控，並處理獲取到的資料，確保資料完整且適合分析。

隨後，我們構建一個系統，當偵測到異常流量時，會即時提醒使用者，以加強網路安全。這些提示會包含一些基本資訊，幫助使用者快速了解潛在的風險。此外，我們還設計了模型的持續學習機制，確保模型能應對不斷變化的新型威脅。

2.2 研究方法

2.2.1 訓練資料

本研究使用網路入侵偵測系統的標竿資料集 CIC-IDS2017 (Canadian Institute for Cybersecurity Intrusion Detection Systems 2017)[7]。

CIC-IDS2017 資料集是由加拿大網路安全研究院 (Canadian Institute for Cybersecurity) 於 2017 年釋出，用於評估入侵偵測系統 (IDS) 和入侵防禦系統 (IPS) 效能。它解決了現有資料集缺乏流量多樣性和即時性的問題，提供了包括良性和最新攻擊在內的真實世界流量。

該資料集的生成過程包括了使用 B-Profile 系統生成自然的良性背景流量，以及在特定時間段內施行不同類型的攻擊，如 Brute Force FTP、Brute Force SSH、DoS、Heartbleed、Web Attack 等。資料集還包括了攻擊和良性流量的標記資料，以及從產生的網路流量中提取的 80 多個網路流量特徵。這些特徵可作為訓練入侵偵測系統的依據，幫助提高系統的準確性和效能[3]。

該資料集的建立符合可靠的基準，包括完整的網路設定、完整流量捕獲、標記資料集、完整的互動和攻擊多樣性等要求[8]。

2.2.2 深度學習框架

在建構語言模型的過程中，我們採用了 Python 語言中的 Simple Transformers 套件[9]。Simple Transformers 是一個基於 Transformers 和 Hugging Face 函式庫的工具，專為解決自然語言處理（NLP）任務而設計。它提供了一個簡單使用的介面，讓使用者能夠輕鬆地建立、訓練和部署各種 NLP 模型，如文本分類、情感分析、命名實體識別等。這個工具具有幾個主要特點。首先，它的易用性非常高，使用者可以通過簡單而直觀的 API 快速開始建立和訓練模型，無需深入了解 Transformers 和深度學習的細節。其次，它支援多種 NLP 任務，涵蓋了文本處理的各個方面。它基於 Hugging Face 的 Transformers 庫，使用預訓練的 Transformer 模型，如 BERT、RoBERTa、ALBERT 等，從而具有強大的性能和擴展性。同時，Simple Transformers 支援在 GPU 和 TPU 上進行模型訓練，能夠加速訓練過程，提高效率。最後，它提供了豐富的示例代碼和詳細的文檔，幫助使用者快速上手並解決問題。

2.2.3 預訓練語言模型

在預訓練語言模型的選擇上，我們預計優先嘗試 BERT 和 RoBERTa 等模型，因為這兩個模型在語言模型領域具有較大的影響力和廣泛的應用。此外，我們也考慮嘗試 ALBERT 模型，因為我們在後續的實作方面需要一個更輕量級、更小巧，同時依舊保持較高性能水平的語言模型，以提高整體效率。最後，使用相同的參數設置，包括 Epoch、Batch size、Iteration 等等，觀察不同模型所訓練出來的結果，並決定最適合我們需求的訓練模型。

BERT (Bidirectional Encoder Representations from Transformers) 模型其特色為一種雙向的預訓練語言模型，通過雙向遞歸神經網路 (Bidirectional Transformer) 從大量無標籤文本中學習詞彙的表徵。BERT 使用了 Transformer 的多層編碼器，這些編碼器能夠將輸入序列轉換為一系列隱藏表示，同時保留了序列中每個位置的上下文信息。而 BERT 進一步將這些表示稱為上下文敏感的「Token Embedding」，並通過將隱藏表示與固定的位置嵌入結合，以保持位置信息。這樣的設計改變了傳統的從左到右或從右到左的單向模型，使 BERT 模型能夠理解整個文本序列的上下文和關聯性，並在處理 NLP 任務時表現出色。BERT 模型通常通過預訓練和微調兩個階段來使用。在預訓練階段，模型通過大量未標記的文本數據進行自我監督訓練，從而學習到語言的通用表示。在微調階段，模型通過在特定任務的標記數據上進行微調，以適應該任務的特定要求。

RoBERTa (Robustly optimized BERT approach) 是一種基於自注意力機制的預訓練語言表示模型。它是基於 BERT 模型進行優化和擴展的，旨在解決 BERT 中存在的一些問題並提高性能。RoBERTa 通過使用更長的訓練時間、更大的訓練數據集、移除下一句預測任務、動態調整訓練參數等方式來改進 BERT。

ALBERT (A Lite BERT) 是一種基於 BERT 模型的輕量級版本。它針對 BERT 存在的一些問題進行了優化，旨在提高模型的效率和性能。ALBERT 通常具有比原始的 BERT 模型更小的模型尺寸和更高的效率，同時在各種自然語言處理任務中表現出色。ALBERT 通常會對 BERT 模型的一些關鍵組件進行改進，如詞彙表大小、嵌入維度、隱藏層大小等，以實現更好的性能和更高的效率。

2.2.4 模型訓練可視化工具

在評估我們模型效能的過程中，我們使用了 Wandb (Weights and Biases) 這個工具，它是一個專門為機器學習開發者設計的實驗追蹤和監控平台。

Wandb 主要用來實時記錄並可視化模型訓練過程中的各種指標，如損失值、準確度、精確率、召回率等，這使我們能夠即時檢查模型的表現，即時發現異常並進行調整。除此之外，Wandb 讓我們能夠方便地管理多個實驗，將不同的訓練結果進行比較，並保存所有的訓練配置、數據版本、模型架構等資訊，這對於後續的實驗複現以及團隊間的協作都非常有幫助。

透過其強大的報告功能，我們可以生成詳細的可視化圖表，並進行深入的分析。此外，Wandb 支援與 Simple Transformers 整合，能在分佈式訓練中高效追蹤各個節點的數據。這不僅提升了我們調參數和測試不同模型的效率，還能幫助我們更準確地了解模型的性能表現，並選擇最合適的模型進行部署。

