

國立高雄第一科技大學
資訊管理系

碩士論文

基於機器學習之黑箱資料隱碼安全檢測機制之研究

The Study of Black-box SQL Injection Security
Detection Mechanisms Based on Machine Learning

研 究 生：黃琮仁

指導教授：莊文勝 博士

中華民國 一〇六 年 十二 月

基於機器學習之黑箱資料隱碼安全檢測機制之研究
The Study of Black-box SQL Injection Security Detection Mechanisms
Based on Machine Learning

研 究 生：黃琮仁 Cong-Ren Huang

指導教授：莊文勝 Wen-Shenq Juang

國立高雄第一科技大學

資訊管理系

碩士論文

A Thesis Submitted to
Department of Information Management
National Kaohsiung First University of Science and Technology
in Partial Fulfillment of the Requirements
for the Degree of Master
In
Information Management

December 2017
Yenchao, Kaohsiung, Taiwan, Republic of China

中華民國 一〇六年 十二月

國立高雄第一科技大學學位論文考試審定書

_____ 資訊管理 _____ 系(所) ☒ 碩士班
☐ 博士班

研究生 _____ 黃琮仁 _____ 所提之論文

論文名稱(中文)：基於機器學習之黑箱資料隱碼安全檢測機制之研究

論文名稱(英/日/德文)：The Study of Black-box SQL Injection Security Detection Mechanisms Based on Machine Learning

_____ ☒ 碩士
經本委員會審查，符合 _____ 學位論文標準。
☐ 博士

學位考試委員會

召	集	人	<u>張弘毅</u>	
委	員		<u>張弘毅</u>	<u>莊文勝</u>
			_____	<u>朱彥宏</u>
			_____	_____
			_____	_____

指導教授 莊文勝
主任(所長) 黃文楨

中華民國 106 年 12 月 28 日

保存期限：永久

中文摘要

隨著資訊安全日漸被重視，金融企業對網站進行安全性檢測的意願也漸漸的提高，黑箱檢測大致分為軟體自動測試與人工測試。

軟體自動測試多為廠商事先設定好的弱點規則庫進行檢測，通常無法在存在網頁應用程式防火牆（Web Application Firewall）或入侵檢測系統（Intrusion-detection system）保護的網路環境下準確地找出安全問題，檢測結果容易有誤判或者未檢出的狀況，而人工測試會因專家的水平不同及時間有限等條件下對檢測報告的準確度也有相當大的影響。因此本研究設計出基於機器學習之黑箱資料隱碼安全檢測機制進而改善自動化測試的缺點並具有高擴充性和高準確性優點。

關鍵字：機器學習、推薦系統、決策樹、資料隱碼

Abstract

With the increasing emphasis on information security, financial industries are more willing to have security inspection for their websites. Black Box Testing can be divided into Software Automation Testing and Manually Testing. Software Automation Testing inspects the weakness policies database preinstalled by manufacturers. It cannot find security problems precisely when the network environment is protected by a web application firewall or an intrusion-detection system. The testing report may have misdetection or cannot find the problem of the system. Manually Testing will generate different testing reports that may be depended on tester's professional ability and the limited time. In this thesis, we design a black-box testing mechanism for detecting SQL injection based on Machine Learning. Our result can improve the drawbacks of automatic testing, and provide the advantages of high scalability and high accuracy.

Keyword: Machine Learning, Recommended system, Decision Tree, SQL Injection

致謝

感謝莊文勝老師在我大學時期時，就提供了我許多資訊安全方面的許多建議以及資源，幫助我參加許多的資安活動，因此在研究所的期間的學期能夠更加的順利。

在論文研究期間，感謝指導教授莊文勝老師細心與耐心的和我共同探討論文方向與題目，也感謝黃承龍老師教導資料探勘的知識，讓我對機器學習方法有更深的瞭解。

最後，要感謝的是我的家人在這段學習生涯中的支持與鼓勵，讓我能致力於學習，順利完成碩士學位，朝向下個人生目標邁進。



目錄

中文摘要	I
ABSTRACT	II
致謝	III
目錄	IV
圖目錄	VI
表目錄	VII
第一章、緒論	1
1.1 研究動機	1
1.2 研究目的	2
第二章、文獻探討	4
2.1 推薦系統	4
2.3.1 記憶導向為基礎的協同過濾	4
2.3.2 模型為基礎的協同過濾	4
2.2 餘弦相似性	5
2.3 決策樹	5
第三章、研究方法	7
3.1 機器學習模型訓練階段	8
3.1.1 產生模糊測試 (Fuzzing) 語句	9

