การพัฒนาระบบตรวจจับภัยคุกคามในระบบเครือข่ายโดยใช้เทคโนโลยีปัญญาประดิษฐ์

นายสีสุก สายเวหา

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์ข้อมูลเพื่อนวัตกรรม
บัณฑิตวิทยาลัย มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ
ปีการศึกษา 2567

การพัฒนาระบบตรวจจับภัยคุกคามในระบบเครือข่ายโดยใช้เทคโนโลยีปัญญาประดิษฐ์

นายสีสุก สายเวหา

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาหลักสูตร
วิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาศาสตร์ข้อมูลเพื่อนวัตกรรม
บัณฑิตวิทยาลัย มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ
ปีการศึกษา 2567

ใบรับรองวิทยานิพนธ์

บัณฑิตวิทยาลัย มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

เรื่อง การพัฒนาระบบตรวจจับภัยคุกคามในระบบเครือข่ายโดยใช้เทคโนโลยีปัญญาประดิษฐ์

โดย นายสีสุก สายเวหา

ได้รับอนุมัติให้นับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาวิทยาศาสตร์ข้อมูลเพื่อนวัตกรรม

---------- คณบดีบัณฑิตวิทยาลัย / หัวหน้าภาควิชา

(ผู้ช่วยศาสตราจารย์ ดร.สุพจน์ จันทร์วิพัฒน์)

คณะกรรมการสอบวิทยานิพนธ์

---------- ประธานกรรมการ

(ดร.ชูชาติ หฤไชยะศักดิ์)

---------- อาจารย์ที่ปรึกษา

(ผู้ช่วยศาสตราจารย์ ดร.มาลีรัตน์ มะลิแย้ม)

---------- กรรมการ

(อาจารย์ ดร.กาญจนา วิริยะพันธ์)

ชื่อ : นายสีสุก สายเวหา
ชื่อวิทยานิพนธ์ : การพัฒนาระบบตรวจจับภัยคุกคามในระบบเครือข่ายโดยใช้เทคโนโลยีปัญญาประดิษฐ์
สาขาวิชา : วิทยาศาสตร์ข้อมูลเพื่อนวัตกรรม
มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ
อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก : ผู้ช่วยศาสตราจารย์ ดร.มาลีรัตน์ มะลิแย้ม
ปีการศึกษา : 2567

บทคัดย่อ

งานวิจัยนี้มีวัตถุประสงค์เพื่อพัฒนาระบบต้นแบบสำหรับตรวจจับภัยคุกคามในเครือข่ายคอมพิวเตอร์ โดยใช้แบบจำลองโครงข่ายประสาทเทียมที่ฝึกด้วยชุดข้อมูลมาตรฐานสำหรับการตรวจจับการบุกรุกในระบบเครือข่าย และนำมาประยุกต์ใช้งานผ่านเว็บไซต์โดยไม่ต้องพึ่งพาเครื่องแม่ข่ายภายนอก ระบบสามารถประมวลผลข้อมูลจากแฟ้มจำลองและข้อมูลจราจรจริงที่ดักจับได้ พร้อมแสดงผลผ่านแผงควบคุมแบบใกล้เคียงเวลาจริง แม้ยังไม่รองรับการประมวลผลข้อมูลสด ผลการทดลองแสดงว่าแบบจำลองมีความแม่นยำร้อยละ 90.2 ซึ่งสูงกว่าแบบจำลองเวกเตอร์สนับสนุนและกลุ่มต้นไม้ตัดสินใจหลายต้น อีกทั้งสามารถจำแนกภัยคุกคามสำคัญ เช่น การโจมตีเพื่อทำให้ระบบไม่สามารถให้บริการได้ การสุ่มรหัสผ่าน การแทรกคำสั่งฐานข้อมูล และการหลอกลวงเพื่อขโมยข้อมูล ได้อย่างถูกต้องและมีประสิทธิภาพ จึงเป็นแนวทางที่มีศักยภาพในการพัฒนาระบบตรวจจับภัยคุกคามที่สามารถนำไปประยุกต์ใช้ได้จริงในอนาคต

(มีจำนวนทั้งสิ้น 82 หน้า)