2.2.5 擷取封包

在此階段中，我們使用 TShark 工具，它是 Wireshark 的命令行版本，也是一款開源的網路封包分析軟體。TShark 可以捕獲並分析經過計算機網路的數據封包，並且更適合用於自動化監控和腳本化操作。我們利用 TShark 進行定期的排程監控，每隔 3 分鐘捕獲一次封包，並將捕獲到的封包以 pcap 檔案格式保存。由於 TShark 支持命令行操作，這使得我們能夠更方便地將其整合到自動化流程中，例如，定時執行封包捕獲，並自動存儲結果，進行後續的資料處理和分析。

2.2.6 提取特徵

在此階段中，我們使用 CICFlowMeter 工具，從這些 pcap 檔案中提取封包的特徵訊息。

CICFlowMeter 是一款基於 Java 的網路流量分析工具，專門用於生成雙向流量（Biflow）並提取出多達 80 多個維度的特徵，包括封包數量、封包長度、流量持續時間等。這些特徵分別從流量的前向（來源到目的地）和反向（目的地到來源）進行計算，因此能夠提供更詳細的數據分析。CICFlowMeter 可以從 pcap 檔案中讀取網路封包，並將每個 TCP 和 UDP 的流量以 Flow 單位進行處理，最終生成 CSV 檔案格式的數據報告。

該工具的靈活性在於，使用者可以根據需要調整和添加特徵，並控制流量超時的持續時間。它已經被廣泛應用於多個網路安全相關的數據集，例如 Android Adware 和 DDoS 攻擊的數據集。這使得 CICFlowMeter 成為研究網路異常檢測的重要工具。

2.2.7 入侵偵測系統（IDS）

在此階段中，我們利用 Python 的 CustomTkinter 套件在 Windows 系統上建立彈出式視窗，作為我們入侵偵測系統的使用者介面。

與 Tkinter 相比，CustomTkinter 提供了更多的自定義選項，如漸層色彩、圓角按鈕、以及更靈活的版面配置。這使我們能夠在不影響使用者體驗的前提下，設計出更現代化且符合需求的視窗界面。此彈出式視窗顯示了關於檢測到的可疑流量的詳細信息，如時間戳、來源 IP 地址、攻擊手法、以及使用的協議，幫助使用者更快速地作出反應。CustomTkinter 的 API 讓我們能更輕鬆地控制視窗樣式和佈局，提供更豐富的互動介面。

這樣的設計使得我們的入侵偵測系統不僅能夠即時提醒使用者，還能確保介面操作簡單，視覺上更加現代化。

2.3 文獻回顧與探討

為深入了解語言模型與特徵機制之整合式 IDS 研究現況與關聯影響，翻閱許多文獻和研究。本計畫使用網路安全、入侵偵測系統、深度學習、語言模型、CICIDS2017 等不同關鍵字組合，搜尋與本計畫相關之論文，並挑選重要的期刊論文進行研讀。

Ahmad 等人[10] 於 2020 年回顧了基於 Machine Learning, ML 和 Deep Learning, DL 方法的網路入侵檢測機制。研究顯示，大約 80% 的提出的解決方案都是基於 DL 方法，其中 Autoencoder 和 Deep Neural Network 是最常用的演算法，不過基於 ML 的解決方案較簡易、需要較少的運算資源。研究亦提出強調了訓練時應使用 CSE-CIC-IDS2018 等較新的資料集，最後，本文強調了在實現低頻攻擊的模型性能和減少提出模型的複雜性方面的研究差距，並表示未來研究的方向可能為使用較不複雜的 DL 演算法、並盡量確保模型的有效性。

Dini 等人[11] 於 2023 年對機器學習在入侵偵測系統中的應用做了深度的研究，並著重於資料集、演算法和性能。結果顯示，使用決策樹和隨機森林選定資料集時，無論是二元分類還是多類分類，始終表現優於其他模型。此外，研究探索了深度學習模型的潛力，強調了其在各種任務中的有效性，並且對於高流量網路極其重要，也可以整合進嵌入式系統中。

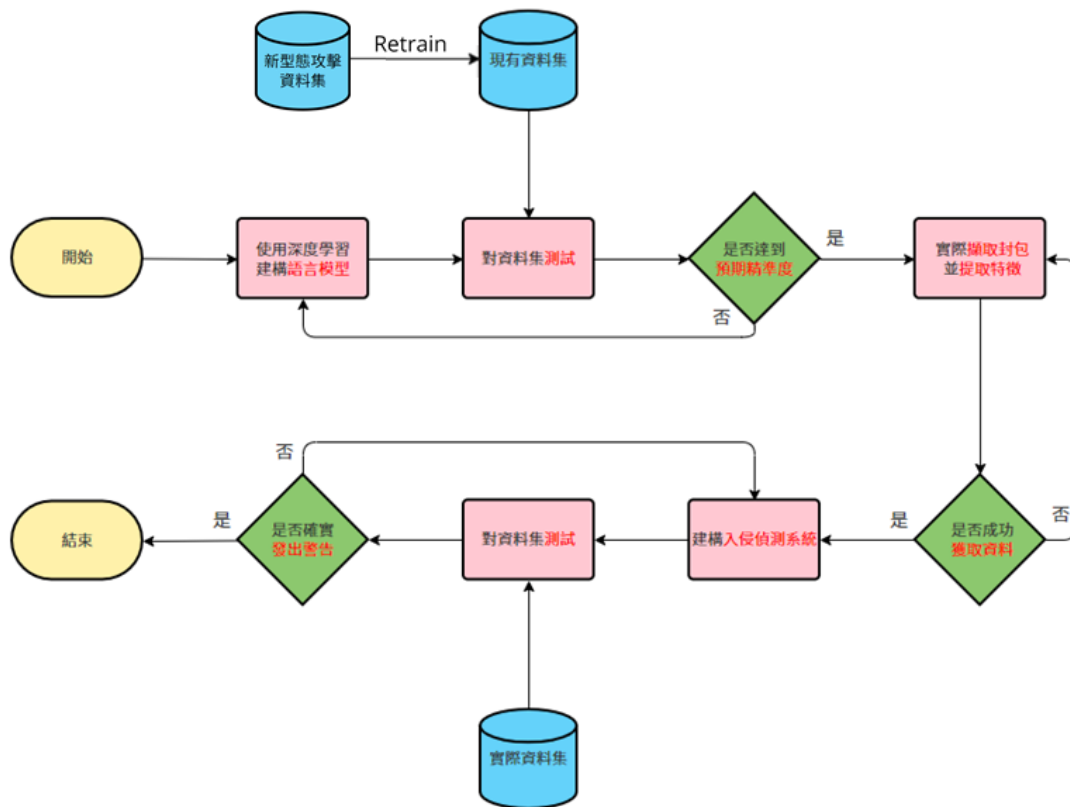
Kumar 等人[12] 在 2010 年討論了人工智慧技術在入侵偵測系統中的應用及優點和限制，並提出需要被解決的問題包含自動調整高效率的辨識技術、特徵縮減技術、高速處理即時分析、低強度攻擊識別。