3.1.2 產生場景環境	10
3.1.3 模糊測試 (Fuzzing)	12
3.1.4 訓練資料前處理	14
3.1.5 決策樹 (Decision Tree)	14
3.2 軟體自動化測試運作階段	15
3.2.1 冷啟動推薦	16
3.2.2 推薦階段	16
3.3 系統擴充	18
3.3.1 場景環境的擴充	18
3.3.2 SQL 語句的擴充	18
3.3.3 執行階段的維護	19
第四章、實驗分析	20
第五章、結論與未來研究方向	22
參考文獻	23

圖目錄

圖 1. SQL 語句經過流程	7
圖 2. 機器學習模型訓練步驟	8
圖 3. 軟體自動化測試運作階段	10
圖 4. 場景環境	11
圖 5. 模糊測試產生出的 SQL 語句	13
圖 6. 模糊測試 SQL 語句	13
圖 7. 軟體自動化測試運作階段	15



表目錄

表 1. SQL 語句和場景環境	7
表 2. SQL 等架語句	9
表 3. SQL 語句前處理	9
表 4. 模糊測試範例	12
表 5. 訓練資料處理	14
表 6. 決策樹	15
表 7. 推薦系統產生的問題	17
表 8. 場景環境的擴充	18
表 9. SQL 語句的擴充	19
表 10. 實際測試環境	20
表 11. 效果比較表	21

第一章、緒論

1.1 研究動機

OWASP(Open Web Application Security Project)[1] 是一個開放社群、非營利性組織，長期協助瞭解並改善網頁應用程式與網頁服務的安全性。他們其中一個 OWASP Top 10 計畫 (OWASP Top Ten Project) 每隔三年會發布十大安全弱點防護原則，而資料隱碼攻擊 (SQL Injection) 從 2004 以來都一直在榜單中，可見資料隱碼攻擊一直是網頁應用服務的一大安全問題。

資料隱碼攻擊是發生於在設計不良的程式，將使用者輸入的字串夾帶進 SQL 指令中，並送至資料庫伺服器中解析導致惡意 SQL 指令被執行。在實際檢測資料隱碼攻擊漏洞的方式不外乎分為黑箱檢測 (Black-Box Testing) 及白箱檢測 (White-Box Testing)，白箱檢測需要透過提供原始碼或和資料來進行檢測，若該企業沒有具備足夠專家知識的資安人員往往只能尋求外部的資安公司進行協助，因此企業就會擔心自家有價值的商業資訊或其他資訊的外流問題。

黑箱檢測方式不需要透過程式碼和資料就可以進行漏洞尋找，往往企業在委託外部公司進行檢測時會選擇這種較低風險的方式，黑箱檢測方式大致又分為軟體自動化測試和人工測試。軟體自動測試多半為軟體廠商使用事先設定好的規則庫去進行檢測，通常無法在複雜的真實環境下正常的找出安全問題，檢測結果很容易有誤判或者未檢出，例如：目標受到 Web

應用程式防火牆（Web Application Firewall）或入侵檢測系統（Intrusion-detection system）的保護。

人工檢測可在網站架構及環境不同的情況根據經驗及回應內容做出不同的檢測方式，因此擁有較高的準確度，但檢測專家的水平不同及時間有限的情況對報告的準確度也有相當大的影響，在一個中小型的網站可能就存在成千上萬的參數，要完全使用人工檢測就必須耗費相當大的人力成本。目前常見的平衡做法就是混合軟體自動化檢測先進行初步掃描判斷，再透過人工檢測方式混合進行，但還是沒有辦法讓軟體自動化檢測更接近人工檢測的水平。

1.2 研究目的

目前多數研究都專注在如何從攻擊流量、日誌或接收到的參數進行檢測，過濾掉具有惡意行為的封包，雖然這種方式可以大大提高駭客利用漏洞的難度，但治標不治本容易透過混淆或新型態的攻擊方式繞過檢測。因此本研究提出透過機器學習方式解決軟體自動測試無法在複雜的環境下根據經驗做出選擇合理的檢測封包。

透過機器學習方式來解決軟體自動測試時所選擇的模型需要具備缺項預測、巨量的錯誤資料容忍和能夠輕易擴增。需要具備缺項預測能力的原因在於檢測工具尚未發送任何封包到目標時並不會得知任何資訊，每當發送一個檢測封包所有得到回應的結果做為下次發送封包的依據，在現實環境中不可能當所有檢測封包檢測完畢後才進行預測，故需要具備缺項時就能夠進行預測。

需具備容忍巨量錯誤資料的原因在於檢測封包可能會遇到 Web 應用程式防火牆或入侵檢測系統的阻擋，導致回應內容並非由目標所回應，因此機器學習模型必須能容忍被阻擋所造成的錯誤資料。

為了因應現實環境中不停的改變，例如不斷更新的網站伺服器軟體、不同版本間會存在不同的特性、不同的資料庫伺服器軟體也存在不可兼容的語句。若當軟體或環境改變時機器學習模型無法輕易的擴充須不斷的重新訓練是非常耗時且不可行的。