คำสำคัญ : ระบบตรวจจับการบุกรุกเครือข่าย, เครือข่ายประสาทเทียม, การตรวจจับแบบเรียลไทม์,

_____ อาจารย์ที่ปรึกษาวิทยานิพนธ์หลัก

Name : Mr. Sisouk Saiveha
Thesis Title : Development of a Network Threat Detection System using Artificial Intelligence
Major Field : Data Science for Innovation
King Mongkut's University of Technology North Bangkok
Thesis Advisor : Assistant Professor Dr. Maleerat Maliyam
Academic Year : 2024

ABSTRACT

This research aims to develop a prototype system for detecting threats in computer networks using an artificial neural network (ANN) model trained with a standard intrusion detection dataset. The model is implemented on a web-based platform that allows client-side processing without relying on external servers. The system can process both simulated data files and real network traffic captured using packet analysis tools and presents results through a dashboard that simulates near real-time display, although it does not yet support live network traffic processing. Experimental results show that the ANN model achieved an accuracy of 90.2%, outperforming support vector machine (SVM) and random forest models. The system can accurately detect major threats such as denial-of-service attacks, brute-force attempts, SQL injection, and phishing. This research provides a promising direction for developing practical and efficient network threat detection systems for real-world applications in the future.

(Total 82 Pages)

Keywords: Network Intrusion Detection System, Artificial Neural Networks, Real-Time Detection, Cybersecurity.

_____ Advisor

กิตติกรรมประกาศ

วิทยานิพนธ์ฉบับนี้สำเร็จลุล่วงได้ด้วยความกรุณาและการสนับสนุนจากหลายภาคส่วนที่มีความสำคัญอย่างยิ่ง ผู้วิจัยขอกราบขอบพระคุณ ผู้ช่วยศาสตราจารย์ ดอกเตอร์ มาลีรัตน์ มะลิแย้ม อาจารย์ที่ปรึกษา ซึ่งได้ให้คำแนะนำอย่างละเอียดรอบคอบ ถ่ายทอดความรู้ทางวิชาการ รวมถึงช่วยชี้แนะแนวทางในการแก้ไขปัญหาต่าง ๆ ด้วยความเอาใจใส่และเมตตา ทำให้ผู้วิจัยสามารถดำเนินการวิจัยและจัดทำวิทยานิพนธ์ได้อย่างเป็นระบบและครบถ้วน นอกจากนี้ ผู้วิจัยขอแสดงความขอบคุณต่อคณาจารย์ทุกท่านในคณะเทคโนโลยีสารสนเทศและนวัตกรรมดิจิทัล ที่ได้มอบความรู้และประสบการณ์อันมีคุณค่า ซึ่งมีส่วนสำคัญอย่างยิ่งในการนำมาประยุกต์ใช้ในงานวิจัยฉบับนี้อย่างมีประสิทธิภาพ

ผู้วิจัยขอขอบพระคุณเจ้าหน้าที่ประจำคณะเทคโนโลยีสารสนเทศและนวัตกรรมดิจิทัลทุกท่าน ที่ได้ให้คำแนะนำ อำนวยความสะดวก และช่วยเหลือในด้านต่าง ๆ ด้วยความเต็มใจตลอดระยะเวลาการศึกษา อีกทั้งขอขอบคุณผู้เชี่ยวชาญที่ได้ให้ความกรุณาในการให้คำปรึกษา ประเมินคุณภาพของผลงาน พร้อมทั้งเสนอแนะแนวทางในการปรับปรุงเนื้อหาเพื่อให้วิทยานิพนธ์ฉบับนี้มีความสมบูรณ์ยิ่งขึ้น ซึ่งถือเป็นประโยชน์อย่างยิ่งต่อการพัฒนาผลงานวิชาการของผู้วิจัยในระยะยาว