Dr. Tiberiu-Marian Georgescu [13] 在 2020 年討論了關於自然語言處理在自動分析與資安相關文件上的應用，包含了如何利用自然語言處理技術來分析與資安相關的文件，例如安全漏洞報告、威脅情報、安全策略等。

Benyamin Ghogh [14] 等人在 2019 年發布的論文中回顧了不同的常見特徵選擇和提取方法的理論和動機，並介紹了它們的一些應用，以及展示了這些方法的一些數值實現。最後，將幾種方法進行了比較。

透過查閱上述論文得知，IDS 系統在當今的網路安全中扮演著至關重要的角色。了解到訓練資料集的選擇至關重要，因此我們以 CICIDS2017 為核心、佐以其他資料集，以提高模型的準確性和普遍性。未來研究方向包括高速即時分析、使用更簡單的演算法及輕量的模型，並且本次研究將嘗試模仿出一套類似的系統。

第三章 研究步驟



圖一：研究流程圖

專案開發流程如圖一所示，包括建構語言模型、擷取封包並提取特徵、建構入侵偵測系統。

3.1 建構語言模型

使用 Python 的 SimpleTransformers 套件來建構語言模型，並透過多種評估指標來分析和比較模型的性能，最終選出表現最佳的模型，以確保達到我們的預期目標。

3.2 擷取封包並提取特徵

使用 TShark 工具定期捕獲網路封包，並藉助 CICFlowMeter 提取封包中的各項特徵，最後將這些特徵匯出並儲存為 CSV 檔案，便於後續的分析與處理。

3.3 建構入侵偵測系統

使用 CustomTkinter 建構入侵偵測系統，當系統偵測到惡意攻擊時，會在螢幕上彈出警告視窗、紀錄 log，並自動發送 Gmail 通知給管理員，確保及時應對潛在的安全威脅。

第四章 實驗與結果

4.1 流量占比不平衡之預處理

CIC-IDS2017 資料集涵蓋了各種不同類型的網路活動，旨在模擬現實世界中的網路攻擊和正常流量情境。其中包括正常流量以及 14 種不同種類的攻擊，相對應的資料與筆數如表一所示。然而，由於正常流量占比較多，而一些惡意流量的數量相對較少，導致實驗結果未達預期效果。

為解決這種不平衡的情況，我們對資料集進行了預處理，將資料按標籤分為 15 個子資料集。接著，從 14 種不同攻擊類型的資料集中分別選取 80% 作為訓練集，20% 作為測試集，同時調整正常流量和惡意流量的比例為 1:1，相對應的資料與筆數如表二所示。最後，在每次執行時，對資料進行打亂處理，確保了流量占比和資料分布的平衡性。這樣的處理方式有助於提升實驗結果的準確性和可信度。

表一：CICIDS 2017 攻擊資料集的資料分布情形

標籤 (Label)	樣本數 (Samples)	佔比 (Composition)
BENIGN	2273097	80.301%
DoS Hulk	231073	8.163%
PortScan	158930	5.615%
DDoS	128027	4.523%
DoS GoldenEye	10293	0.364%
FTP-Patator	7938	0.281%
SSH-Patator	5897	0.209%
DoS slowloris	5796	0.205%
DoS Slowhttptest	5499	0.195%
Bot	1966	0.07%
Web Attack-Brute Force	1507	0.054%
Web Attack - XSS	652	0.024%
Infiltration	36	0.002%
Web Attack- Sql Injection	21	0.001%
Heartbleed	11	0.001%
Total	2830743	100%

表二:調整後 CICIDS 2017 攻擊資料集的資料分布情形

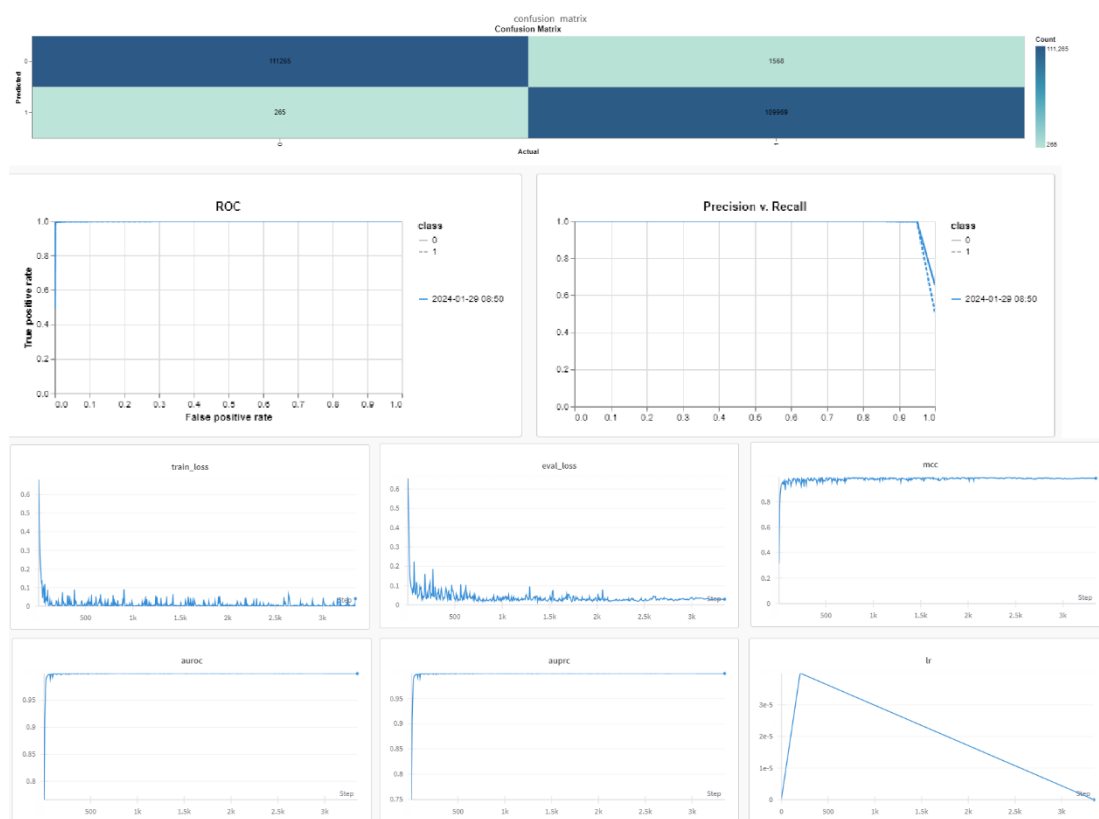
標籤 (Label)	樣本數 (Samples)	佔比 (Composition)
BENIGN	557649	50%
DoS Hulk	231073	20.719%
PortScan	158930	14.25%
DDoS	128027	11.479%
DoS GoldenEye	10293	0.923%
FTP-Patator	7938	0.712%
SSH-Patator	5897	0.529%
DoS slowloris	5796	0.52%
DoS Slowhttptest	5499	0.493%
Bot	1966	0.176%
Web Attack-Brute Force	1507	0.135%
Web Attack - XSS	652	0.058%
Infiltration	36	0.003%
Web Attack- Sql Injection	21	0.002%
Heartbleed	11	0.001%
Total	1115295	100%