在本研究中將透過有限的條件下進行模糊測試產生測試用的 SQL (Structured Query Language) 語句，再結合機器學習 (machine learning) 中的推薦系統 (Recommended system) 的協同過濾 (Collaborative Filtering) 技術及決策樹 (Decision Tree) 中的分類樹 (Classification Tree) 技術混合使用來進行資料隱碼弱點檢測。

第二章、文獻探討

2.1 推薦系統

推薦系統常見用於推薦音樂、電影、新聞、書籍等，透過不同的演算法或模型達到對於不同使用者推薦出個人化的可能會感興趣的清單。常見的推薦系統通常都會使用的技術是基於內容推薦(Content-Based)[2][3]、協同過濾推薦(Collaborative Filtering)[4][5]和混合方式推薦。

內容推薦是根據同類型的項目提取特徵進行訓練，例如一部電影可以提取導演和演員當特徵，但很多情況下會出現很難抽出項目特徵的問題。

協同過濾推薦藉由使用者的喜好或使用者的項目屬性之間的關係找出共同的喜好，進而預測使用者偏好。協同過濾常見有兩種基礎[6]：

2.3.1 記憶導向為基礎的協同過濾

記憶導向為基礎的協同過濾(Memory-based collaborative filtering)又分為使用者為基礎導向(User-based)和項目為基礎導向(Item-based)[7][8]。使用者為基礎導向是透過相似統計的方法計算出擁有類似項目的使用者(Nearest Neighbors)，通常是透過該項目的評分或者是否購買來做計算。項目為基礎的協同過濾是透過該項目與項目間的相似性去計算。

2.3.2 模型為基礎的協同過濾

模型為基礎的協同過濾(Model-based collaborative filtering)是根據歷史進行訓練而得到一個模型，再透過該模型進行使用者的喜好預

測，常見的模型有群集模型（Cluster model）、潛在因子模型（Latent factor model）和貝氏模組（Bayesian model）。

2.2 餘弦相似性

餘弦相似性（cosine similarity）是通過測量兩個向量之間夾角的餘弦值來計算他們的相似性[9]。假設 A 和 B 為兩個向量，透過餘弦定理計算出夾角 θ ，透過夾角的大小可以判斷兩向量的相似度。假定 A 和 B 為 n 維向量。

$$\text{similarity} = \cos \theta = \frac{\sum_{i=1}^n (A_i \times B_i)}{\sqrt{\sum_{i=1}^n (A_i)^2} \times \sqrt{\sum_{i=1}^n (B_i)^2}}$$

2.3 決策樹

機器學習中的決策樹（Decision Tree）[10] 是一個預測模型，他代表是對象屬性與對象值之間的一種映射關係。決策樹選出其中分類能力最佳的屬性項目做為樹的節點，結點不斷重複上面過程向下展開，直到滿足終止條件為止。決策樹可以用來分析數據資料也可以來預測使用，一般分成分類樹（Classification Tree）、回歸樹（Regression Tree）和 CART（Classification And Regression Tree）。分類樹通常用於離散類型（對錯，幾種狗的種類等）。回歸樹通常用於連續數值上（股價，年紀等）。CART 是結合分類樹和回歸樹的概念，可以處理混合有分類及數值資料。在建構決策樹時選擇屬性項目的時候會選擇最好的屬性當作結點，選擇屬性的方

式有很多種信息增益（Information gain）、獲利比率（gain ratio）和吉尼係數（Gini index）。



第三章、研究方法

本研究所提出基於機器學習之黑箱資料隱碼安全檢測機制，核心是透過推薦系統中記憶導向為基礎的協同過濾方式和決策樹中的分類樹達成。表 1 中的場景環境作用在於模擬現實環境，因為一個資料隱碼攻擊語句在一般情況下不會直接被送到資料庫伺服器去執行，通常封包進入到網頁伺服器前會先經過 Web 應用程式防火牆或入侵檢測系統的過濾後才會進到網頁伺服器。到達網頁伺服器處理後會將內容送到網頁開發人員所撰寫的網頁程式語言處理，最後到達資料庫伺服器如圖 1 的過程，在過程中的任何一個階段被過濾或是 SQL 語句執行錯誤就視同攻擊失敗，若失敗在表 1 中標示為 X，成功則標示成 V。

表 1. SQL 語句和場景環境

語句 場景環境	語句 1 and 1=1	語句 2 and 1=1 --	語句 3 and 1=1) --
A 場景	✓	✓	✗
B 場景	✗	✗	✓
預測場景		✗	