นอกจากนี้ ผู้วิจัยขอขอบพระคุณสำนักงานความร่วมมือเพื่อการพัฒนาระหว่างประเทศ (Thailand International Cooperation Agency: TICA) ที่ให้การสนับสนุนด้านทุนการศึกษา ทรัพยากร และโอกาสในการดำเนินงานวิจัย รวมถึงการส่งเสริมความร่วมมือทางวิชาการระหว่างประเทศ ซึ่งมีบทบาทสำคัญในการเสริมสร้างศักยภาพและพัฒนาความรู้ของผู้วิจัยให้สามารถก้าวทันต่อการเปลี่ยนแปลงของเทคโนโลยีในยุคปัจจุบัน สุดท้ายนี้ ผู้วิจัยขอขอบพระคุณบิดา มารดา ครอบครัว และเพื่อน ๆ ที่อยู่เคียงข้าง คอยสนับสนุน ให้กำลังใจ และเป็นแรงผลักดันสำคัญตลอดระยะเวลาการศึกษา หากมีข้อผิดพลาดหรือบกพร่องประการใดในวิทยานิพนธ์ฉบับนี้ ผู้วิจัยขอน้อมรับไว้ด้วยความเคารพเพื่อเป็นแนวทางในการพัฒนาต่อไปในอนาคต.

สีสุก สายเวหา

สารบัญ

สารบัญตาราง

สารบัญรูปภาพ

หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในยุคเศรษฐกิจและสังคมดิจิทัลที่เทคโนโลยีสารสนเทศและการสื่อสารมีความเจริญก้าวหน้าอย่างรวดเร็ว ระบบเครือข่ายอินเทอร์เน็ตได้กลายเป็นโครงสร้างพื้นฐานที่สำคัญต่อการดำรงชีวิตและการดำเนินกิจกรรมในทุกภาคส่วนของสังคม ทั้งภาครัฐ ภาคเอกชน และภาคประชาชน ต่างนำระบบเครือข่ายมาใช้ในการจัดเก็บ ส่งต่อ และประมวลผลข้อมูลจำนวนมหาศาล ซึ่งมีความเกี่ยวข้องโดยตรงกับการทำธุรกรรมออนไลน์ การสื่อสาร การเรียนการสอน การให้บริการทางสุขภาพ และระบบสาธารณูปโภค (Kshetri, 2018) การบูรณาการของเทคโนโลยีเหล่านี้ได้เพิ่มประสิทธิภาพในการทำงานขององค์กรและอำนวยความสะดวกแก่ผู้ใช้งานในระดับบุคคลอย่างมาก แต่ในขณะเดียวกันก็เป็นการเปิดช่องทางให้เกิดความเสี่ยงด้านความมั่นคงปลอดภัยทางไซเบอร์ตามมาอย่างหลีกเลี่ยงไม่ได้

ภัยคุกคามทางไซเบอร์ในปัจจุบันมีแนวโน้มเพิ่มขึ้นทั้งในด้านปริมาณและความซับซ้อน โดยเฉพาะเมื่อการโจมตีได้รับการพัฒนาให้สามารถหลบเลี่ยงการตรวจจับของระบบแบบเดิมที่พึ่งพาการจับคู่ลายเซ็น (Signature-Based Detection) ซึ่งอาศัยฐานข้อมูลของรูปแบบการโจมตีที่รู้จักแล้ว การโจมตีที่ไม่มีลายเซ็น หรือมีลักษณะเฉพาะเจาะจงที่แตกต่างจากรูปแบบเดิมจึงสามารถหลุดรอดการตรวจจับได้ง่าย เช่น การโจมตีแบบ Zero-Day ที่เกิดจากช่องโหว่ใหม่ที่ยังไม่มีการแก้ไข หรือการใช้มัลแวร์ที่มีลักษณะพลวัต (Polymorphic Malware) ทำให้ตรวจจับได้ยากยิ่งขึ้น (Samarati et al., 2020; Garcia-Teodoro et al., 2009) นอกจากนี้ ยังมีภัยคุกคามในลักษณะเชิงพฤติกรรม เช่น การเข้าสู่ระบบหลายครั้งอย่างผิดปกติ หรือการรับส่งข้อมูลจำนวนมากภายในเวลาสั้น ๆ ซึ่งมักเกิดจากการพยายามเจาะระบบโดยอาศัยเทคนิค Brute Force หรือ DDoS การโจมตีลักษณะนี้ไม่สามารถระบุได้จากลายเซ็น แต่ต้องอาศัยการวิเคราะห์พฤติกรรมที่ผิดปกติของข้อมูลจราจรในระบบเครือข่าย