4.2 建構流量分析語言模型

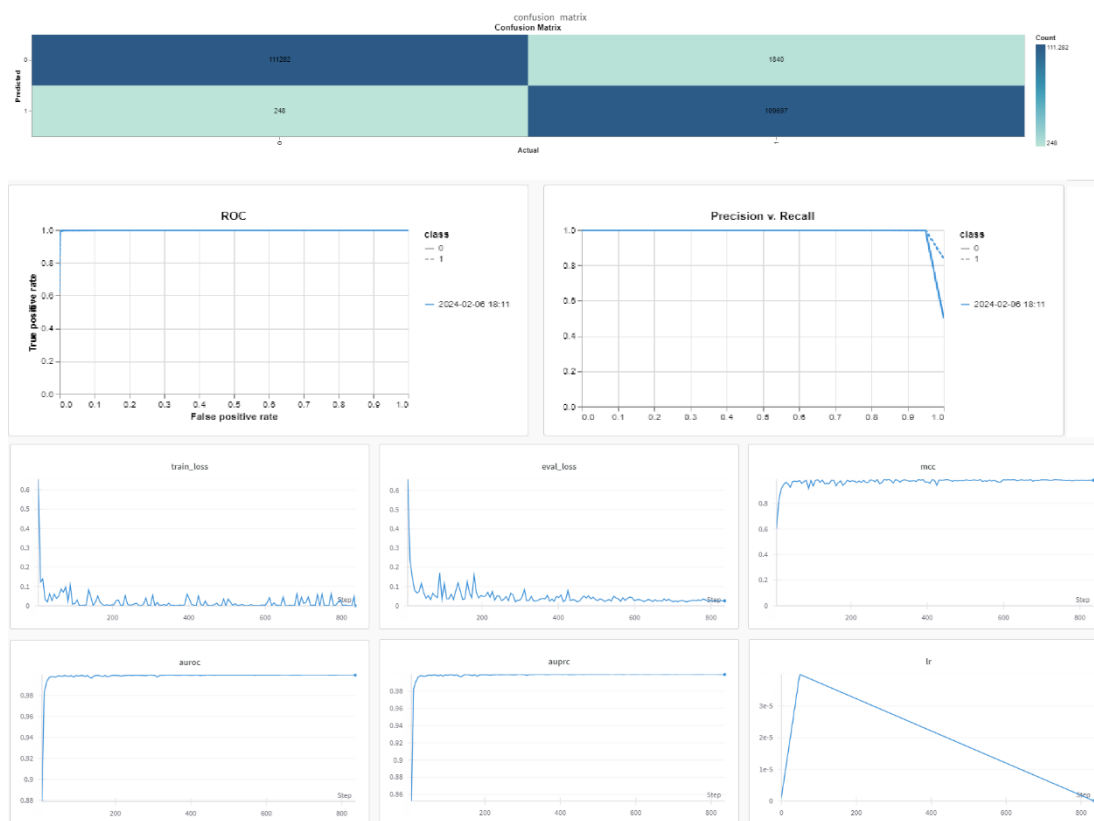
在對經過預處理的資料集進行訓練時，我們採用了 SimpleTransformers 工具，並對多種預訓練語言模型進行了測試，包括 BERT、RoBERTa 和 ALBERT（如圖二、圖三、圖四所示）。在相同的參數設置下，我們對這些模型進行了性能比較，並使用了多種評估指標來判斷其表現，包括混淆矩陣（Confusion Matrix）、精確率（Precision）、召回率（Recall）、ROC 曲線（Receiver Operating Characteristic Curve）以及 F1 score。

通過這些評估，我們全面分析了各模型在分類任務中的效率。結果顯示，儘管 BERT 和 RoBERTa 的表現也十分優秀，但在相似的準確度和 F1 score 下，ALBERT 憑藉其輕量化的設計和更低的計算成本，成為最具效率的選擇。因此，我們最終選擇了 ALBERT 作為本研究的核心模型，既能在保持高效能的同時大幅降低資源消耗，對於大規模資料集的應用具有更高的實用性和擴展性。

這樣的選擇不僅提高了模型的效能，同時優化了系統的運行效率，確保在實際應用中能以更小的資源消耗完成高精度的語言處理任務。



圖二：BERT 訓練數據



圖三：RoBERTa 訓練數據



圖四：ALBERT 訓練數據

4.3 遷移學習 (Transfer Learning)

在入侵偵測模型的研究過程中，一個常見的疑慮是：「這個模型是否只能有效處理訓練資料中已知的攻擊手法？」「如果遇到新型態的攻擊，該如何應對？」針對這些挑戰，我們使用了遷移學習技術，以解決模型對新型態攻擊防禦的不足。

遷移學習的核心理念是透過將先前訓練過的模型作為基礎，並額外加入新型態攻擊的訓練資料進行再訓練。具體來說，我們將原本訓練於 ALBERT 模型之上的預訓練模型，進一步訓練以涵蓋新型態攻擊數據。這種方法不僅能保持模型對舊型攻擊的強大檢測能力，同時也能使模型對於新出現的攻擊類型具備更強的防禦能力。