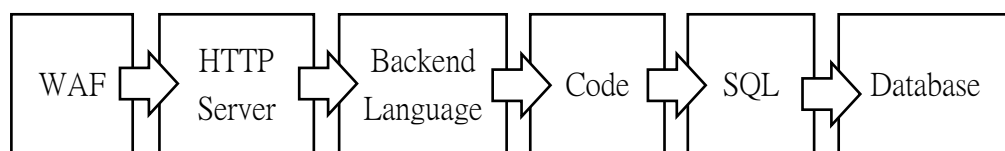


圖 1. SQL 語句經過流程

表 1 中的 A 場景和 B 場景表示為現實生活中的某個場景，A 場景中可以成功的 SQL 攻擊代碼為語句 1 和語句 2 其餘為失敗，B 場景中可以成功的 SQL 攻擊代碼為語句 3 其餘為失敗，以此類推。當表 1 中的預測場景測試過語句 2 後得到結果為失敗時，透過計算餘弦相似性計算出跟預測場景最接近的環境為 B 場景，而 B 場景中可執行成功的 SQL 語句為語句 3，因此會推薦出建議下一條攻擊測試語句為語句 3，詳細的細節將在後面介紹。本研究將會分成三大部分介紹：機器學習模型訓練階段、實際進行軟體自動化測試運作階段和系統擴增。

3.1 機器學習模型訓練階段



圖 2. 機器學習模型訓練步驟

3.1.1 產生模糊測試 (Fuzzing) 語句

在現實環境中進行人工測試時若遇到 Web 應用程式防火牆 (Web Application Firewall) 或入侵檢測系統 (Intrusion-detection system) 的阻擋時，人會根據過去的經驗判斷資料隱碼攻擊 (SQL Injection) 語句中是被定義成惡意特徵，人可以根據該特徵對 SQL 語句做出變異，產生出不同的等價語句，如表 2 中的被阻擋 SQL 語句可能因為等號被過濾，人可以嘗試將其變換為等價語句 A，若是等號和減號被過濾可替換成等價語句 B。

表 2. SQL 等價語句

被阻擋語句	and 1=1 --
等價語句 A	and 2>0 --
等價語句 B	and(1)#

為了盡可能產生最多可行的資料隱碼的測試語句就必須進行模糊測試，找出同一條 SQL 的指令可以替換成許多種方式表示，因此必須先對一般的測試語句進行處理，將該語句的特徵像是空白、數字、比較、條件字符替換成標示符號如：

表 3. SQL 語句前處理

處理前	and 1=1 --
處理後	and[Space][Num][Comp][Num][Space][Com]

資料隱碼的測試語句也會因為資料庫軟體的不同導致支援不同的語法，甚至相同軟體不同版本號對相同的語句也會有不同的解釋，

本研究實作時是產生 MYSQL 5.1 版本至 MYSQL 5.7 版本及可以通用在不同資料庫軟體的 SQL 語句，最後透過轉換後保存為 XML 格式如圖 3。

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <root>
3   <test>
4     <id>1</id>
5     <payload>[KEYWORD][WHITE][RANDNUM][COMP][RANDNUM]</payload>
6     <keyword>AND</keyword>
7     <comparison>=,&lt;,&gt;</comparison>
8     <comment>GENERIC</comment>
9     <white>GENERIC,MYSQL</white>
10  </test>
11  <test>
12    <id>2</id>
13    <payload>[KEYWORD][WHITE][RANDNUM][COMP][RANDNUM]</payload>
14    <keyword>AND</keyword>
15    <comparison>=,&lt;,&gt;</comparison>
16    <comment>#</comment>
17    <white>GENERIC,MYSQL</white>
18  </test>
19  <test>
20    <id>3</id>
21    <payload>[KEYWORD][WHITE][RANDNUM]</payload>
22    <keyword>AND</keyword>
23    <comment>#</comment>
24    <white>GENERIC,MYSQL</white>
25  </test>
```

圖 3. 軟體自動化測試運作階段

3.1.2 產生場景環境

場景環境作用在於盡可能模擬出真實環境會遇到的狀況，在建置時分成 WAF 層、HTTP Server 層、後端程式語言、程式碼層、SQL 層和資料庫層。每層中的場景元件盡量都透過 Dokcer 方式建置，原因在於如圖 4 中的 PHP 在不同版本間就有非常大的版本差異 PHP 5.3.4 版前存在 Null Byte 的問題，相同語法處理方式在不同版本可能也會略

有不同，因此要盡可能將每個元件都建立出足夠數量存在差異的軟體及版本。











WAF	HTTP Server	Backend Language	Code	SQL	Database
					
					
					
					

圖 4. 場景環境

每個場景環境是由每層中其中一種軟體和該軟體其中一種版本和不同過濾規則組合而成的如圖 4 中紅色框框為其中一種的軟體組合，因此透過不同的排列組合可以盡可能模擬出接近現實環境中的組合。在後端程式碼層中和 SQL 層是至關重要的一層，在現實環境中往往過

濾惡意的使用者輸入不會只依靠 Web 應用程式防火牆或入侵檢測系統，最容易遇到的是開發人員所撰寫的過濾程式，跟 WAF 層不同的是該層通常會直接將識別的惡意封包丟棄，而開發人員所撰寫的程式可能會出現取代、跳脫、黑白名單等，因此對惡意的輸入值處理過後會產生不同的結果。SQL 層是負責模擬現實中可能會產生的 SQL 語句類型，負責將前面層所傳遞過來的資料合併上 SQL 語句拋向資料庫層。

3.1.3 模糊測試 (Fuzzing)

在模糊測試階段會將 3.1.1 所產生的規則窮舉出所有的組合，但這個過程中並不是如表 4 中可窮舉出等價 A 和等價 B，如果繼續窮舉下去也可以產生等價 C，因此這個過程必須受到一定條件的限制，在限制條件下窮舉出所有的組合（圖 5）。接著將產生出來的 SQL 語句傳遞給場景環境執行，傳遞在不同層時被其中一層阻擋下來就視為執行失敗，最後傳遞到資料庫層時判別是否 SQL 語句是否可以被成功執行，若執行成功將其結果記錄下來（圖 6）。

表 4. 模糊測試範例

處理前	and[Space][Num][Comp][Num][Space][Com]
等價 A	and(1)=(1)--
等價 B	and((1))=(1)--
等價 C	and(((1)))=(1)--