เพื่อรับมือกับภัยคุกคามที่มีลักษณะซับซ้อนและเปลี่ยนแปลงตลอดเวลา นักวิจัยและผู้พัฒนาระบบจึงหันมาให้ความสนใจกับการประยุกต์ใช้เทคโนโลยีปัญญาประดิษฐ์ (Artificial Intelligence: AI) โดยเฉพาะการเรียนรู้ของเครื่อง (Machine Learning: ML) และการเรียนรู้เชิงลึก (Deep Learning: DL) ที่สามารถจำแนกความผิดปกติของข้อมูลได้โดยไม่จำเป็นต้องอาศัยลายเซ็น (Shone et al., 2018; Kim et al., 2020) โครงข่ายประสาทเทียม (Artificial Neural Networks: ANN) เป็นหนึ่งในเทคนิคที่ได้รับความนิยม เนื่องจากสามารถเรียนรู้ลักษณะที่ไม่เป็นเชิงเส้น วิเคราะห์ความสัมพันธ์ซับซ้อนระหว่างฟีเจอร์ในข้อมูลเครือข่าย และปรับปรุงความแม่นยำในการจำแนกกิจกรรมปกติกับกิจกรรมที่เป็นภัยคุกคามได้อย่างมีประสิทธิภาพ (Zhang et al., 2019) เทคโนโลยี

ANN ยังสามารถเรียนรู้จากข้อมูลในอดีตและปรับตัวตามข้อมูลใหม่ ๆ ได้ ซึ่งทำให้สามารถตรวจจับภัยคุกคามใหม่ที่ยังไม่เคยปรากฏในระบบมาก่อนได้อย่างเหมาะสม

นอกจากโมเดล ANN แล้ว ยังมีการนำไลบรารี TensorFlow.js ซึ่งเป็นเครื่องมือประมวลผล ML บน JavaScript มาใช้เพื่อให้ระบบสามารถรันแบบจำลอง ANN ได้ในฝั่งผู้ใช้งาน (Client-side) โดยไม่ต้องพึ่งพาเซิร์ฟเวอร์ภายนอก (Smilkov et al., 2019) ข้อดีของแนวทางนี้คือการลดเวลาในการประมวลผล ลดความล่าช้าจากเครือข่าย และเพิ่มความเร็วในการตอบสนอง ซึ่งเป็นคุณสมบัติสำคัญของระบบตรวจจับภัยคุกคามแบบเรียลไทม์ (Real-Time Detection) แนวทางดังกล่าวยังสอดคล้องกับแนวคิด Edge Computing ซึ่งเน้นการประมวลผลใกล้กับแหล่งกำเนิดข้อมูล เพื่อลดภาระการส่งข้อมูลกลับไปยังเซิร์ฟเวอร์กลาง และเพิ่มความปลอดภัยของข้อมูลในระดับอุปกรณ์หรือองค์กร (Alshamrani et al., 2020) ดังนั้น การพัฒนาและประยุกต์ใช้ ANN ร่วมกับ TensorFlow.js ถือเป็นแนวทางที่มีศักยภาพในการสร้างระบบตรวจจับภัยคุกคามที่ทันสมัย ยืดหยุ่น และสามารถต่อยอดไปสู่ระบบปฏิบัติการจริงได้ในอนาคต

1.2 วัตถุประสงค์ของการวิจัย

- เพื่อพัฒนาโมเดล Artificial Neural Network (ANN) สำหรับตรวจจับภัยคุกคามในระบบเครือข่ายได้อย่างแม่นยำ และมีประสิทธิภาพ
- เพื่อพัฒนาเว็บแอปพลิเคชันสำหรับตรวจจับภัยคุกคามในระบบเครือข่าย โดยใช้งานโมเดลการเรียนรู้ของเครื่องที่พัฒนาขึ้น