在實驗設計中，我們分別比較了三種模型：

- 僅使用 CICIDS2017 訓練的模型。
- 整合 CICIDS2017、CICIDS2018 和 CICIDS2019 訓練的模型。
- 基於 CICIDS2017 訓練，並對 CICIDS2018 和 CICIDS2019 資料進行遷移學習的模型。

測試集選取了部分 CICIDS2019 的數據進行驗證（詳見表三）。測試結果顯示，透過遷移學習技術訓練的模型綜合表現最佳，顯著優於僅透過原始數據進行訓練的模型。即便是看過測試集的模型，其表現也無法與遷移學習模型相媲美，這可能歸因於直接訓練過大資料集容易導致模型的泛化效果差，從而無法應對新型態的攻擊手法。

透過這次實驗，我們進一步證明了遷移學習對於模型在處理新型態網路攻擊上的重要性。這項技術不僅能夠提升模型對於未見數據的適應性，也能確保模型在面對多變的網路威脅環境中保持高效且精準的防禦能力，這對於現代網路安全系統的構建至關重要。

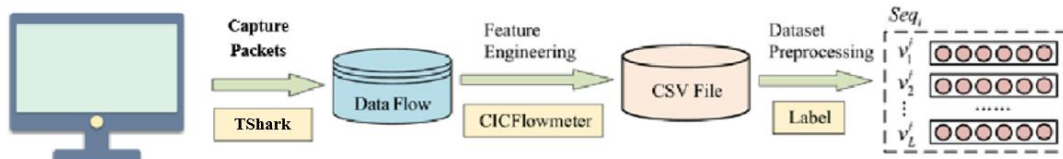
測試集	僅使用 CICIDS 2017	整合所有訓練集	遷移學習
惡性資料	5.23%	9.79%	76.08%
良性資料	99.64%	97.08%	75.66%

表三:測試不同模型遇新型態攻擊之結果

4.4 實際擷取封包與提取特徵

在此階段中，我們使用到 TShark 和 CICFlowMeter 兩個重要工具，流程步驟如圖五所示，細節如下：

首先，我們使用 TShark 工具進行定期的封包捕獲和監控排程，每隔 3 分鐘自動捕獲一次封包並將其保存為 pcap 檔案格式（如圖六所示）。接著，我們利用 CICFlowMeter 工具對捕獲的 pcap 檔案進一步的處理與分析，提取封包中的各種流量特徵資訊（如圖七所示）。最後，我們進行資料的前處理，主要任務包括移除無關的 Label，並確保所保留的大約 80 個 Label 以字串格式呈現。此過程的目的是確保資料的完整性和一致性，從而提高模型在異常檢測中的判斷準確性。



圖五：實際擷取封包與提取特徵之流程圖

圖六：未經過 CICFlowMeter 提取特徵

#	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	Flow ID	Src IP	Src Port	Dst IP	Dst Port	Protocol	Timestamp	Flow Dst	Tot Fwd Pkts	Tot Bwd Pkts	Tot Len	Fwd Len	Bwd Len	Tot Fwd Pkts	Tot Bwd Pkts	Tot Len	Fwd Len	Bwd Len	Tot Fwd Pkts	Tot Bwd Pkts	Tot Len	Fwd Len	Bwd Len	Tot Fwd Pkts	Tot Bwd Pkts	Tot Len
2	142.251.42.142:42	443	140.124.1.140	59687		6.16092024	516	1	73	73	73	73	73	0	0	0	0	0	141472.9	3875.969	516	0	516	516	0	516
3	140.124.1.140:140	60305	52.168.112	443	6.16092024	5847958	12	14	958	5181	517	0	79.83333	186.2993	1460	0	370.0714	602.5338	1049.768	4.445996	233918.3	356363.5	1015468	0	0	0
4	140.124.1.140:140	5553	224.0.0.25	5553	17.16092024	1016517	8	0	248	0	31	31	31	0	0	0	0	0	0	0	0	0	0	0	0	0
5	140.124.1.140:140	5553	224.0.0.25	5553	17.16092024	1009894	2	0	86	43	43	43	43	0	0	0	0	0	0	0	0	0	0	0	0	0
6	140.124.1.140:140	5553	224.0.0.25	5553	17.16092024	9466979	34	0	1054	0	31	31	31	0	0	0	0	0	0	0	0	0	0	0	0	0
7	140.124.1.140:140	63370	224.0.0.25	5553	17.16092024	410018	2	0	50	0	25	25	25	0	0	0	0	0	0	0	0	0	0	0	0	0
8	140.124.1.140:140	56967	224.0.0.25	5553	17.16092024	411502	2	0	50	0	25	25	25	0	0	0	0	0	0	0	0	0	0	0	0	0
9	140.124.1.140:140	5553	224.0.0.25	5553	17.16092024	19627	12	0	810	0	77	39	67.5	17.18615	0	0	0	0	0	0	0	0	0	0	0	0
10	140.124.1.140:140	51902	224.0.0.25	5553	17.16092024	409543	2	0	50	0	25	25	25	0	0	0	0	0	0	0	0	0	0	0	0	0
11	140.124.1.140:140	54933	229.255.22	1900	17.16092024	3019237	4	0	700	0	175	175	175	0	0	0	0	0	0	0	0	0	0	0	0	0
12	140.124.1.140:140	63333	224.0.0.25	5553	17.16092024	409509	2	0	50	0	25	25	25	0	0	0	0	0	0	0	0	0	0	0	0	0
13	140.124.1.140:140	53	140.124.1	63798	17.16092024	3049238	3	0	252	0	84	84	84	0	0	0	0	0	0	0	0	0	0	0	0	0
14	0.0.0.0-255.0.0.0	68	255.255.22	1900	17.16092024	9317502	48	0	13842	0	300	270	284	2083	6.912703	0	0	0	0	0	0	0	0	0	0	0
15	140.124.1.140:140	50485	229.255.22	1900	17.16092024	6005920	3	0	411	0	137	137	137	0	0	0	0	0	0	0	0	0	0	0	0	0
16	140.124.1.140:140	62820	140.124.1	7680	6.16092024	3022544	3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
17	140.124.1.140:140	64533	224.0.0.25	5553	17.16092024	424702	2	0	60	0	30	30	30	0	0	0	0	0	0	0	0	0	0	0	0	0
18	140.124.1.140:140	137	140.124.1	137	17.16092024	6494236	7	0	350	0	50	50	50	0	0	0	0	0	0	0	0	0	0	0	0	0
19	140.124.1.140:140	137	140.124.1	137	17.16092024	3911576	5	0	250	0	50	50	50	0	0	0	0	0	0	0	0	0	0	0	0	0
20	140.124.1.140:140	5553	224.0.0.25	5553	17.16092024	9459662	24	0	864	0	36	36	36	0	0	0	0	0	0	0	0	0	0	0	0	0
21	140.124.1.140:140	60168	224.0.0.25	5553	17.16092024	414793	2	0	60	0	30	30	30	0	0	0	0	0	0	0	0	0	0	0	0	0
22	140.124.1.140:140	58427	224.0.0.25	5553	17.16092024	422038	2	0	92	0	46	46	46	0	0	0	0	0	0	0	0	0	0	0	0	0
23	140.124.1.140:140	137	140.124.1	137	17.16092024	1502762	3	0	150	0	50	50	50	0	0	0	0	0	0	0	0	0	0	0	0	0
24	140.124.1.140:140	5553	224.0.0.25	5553	17.16092024	416577	2	0	60	0	30	30	30	0	0	0	0	0	0	0	0	0	0	0	0	0
25	140.124.1.140:140	53902	224.0.0.25	5553	17.16092024	410137	2	0	50	0	25	25	25	0	0	0	0	0	0	0	0	0	0	0	0	0
26	140.124.1.140:140	62155	229.255.22	1900	17.16092024	3023779	4	0	700	0	175	175	175	0	0	0	0	0	0	0	0	0	0	0	0	0
27	140.124.1.140:140	5553	224.0.0.25	5553	17.16092024	782529	6	0	1649	0	545	41	274.8333	256.9237	0	0	0	0	0	0	0	0	0	0	0	0
28	140.124.1.140:140	49794	224.0.0.25	5553	17.16092024	409804	2	0	50	0	25	25	25	0	0	0	0	0	0	0	0	0	0	0	0	0
29	140.124.1.140:140	61288	140.124.1	7680	6.16092024	4011467	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
30	140.124.1.140:140	5553	224.0.0.25	5553	17.16092024	1013708	4	0	112	0	28	28	28	0	0	0	0	0	0	0	0	0	0	0	0	0
31	140.124.1.140:140	53	140.124.1	33153	17.16092024	5006160	2	0	144	0	72	72	72	0	0	0	0	0	0	0	0	0	0	0	0	0
32	140.124.1.140:140	137	140.124.1	137	17.16092024	4631296	6	0	300	0	50	50	50	0	0	0	0	0	0	0	0	0	0	0	0	0
33	140.124.1.140:140	64381	224.0.0.25	5553	17.16092024	410051	2	0	50	0	25	25	25	0	0	0	0	0	0	0	0	0	0	0	0	0