```

1 'AND[0x09][RANDNUM]=[RANDNUM]--[0x09][RANDSTR]
2 'AND[0x0a][RANDNUM]=[RANDNUM]--[0x0a][RANDSTR]
3 'AND[0x0d][RANDNUM]=[RANDNUM]--[0x0d][RANDSTR]
4 'AND[0x20][RANDNUM]=[RANDNUM]--[0x20][RANDSTR]
5 'AND[0xa0][RANDNUM]=[RANDNUM]--[0xa0][RANDSTR]
6 'AND/*[RANDSTR]*/[RANDNUM]=[RANDNUM]--/*[RANDSTR]*/[RANDSTR]
7 'AND[0x09][RANDNUM]<[RANDNUM]--[0x09][RANDSTR]
8 'AND[0x0a][RANDNUM]<[RANDNUM]--[0x0a][RANDSTR]
9 'AND[0x0d][RANDNUM]<[RANDNUM]--[0x0d][RANDSTR]
10 'AND[0x20][RANDNUM]<[RANDNUM]--[0x20][RANDSTR]
11 'AND[0xa0][RANDNUM]<[RANDNUM]--[0xa0][RANDSTR]
12 'AND/*[RANDSTR]*/[RANDNUM]<[RANDNUM]--/*[RANDSTR]*/[RANDSTR]
13 'AND[0x09][RANDNUM]>[RANDNUM]--[0x09][RANDSTR]
14 'AND[0x0a][RANDNUM]>[RANDNUM]--[0x0a][RANDSTR]
15 'AND[0x0d][RANDNUM]>[RANDNUM]--[0x0d][RANDSTR]
16 'AND[0x20][RANDNUM]>[RANDNUM]--[0x20][RANDSTR]
17 'AND[0xa0][RANDNUM]>[RANDNUM]--[0xa0][RANDSTR]
18 'AND/*[RANDSTR]*/[RANDNUM]>[RANDNUM]--/*[RANDSTR]*/[RANDSTR]
19 "AND[0x09][RANDNUM]=[RANDNUM]--[0x09][RANDSTR]
20 "AND[0x0a][RANDNUM]=[RANDNUM]--[0x0a][RANDSTR]
21 "AND[0x0d][RANDNUM]=[RANDNUM]--[0x0d][RANDSTR]

```

圖 5. 模糊測試產生出的 SQL 語句

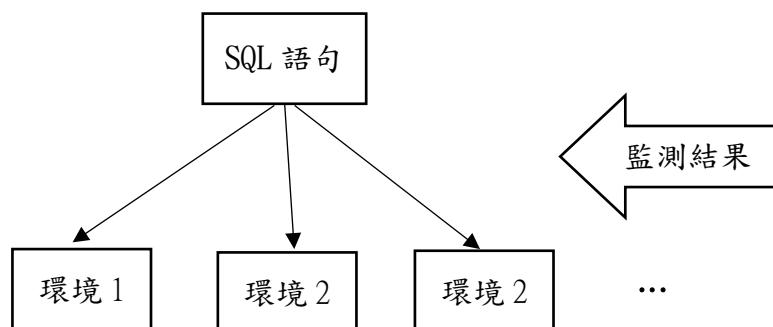


圖 6. 模糊測試 SQL 語句

3.1.4 訓練資料前處理

在 3.1.3 中模糊測試階段產生的資料在訓練模型前還須先去除一些多餘 SQL 語句和場景，例如在表 5 中語句 3 並沒有一個場景環境可以成功執行，代表模糊測試出來的此條語句可能是損毀的導致不能被執行成功，在場景 C 中無法成功執行任何 SQL 語句，代表該場景環境可能沒有辦法成功執行惡意的 SQL 指令，因此將這些沒有執行成功的 SQL 語句和場景環境從資料中去除。

表 5. 訓練資料處理