ปัจจุบันผู้วิจัยได้พัฒนาโมเดล Artificial Neural Network (ANN) พร้อมสร้างระบบต้นแบบที่สามารถประมวลผลและแสดงผลลัพธ์ผ่านเว็บแอปพลิเคชันได้อย่างมีประสิทธิภาพ อย่างไรก็ตาม ระบบดังกล่าวยังไม่เชื่อมต่อกับข้อมูลเครือข่ายแบบเรียลไทม์ เนื่องจากข้อจำกัดด้านเวลาและทรัพยากร ดังนั้น ระบบนี้จึงจัดอยู่ในระดับต้นแบบที่สามารถนำไปต่อยอดเพื่อพัฒนาการประมวลผลแบบเรียลไทม์ในอนาคต

1.3 ขอบเขตของการวิจัย

การวิจัยนี้มุ่งเน้นการพัฒนาระบบตรวจจับภัยคุกคามในระบบเครือข่ายโดยใช้เทคโนโลยี Machine Learning โดยเฉพาะ Artificial Neural Networks (ANNs) ซึ่งมีขอบเขตที่ชัดเจนดังนี้:

1.3.1 การใช้ชุดข้อมูล CICIDS 2018

ชุดข้อมูล CICIDS 2018 จะถูกนำมาใช้ในการฝึกฝนโมเดล Machine Learning ซึ่งประกอบไปด้วยข้อมูลจากการโจมตีประเภทต่างๆ เช่น Distributed Denial of Service (DDoS), Phishing, Brute Force, SQL Injection และการโจมตีประเภทอื่นๆ ที่เกิดขึ้นในเครือข่าย ชุดข้อมูลนี้เป็นชุด

ข้อมูลที่เหมาะสมสำหรับการทดสอบและพัฒนาโมเดลการตรวจจับภัยคุกคามในเครือข่าย เนื่องจากมีข้อมูลจากสถานการณ์จริงในระบบเครือข่ายที่หลากหลายและมีความสมจริง

นอกจากนี้ เพื่อให้ระบบสามารถทดสอบในสถานการณ์จริงได้มากยิ่งขึ้น ผู้วิจัยได้ทำการดักจับข้อมูลจากเครือข่ายจริงด้วยเครื่องมือ Wireshark โดยจำลองการใช้งานที่มีความเสี่ยง เช่น การเข้าสู่ระบบซ้ำๆ, การเปิดเว็บไซต์ต้องสงสัย และการส่งคำสั่ง SQL ที่ผิดปกติ จากนั้นนำข้อมูลที่ได้แปลงให้อยู่ในรูปแบบที่เหมาะสมกับโมเดลที่พัฒนาไว้ เพื่อวัดความสามารถของระบบในการตรวจจับภัยคุกคามที่ไม่เคยพบมาก่อน อย่างไรก็ตาม ข้อมูลที่ดักจับด้วย Wireshark นี้ถูกนำมาแปลงและใช้งานในรูปแบบ batch ผ่านไฟล์ CSV ในระบบต้นแบบ โดยยังไม่ได้เชื่อมต่อกับระบบดักจับข้อมูลแบบสด (real-time streaming)

1.3.2 การพัฒนาและฝึกโมเดล ANN

ในการวิจัยนี้จะมีการพัฒนาโมเดล Artificial Neural Networks (ANNs) เพื่อใช้ในการตรวจจับภัยคุกคาม โดยจะใช้ชุดข้อมูล CICIDS 2018 ในการฝึกโมเดลเพื่อเรียนรู้พฤติกรรมการโจมตีและพฤติกรรมปกติของการใช้งานเครือข่าย การพัฒนาโมเดลจะใช้ TensorFlow.js ซึ่งเป็นเครื่องมือที่สามารถใช้งานได้ในเว็บเบราว์เซอร์โดยตรง การฝึกโมเดลจะทำให้โมเดลสามารถเรียนรู้ลักษณะต่างๆ ของภัยคุกคามในเครือข่ายได้ โดยใช้เทคนิคการฝึกเชิงลึก (Deep Learning) ที่สามารถปรับตัวให้เข้ากับข้อมูลที่ซับซ้อนได้