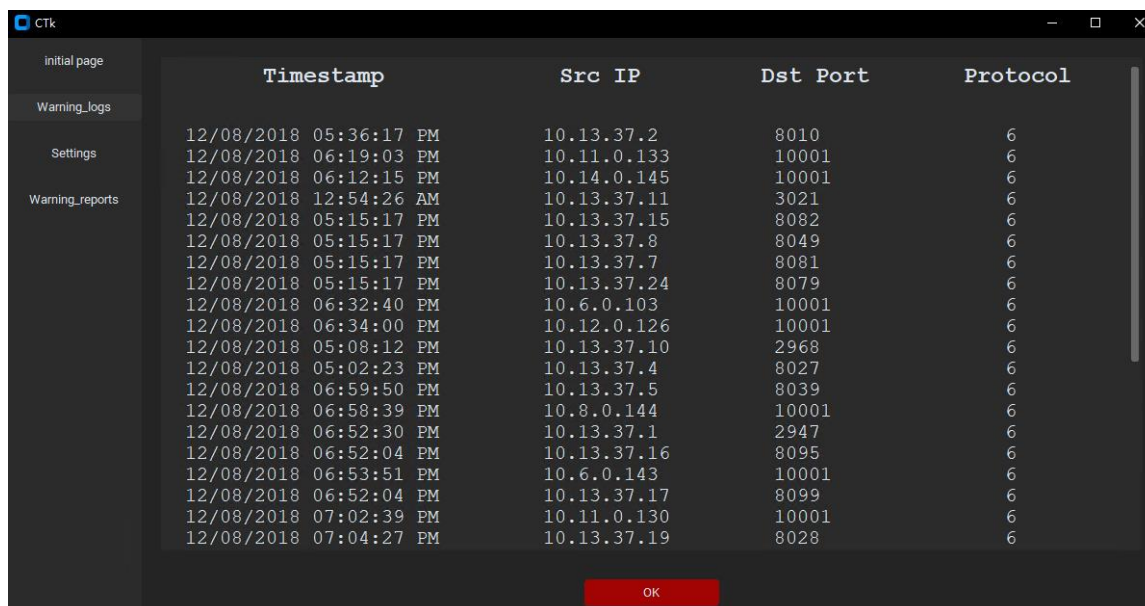
圖七：經過 CICFlowMeter 提取特徵

4.5 實作入侵偵測系統 (IDS)

在此階段中，我們使用 Python 的 CustomTkinter 套件 來開發入侵偵測系統的使用者介面，實現即時的彈出式視窗功能。作為此系統的核心視覺介面，彈出視窗會在系統檢測到可疑或惡意流量時自動觸發，並向使用者展示關鍵的安全資訊（如圖八、圖九所示）。這些資訊包括時間戳記、來源 IP 地址、目標連接埠、通訊協議以及過去攻擊事件的統計資料。