語句 場景環境	語句 1	語句 2	語句 3
A 場景	✓	✓	✗
B 場景	✗	✓	✗
C 場景	✗	✗	✗

3.1.5 決策樹 (Decision Tree)

在推薦系統中都存在冷啟動 (Cold-start) [11] 的問題，在本研究中在遇到一個未知的環境時也會出現冷啟動問題，在尚未對該環境送出任何的 SQL 語句時無法計算出與過去訓練時場景的相似度，對目標測試數量較少時也會面臨到推薦極度不準確的問題，因此結合了機器學習中的決策樹 (Decision Tree) 中的分類樹 (Classification Tree) 來解決冷啟動的問題。訓練時透過吉尼係數 (Gini index) 找出前 6 個最佳的 SQL 語句做為初始冷啟動階段時測試的語句。表 6 中的例子中 A、B 和 C 場景環境 A 和 B 較為相似，而 C 較為不同，因此透過決策樹

可以找到語句 3 為最好區分開不相似的場景環境，做為初始階段優先測試的語句可以有效幫助推薦系統更快找到最相似的場景環境。

表 6. 決策樹

語句 場景環境	語句 1 and 1=1	語句 2 and 1=1--	語句 3 and 1=1)--
A 場景	✓	✓	✗
B 場景	✓	✓	✗
C 場景	✗	✗	✓

3.2 軟體自動化測試運作階段

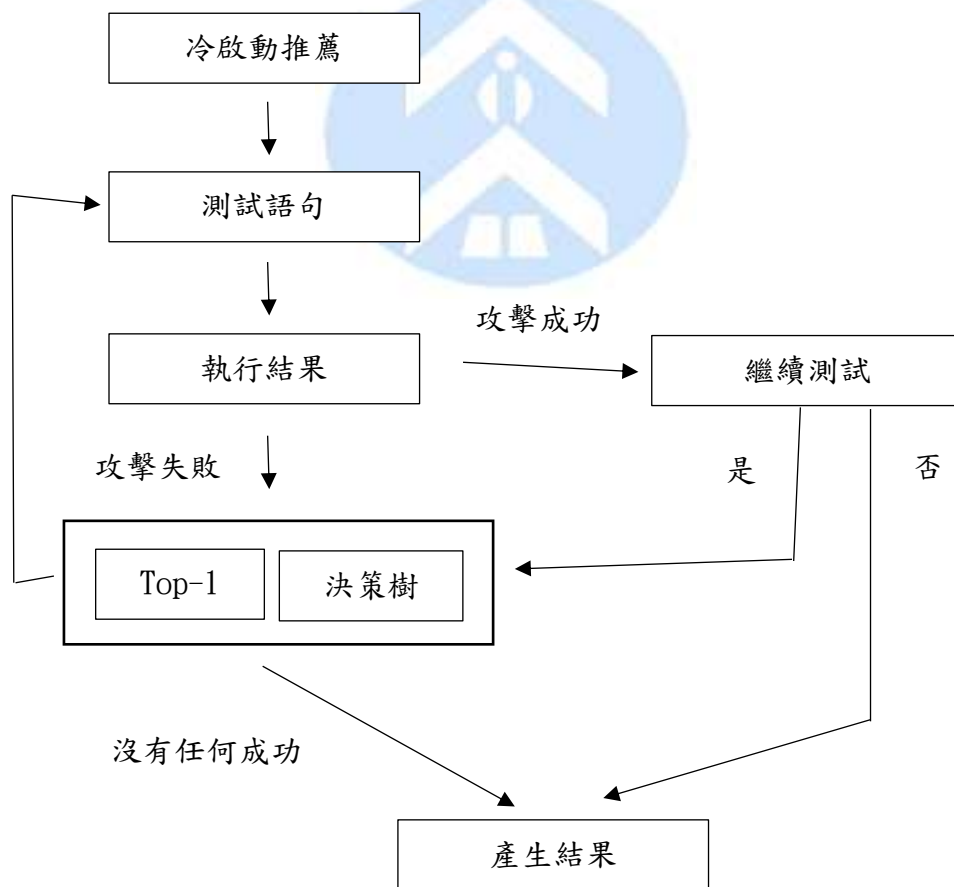


圖 7. 軟體自動化測試運作階段

3.2.1 冷啟動推薦

每當要測試一個未知環境時，冷啟動階段會先從事前訓練好的模型中取出 6 個 SQL 語句送到目標做測試，若有 SQL 語句執行成功代表存在 SQL 資料隱碼注入問題，若失敗則進入到推薦的階段。

3.2.2 推薦階段

本研究軟體自動化測試運作方式就如同人類，每當測試一個 SQL 語句時會根據回應的結果與過去的經驗選擇下一個最有可能執行成功的 SQL 語句，為了達成此目的本研究混合使用推薦系統和決策樹（Decision Tree）來進行推薦 SQL 語句的推薦。

推薦系統的部分使用的是協同過濾中的記憶導向為基礎的協同過濾（Memory-based collaborative filtering）使用者為基礎導向（User-based），透過計算餘弦相似性（cosine similarity）找到與自己最相近的 5 個場景環境，接著找出這 5 個場景環境中最高可能成功的 SQL 語句。

決策樹的部分使用決策樹中的分類樹（Classification Tree），與訓練階段的決策樹使用相同的方法但在做對模型的深度限制在 5 層的高度，避免即時運算時耗費過久的時間，另外這的決策樹與前面冷啟動所使用的決策樹功能不同，在這的目的是要解決推薦系統會產生的問題。假設表 7 中這五個場景都與測試目標為前 5 個相似的場景環境，而目前測試到的階段最相似的為 A，一路遞減最不相似的為 E 場景，假定現在測試的目標場景與 C 場景完全相同，但因為推薦系統會加總

出最有可能的 SQL 語句，因此會推薦語句 3，造成單純使用推薦系統做為推薦時需要花費較多的數次才能夠推薦到語句 5，在這個例子中透過決策樹可以很快的就選擇到語句 5，而在其他的情況下也可以幫助推薦系統用更少的次數推薦到對的 SQL 語句。

表 7. 推薦系統產生的問題