1.3.3 การพัฒนาเว็บแอปพลิเคชันสำหรับการตรวจจับภัยคุกคาม

ในส่วนของการพัฒนาแอปพลิเคชัน ระบบที่พัฒนาสามารถทำงานบนเว็บเบราว์เซอร์ โดยใช้ React.js ในการสร้างแดชบอร์ดที่แสดงผลข้อมูลภัยคุกคามแบบโต้ตอบทันทีหลังจากผู้ใช้ส่งข้อมูลเข้ามา การพัฒนารวมถึงการสร้างระบบแจ้งเตือนภัยคุกคามและการแสดงผลข้อมูลต่างๆ ผ่านกราฟและสถิติ ซึ่งระบบยังอยู่ในระดับต้นแบบ (Prototype) และยังไม่รองรับการประมวลผลแบบ Real-Time จากเครือข่ายจริงโดยตรง

1.3.4 การเปรียบเทียบประสิทธิภาพของโมเดล

ในการวิจัยนี้จะทำการเปรียบเทียบผลลัพธ์ของโมเดล ANN ที่พัฒนาขึ้นกับโมเดล Machine Learning อื่นๆ เช่น Support Vector Machine (SVM) และ Random Forest เพื่อประเมินความสามารถในการตรวจจับภัยคุกคาม โดยใช้ตัวชี้วัดประสิทธิภาพต่างๆ เช่น Accuracy, Precision, Recall และ F1-Score เพื่อเปรียบเทียบว่าโมเดลไหนมีความแม่นยำและสามารถตรวจจับภัยคุกคามได้ดีมากที่สุด

1.3.5 ข้อจำกัดและขอบเขตทางเทคนิค

- **ขอบเขตข้อมูล**: การวิจัยนี้ใช้ชุดข้อมูล CICIDS 2018 ซึ่งประกอบด้วยรูปแบบการโจมตีที่หลากหลาย เช่น DoS, DDoS, Brute Force, Web Attack, Infiltration และ Botnet

อย่างไรก็ตาม แม้ข้อมูลจะมีความครอบคลุมระดับหนึ่ง แต่ยังไม่สามารถแทนภัยคุกคามทุกรูปแบบที่เกิดขึ้นในโลกจริงได้ทั้งหมด โดยเฉพาะภัยคุกคามที่พัฒนาอย่างรวดเร็วจากเทคโนโลยีใหม่ ๆ เช่น Zero-day Exploit หรือ Advanced Persistent Threat (APT)

- **ข้อจำกัดในทรัพยากร**: ข้อจำกัดในทรัพยากร: การฝึกโมเดล Artificial Neural Network (ANN) โดยเฉพาะเมื่อมีโครงข่ายขนาดใหญ่ หรือมีจำนวนฟีเจอร์มาก อาจต้องใช้ทรัพยากรคอมพิวเตอร์สูง รวมถึงระยะเวลาในการฝึกที่ยาวนาน ซึ่งอาจเป็นข้อจำกัดในการปรับปรุงโมเดลในเชิง Online Learning หรือ Re-training แบบต่อเนื่อง
- **การตรวจจับภัยคุกคามที่ไม่เคยพบ**: การตรวจจับภัยคุกคามที่ไม่เคยพบ: โมเดลที่ฝึกจากข้อมูลเทรนเดิมอาจมีความสามารถในการตรวจจับภัยคุกคามรูปแบบใหม่ที่ไม่เคยปรากฏในข้อมูลฝึกได้น้อย เนื่องจากโมเดลเรียนรู้จาก Pattern ที่เคยเกิดขึ้นแล้วเท่านั้น อย่างไรก็ดี ในงานวิจัยนี้ได้มีการออกแบบระบบให้สามารถรับข้อมูลที่ดักจับจากเครือข่ายจริงผ่านเครื่องมือ เช่น Wireshark เพื่อแปลงและประมวลผลด้วยโมเดลที่ฝึกไว้ ซึ่งสามารถช่วยขยายขอบเขตของการตรวจจับให้รองรับทราฟฟิกที่ไม่เคยพบมาก่อนได้ในระดับหนึ่ง