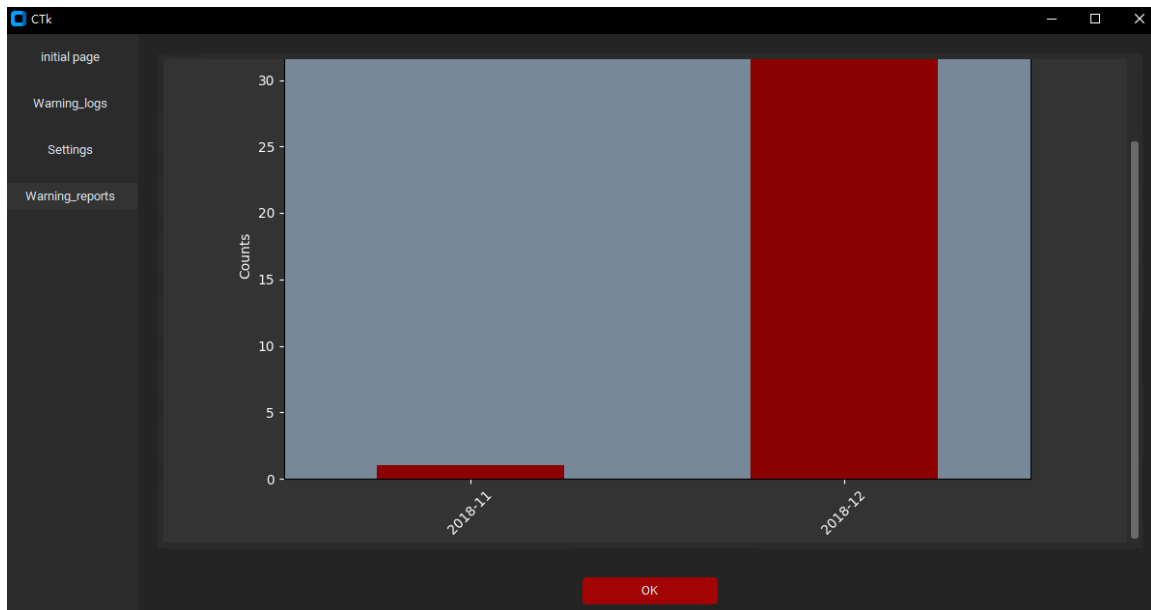
為了確保視窗的可用性與及時性，我們利用 CustomTkinter 提供的元件（如標籤、列表框、按鈕等）來構建一個簡單且功能豐富的介面。使用內建的佈局管理器（例如 grid 和 pack）來確保視窗內容的排列整齊，並適應不同的顯示器尺寸。

在數據處理方面，我們通過從入侵偵測系統獲取的實時數據來驅動視窗顯示，包括惡意流量的詳細資訊。這樣，使用者能夠即時了解系統檢測到的潛在安全威脅，並能夠快速採取防護措施。此外，我們還實現了自動郵件通知功能（如圖十所示），當系統偵測到惡意攻擊時，將立即發送郵件給系統管理員，進一步提高系統的應急響應能力。這不僅確保了入侵偵測系統的即時性，還增強了其可操作性和安全性。



	Timestamp	Src IP	Dst Port	Protocol
Initial page	12/08/2018 05:36:17 PM	10.13.37.2	8010	6
Warning_logs	12/08/2018 06:19:03 PM	10.11.0.133	10001	6
Settings	12/08/2018 06:12:15 PM	10.14.0.145	10001	6
Warning_reports	12/08/2018 12:54:26 AM	10.13.37.11	3021	6
	12/08/2018 05:15:17 PM	10.13.37.15	8082	6
	12/08/2018 05:15:17 PM	10.13.37.8	8049	6
	12/08/2018 05:15:17 PM	10.13.37.7	8081	6
	12/08/2018 05:15:17 PM	10.13.37.24	8079	6
	12/08/2018 06:32:40 PM	10.6.0.103	10001	6
	12/08/2018 06:34:00 PM	10.12.0.126	10001	6
	12/08/2018 05:08:12 PM	10.13.37.10	2968	6
	12/08/2018 05:02:23 PM	10.13.37.4	8027	6
	12/08/2018 06:59:50 PM	10.13.37.5	8039	6
	12/08/2018 06:58:39 PM	10.8.0.144	10001	6
	12/08/2018 06:52:30 PM	10.13.37.1	2947	6
	12/08/2018 06:52:04 PM	10.13.37.16	8095	6
	12/08/2018 06:53:51 PM	10.6.0.143	10001	6
	12/08/2018 06:52:04 PM	10.13.37.17	8099	6
	12/08/2018 07:02:39 PM	10.11.0.130	10001	6
	12/08/2018 07:04:27 PM	10.13.37.19	8028	6

圖八:log 介面



圖九:數據統計介面

04/24/2024 19:55 Attack detection report [Trash x](#)



fakealerts@gmail.com
to me ▾

Your device is under attack

Timestamp	Src IP	Dst Port	Protocol
12/08/2018 05:36:17 PM	10.13.37.2	8010	6

[↩ Reply](#) [➡ Forward](#) [😊](#)

圖十:郵件通知畫面

第五章 結論與未來方向

5.1 結論

本專題使用 Simple Transformers 中的 ALBERT 預訓練模型，並以 CICIDS2017 資料集進行訓練，旨在提供能夠辨別封包特徵的能力，進而有效偵測網路攻擊。我們利用圖形化介面，讓使用者能夠即時掌握系統的監測結果以及過去的統計數據。為應對不斷變化的攻擊手法，我們採用遷移學習技術，對模型進行 retrain，並引入 CICIDS2018 和 CICIDS2019 資料集，這樣既能保留模型原有的精準度，又能提高對新型態攻擊的偵測能力。然而，儘管模型整體性能良好，但由於不同攻擊類型的資料量差異較大，我們發現常見攻擊的資料量往往極為龐大，而一些冷門或少見的攻擊類型則資料稀缺，這導致模型在偵測稀有攻擊時表現較為薄弱，這也是我們未來可以著重改善精進的部分。

5.2 未來方向

在如今人人離不開網路的時代，資安已經成為一個非常重要的課題。未來方向將聚焦於進一步提升模型性能與精準度。一個可行的方向是嘗試使用最新的預訓練模型和深度學習框架，這將有助於改進模型的運算能力和精準度。我們也將考慮引入更多的資料集，以涵蓋更廣泛的攻擊類型，並利用數據擴增技術平衡攻擊類型間的資料量差異，尤其是在處理稀有攻擊類型時，透過調整資料比例權重來強化模型表現，從而提升模型的泛化能力和實際應用價值。

5.3 專題心得

在這段為期一年到一年半的畢業專題製作過程中，我們面對了許多技術上的挑戰，並且從中獲得了寶貴的經驗。儘管在專題剛開始時，教授與學長姐為我們提供了一個明確的方向，但當我們將理論付諸實踐時，遇到的問題遠超預期。