語句 場景環境	語句 1	語句 2	語句 3	語句 4	語句 5	...
A 場景	✓	✓	✓	✗	✗	...
B 場景	✓	✗	✓	✗	✗	...
C 場景	✗	✗	✗	✗	✓	...
D 場景	✓	✗	✓	✗	✗	...
E 場景	✗	✓	✓	✗	✗	...

在此階段推薦系統與決策樹會各選出一個 SQL 語句並給傳給下一個階段做測試，測試後根據結果與先前結果重新計算出推薦下一輪的 SQL 語句，接著不斷的一直重覆這一個循環，直到發現有可執行成功的 SQL 語句為止，當發現有可成功語句後可以選擇是否繼續尋找相同可使用的 SQL 語句。若測試到結束時都沒發現可執行成功的 SQL 語句，可選擇是否執行將從來不成功的 SQL 語句進行測試，假如測試結束後也沒發現可使用的 SQL 語句，代表可能沒有問題存在可利用。

3.3 系統擴充

系統擴充又分為場景環境的擴充、SQL 語句的擴充和執行階段的維護三大部分，本研究所提出的方法可以不需要像以往擴充時需要將模型重新訓練，因此任何的場景環境可以不需要完全依賴自己模擬建置，而是可以直接應用在現實的環境中進行訓練。

3.3.1 場景環境的擴充

每當需要擴充環境場景時只要新增一個場景環境的編號，接著將模糊測試所產生出來的 SQL 語句都進行測試一次，並將其結果記錄下來，甚至因為推薦系統的特性總算沒有測試完所有的 SQL 語句也可以完成擴充（表 8）。

表 8. 場景環境的擴充

語句 場景環境	語句 1	語句 2	語句 3	語句 4	語句 5	...
A 場景	✓	✓	✓	✗	✗	...
B 場景	✓	✗	✓	✗	✗	...
C 場景	✗	✗	✗	✗	✓	...
D 場景	✓	✗	✓	✗	✗	...
擴增		✓		✓	✗	...

3.3.2 SQL 語句的擴充

每當需要擴充一個 SQL 語句時只要新增一個 SQL 語句編號，接著將 SQL 語句傳到每個場景環境中進行測試，並將其結果記錄下來，但有時會因為有些場景可能是現實中的場景導致無法即時測試或其

他問題導致無法測試，與場景環境的擴充同樣可以允許未測試的空值存在（表 9）。

表 9. SQL 語句的擴充

語句 場景環境	語句 1	語句 2	語句 3	語句 4	語句 5	擴增
A 場景	✓	✓	✓	✗	✗	✗
B 場景	✓	✗	✓	✗	✗	✓
C 場景	✗	✗	✗	✗	✓	✗
D 場景	✓	✗	✓	✗	✗	✓
E 場景		✓		✓	✗	

3.3.3 執行階段的維護

在場景環境的擴充和 SQL 語句的擴充因為允許空值所以就衍生出後續執行階段上的維護，當在執行測試時會把結果與過去的場景有值的部分進行比對，當完全相符時且該場景存在空值欄位將可以選擇是否要測試空值欄位的 SQL 語句並將其結果存回。

第四章、實驗分析

為了檢測本研究是否擁有在有網頁應用程式防火牆或入侵檢測系統保護的網路環境下準確地找出安全問題，因此架設了三個實際的場景環境來做測試，場景 A 沒有任何 WAF 上的保護，伺服器層使用 Apache 2.4.26 及後端程式使用 PHP 5.6.30，程式碼中也完全沒對接收到的資料進行處理及檢驗，SQL 層是一個簡單的資料的查詢語法，最後會將資料插入在使用者輸入的位置，資料庫層使用 MySQL 5.6.35 如表 10。

SQL：