一開始，我們在環境設置與測試階段時，嘗試使用 TensorFlow-GPU 讀取 GPU 卻屢次失敗。經過深入排查後才發現，問題出在版本不兼容：Cuda、Cudnn 和 TensorFlow-GPU 各自有對應的版本，這些版本還需要與 Python 版本相匹配。而我們一開始安裝的都是最新版本，導致系統無法正常運作。最終，經過大量資料查詢與版本調整，這個問題才得以解決。然而，隨後我們發現 Simple Transformers 並不支援 TensorFlow-GPU，因此不得不轉換至 PyTorch。幸好，有了之前在環境配置上的經驗，這次的切換過程相對順利，也讓我們更快進入正

軌。這一經歷教會我們一個重要的教訓：最新版本並不總是最佳選擇，過於追求最新技術反而可能帶來兼容性問題。另外，通過這次的實際操作，我們深刻體會到架設開發環境的核心重要性，這不僅是專題成功的基石，也是面對未來技術挑戰的關鍵經驗。儘管起步並不輕鬆，但正因這次經歷，我們在未來的專題進行中能更加得心應手。

在進行專題實作過程中，另一個讓我們印象深刻的挑戰是封包處理的環節。我們需要使用 CICFlowMeter 將抓取到的 pcap 封包檔案轉換成 CSV 格式，然而，CICFlowMeter 在 Windows 系統上僅支援圖形化介面，無法透過 Python 進行自動化操作。為了解決這個問題，我們只能跨平台使用 Linux 系統上的 CICFlowMeter，並首次嘗試利用 Docker 來搭建這個跨環境解決方案。然而，這是我們首次接觸 Docker 工具，過程中遇到了許多設置和操作上的問題，這段時間可以說是卡住了相當長的一段時間。但在教授與學長姐們的耐心指導下，透過反覆查閱資料和大量的 Google 搜索，我們終於順利解決了這個問題，成功完成了封包處理的工作。

在這說長不長，說短不短的專題實作期間，雖然遇到了不少挑戰和技術障礙，特別是在 Debug 上耗費了大量時間，但這些問題的解決也為我們積累了寶貴的經驗，讓我們對技術的理解更加深刻。俗話說得好，「有一種失敗，會讓你更成功。」我們認為失敗並不可恥，重要的是可以在這段失敗中學習到多少東西，可以成長多少。而在這次專題中，這一次次的失敗都讓我們確實地成長了，不只學到了許多相關領域的經驗與問題，還讓我們有了更深厚的友誼，以及更多團隊合作的經驗。團隊合作是將來不論是在職場上，還是在生活上都是不可或缺的一環，該如何去與他人配合，去找出最適合彼此的分工模式以及時間安排。在這次的專題中，都讓我們成長了許多，也更熟悉團隊分工的模式。

最後，最感謝的就是孫教授以及各位研究室學長姐們的協助，當我們遇到問題瓶頸時總會幫我們指引方向，遇到硬體上的問題也是會盡力地滿足我們需要的硬體需求，真的非常感謝您們這一年來的協助，辛苦了！

第六章 參考文獻

- [1] “What is Google?” Computer Hope, 2024. [Online]. Available: <https://www.computerhope.com/jargon/g/google.htm>. [Accessed: 16 2 2024].
- [2] “Introducing ChatGPT” OpenAI, 2022. [Online]. Available: <https://openai.com/blog/chatgpt>. [Accessed: 16 2 2024].
- [3] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, “Toward generating a new intrusion detection dataset and intrusion traffic characterization,” in *Proc. 4th Int. Conf. Inf. Syst. Security and Privacy (ICISSP)*, Portugal, Jan. 2018.
- [4] “tshark(1) Manual Page” Wireshark, 2024. [Online]. Available: <https://www.wireshark.org/docs/man-pages/tshark.html>. [Accessed: 17 9 2024].
- [5] “Common cyberattacks to look out for” Zoho Corporation Pvt. Ltd., 2023. [Online]. Available: <https://www.manageengine.com/log-management/cyber-security-attacks/common-types-of-cyber-attacks.html>. [Accessed: 7 2 2024].
- [6] L. Bilge and T. Dumitraş, “Before we knew it: an empirical study of zero-day attacks in the real world,” in *Proc. 2012 ACM Conf. Computer and Communications Security (CCS '12)*, New York, NY, USA, 2012, pp. 833-844. doi: 10.1145/2382196.2382284.
- [7] K. Salah and A. Kahtani, “Performance evaluation comparison of Snort NIDS under Linux and Windows Server,” *J. Netw. Comput. Appl.*, vol. 33, no. 1, pp. 6–15, 2010. doi: 10.1016/j.jnca.2009.07.005.
- [8] A. Gharib, I. Sharafaldin, A. H. Lashkari and A. A. Ghorbani, “An Evaluation Framework for Intrusion Detection Dataset,” *2016 International Conference on Information Science and Security (ICISS)*, Pattaya, Thailand, 2016, pp. 1-6, doi: 10.1109/ICISSEC.2016.7885840.
- [9] “About - Simple Transformers” Thilina Rajapakse, 2020. [Online]. Available: <https://simpletransformers.ai/about/>. [Accessed: 17 2 2024].
- [10] Z. Ahmad, S. Khan, C. Wai Shiang, J. Abdullah, and F. Ahmad, “Network intrusion detection system: A systematic study of machine learning and deep learning approaches,” *Trans. Emerging Telecommun. Technol.*, vol. 32, no. 3, p. e4150, 2021, doi: 10.1002/ett.4150.

- [11] P. Dini, A. Elhanashi, A. Begni, S. Saponara, Q. Zheng, and K. Gasmi, "Overview on Intrusion Detection Systems Design Exploiting Machine Learning for Networking Cybersecurity," *Applied Sciences*, vol. 13, no. 13, p. 7507, 2023, doi: 10.3390/app13137507.
- [12] G. Kumar, K. Kumar, and M. Sachdeva, "The use of artificial intelligence based techniques for intrusion detection: a review," *Artificial Intelligence Review*, vol. 34, pp. 369–387, 2010, doi: 10.1007/s10462-010-9179-5.
- [13] T.-M. Georgescu, "Natural Language Processing Model for Automatic Analysis of Cybersecurity-Related Documents," *Symmetry*, vol. 12, no. 3, p. 354, 2020, doi: 10.3390/sym12030354.
- [14] B. Ghogh, M. N. Samad, S. A. Mashhadi, T. Kapoor, W. Ali, F. Karray, and M. Crowley, "Feature Selection and Feature Extraction in Pattern Analysis: A Literature Review," *arXiv preprint arXiv:1905.02845*, 2019, doi: 10.48550/arXiv.1905.02845.