```
SELECT `id`, `title`, `content` FROM `news` WHERE id = 使用者輸入
```

場景 B 與場景 A 指差別在場景 B 在程式碼層會做一個空白符號的過濾機制如表 10，而場景 C 與場景 B 差別在場景 C 在 WAF 層加上了 360 CDN 進行網站的保護如表 10。

表 10. 實際測試環境

層 場景環境	WAF	Server & Backend	Code	SQL	Database
A 場景	無	Apache 2.4.26 & PHP 5.6.30	無	相同	MySQL 5.6.35
B 場景	無		空白 過濾		
C 場景	360 CDN		空白 過濾		

SQLMap 是一套專門用來做資料隱碼檢測及注入的工具，因此我們拿來與本研究進行比較，為了比較的公平性 SQLMap 在做測試時會多發送一些探測封包導致整體封包數較多，因此比較時會手動將其去除。檢測結果在 A 場景中 SQLMap 花了六次請求發現了弱點，本研究花了 4 到 6 個請求發現弱點，會有次數不固定的原因是決策樹在選擇節點時，擁有相同基尼係數的 SQL 語句會隨機挑選，導致會有結果不同的情況。在 B 場景和 C 場景中 SQLMap 在沒有人為介入調整參數的情況下都無法檢測出弱點，本研究在場景 C 中在有 WAF 的保護情況下還是可以正常的發現弱點。

表 11. 效果比較表

工具 \ 場景	A 場景	B 場景	C 場景
SQLMap	6 次	無檢測出弱點	無檢測出弱點
本研究	4 - 6 次	4 - 6 次	10 - 74 次

第五章、結論與未來研究方向

本研究所提出之方法在有網頁應用程式防火牆或入侵偵測系統保護的網路環境下能正常檢測出漏洞，以解決目前多數自動化檢測工具無法在複雜環境下準確識別漏洞的問題。我們利用機器學習以跳脫傳統使用預先設計好的清單進行掃描的方式，提高了軟體自動測試的準確度，且讓系統具有容易擴充及自我學習能力。只要不斷的使用更多場景環境訓練，就可以大大提升整體系統檢測準確度。甚至也可以預測目前所遇到的環境背後是什麼網路架構、設備廠牌、程式語言、資料庫軟體等。雖然目前所建置的模擬環境還不夠全面性，但我們的檢測系統已可達到初步的檢測水平。

未來可以結合不同的機器學習模型來提高整體系統準確度，也可以將本研究設計之方法應用在檢測不同的問題上，以解決其他弱點檢測系統所面臨到的問題。

參考文獻

- [1] OWASP Top Ten Project. Available:
https://www.owasp.org/index.php/Main_Page
- [2] M. Balabanovic and Y. Shoham. Fab: Content based, collaborative recommendation. *Communications of the Association for Computing Machinery*, 40(3):66-72, 1997.
- [3] Robin Burke. 2002. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction* 12, 4 (November 2002), 331-370.
- [4] R. B. Allen. User models: Method, theory, and practice. *International Journal of Man-Machine Studies*, 32:511-543, 1990.
- [5] D. Goldberg, D. Nichols, B. Oki, and D. Terry. Using collaborative filtering to weave an information tapestry. *Communications of the Association for Computing Machinery*, 35(12):61-70, 1992.
- [6] Breese, J. S. Heckerman, D. & Kadie, C. 1998, Empirical Analysis of Predictive Algorithms for Collaborative Filtering, In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, 43-52.
- [7] Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. 2001. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web (WWW '01)*. ACM, New York, NY, USA, 285-295.
- [8] George Karypis. 2001. Evaluation of Item-Based Top-N

Recommendation Algorithms. In Proceedings of the tenth international conference on Information and knowledge management (CIKM '01), Henrique Paques, Ling Liu, and David Grossman (Eds.). ACM, New York, NY, USA, 247-254.

- [9] P.-N. Tan, M. Steinbach & V. Kumar, Introduction to Data Mining, Addison-Wesley (2005), ISBN 0-321-32136-7, chapter 8; page 500.
- [10] T. Menzies, Y. Hu, Data Mining For Very Busy People. IEEE Computer, October 2003, pgs. 18-25.
- [11] Huang, Z. Chen, H. & Zeng, D. 2004, Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering, ACM Transactions on Information Systems, 22(1), 116-142.