การวิจัยนี้มุ่งเน้นไปที่การพัฒนาโมเดล ANN สำหรับตรวจจับภัยคุกคามในระบบเครือข่าย โดยผสานเข้ากับ Web Application ที่แสดงผลข้อมูลการตรวจจับในรูปแบบ Dashboard และสามารถแสดงผลแบบกึ่ง Real-time ได้ ทั้งจากข้อมูล CICIDS 2018 และข้อมูลจากเครือข่ายจริงที่ถูกแปลงให้อยู่ในรูปแบบที่เหมาะสม

นอกจากนี้ ยังมีการเปรียบเทียบประสิทธิภาพของโมเดล ANN กับโมเดล SVM และ Random Forest เพื่อประเมินความแม่นยำในการตรวจจับและความสามารถในการใช้งานระบบอย่างเหมาะสมในสถานการณ์ต่าง ๆ

ขอบเขตของการวิจัยนี้จึงไม่เพียงจำกัดอยู่ที่การฝึกและทดสอบโมเดลด้วยข้อมูลเดิมเท่านั้น แต่ยังครอบคลุมถึงความสามารถของระบบในการปรับตัวต่อข้อมูลจากแหล่งเครือข่ายจริง เพื่อรองรับการประยุกต์ใช้งานในโลกความเป็นจริงได้อย่างมีประสิทธิภาพ

1.4 สมมุติฐานการวิจัย

การวิจัยนี้จะมุ่งเน้นไปที่การตรวจจับภัยคุกคามในเครือข่ายโดยใช้ Artificial Neural Networks (ANNs) และ TensorFlow.js โดยคำถามหรือสมมุติฐานการวิจัยที่ตั้งไว้มีดังนี้:

- สมมุติฐานที่ 1: การใช้ ANN ในการฝึกและตรวจจับภัยคุกคามในระบบเครือข่ายสามารถเพิ่มความแม่นยำในการตรวจจับภัยคุกคามที่มีความซับซ้อนได้ดีกว่า ระบบตรวจจับภัยคุกคามแบบเดิม

- สมมุติฐานที่ 2: การใช้ TensorFlow.js สามารถช่วยให้ระบบตรวจจับภัยคุกคามทำงานได้อย่างรวดเร็วในระบบต้นแบบบนเว็บแอปพลิเคชัน และมีศักยภาพในการพัฒนาต่อยอดสู่การทำงานแบบ Real-Time ในอนาคต

1.5 ประโยชน์ที่คาดว่าจะได้รับ

การวิจัยนี้คาดว่าจะสามารถนำเสนอผลการพัฒนา ระบบตรวจจับภัยคุกคามในเครือข่าย ที่มีประสิทธิภาพสูงและสามารถตรวจจับภัยคุกคามได้แบบโต้ตอบทันทีหลังจากได้รับข้อมูล ซึ่งมีประโยชน์ทั้งในด้าน เทคโนโลยี และ การป้องกันระบบเครือข่าย

ประโยชน์ที่คาดว่าจะได้รับ:

- การพัฒนา โมเดล ANN ที่มีความสามารถในการตรวจจับภัยคุกคามใหม่ ๆ ได้แม่นยำยิ่งขึ้น
- การใช้ TensorFlow.js ในการพัฒนาเว็บแอปพลิเคชันที่สามารถทำงานได้อย่างรวดเร็วและโต้ตอบกับข้อมูลได้ทันทีในระบบต้นแบบ
- การทดสอบการตรวจจับภัยคุกคามที่มีประสิทธิภาพในข้อมูลจำลอง

บทที่ 2
ทฤษฎีและงานวิจัยที่เกี่ยวข้อง

2.1 แนวคิดพื้นฐานด้านความมั่นคงปลอดภัยในระบบเครือข่าย

ความมั่นคงปลอดภัยในระบบเครือข่าย (Network Security) เป็นรากฐานสำคัญของระบบสารสนเทศสมัยใหม่ โดยเฉพาะในยุคที่การพึ่งพาเทคโนโลยีสารสนเทศเข้ามามีบทบาทแทบทุกมิติของชีวิต ทั้งในระดับบุคคล องค์กร ไปจนถึงระดับรัฐ ความมั่นคงปลอดภัยดังกล่าวไม่เพียงเป็นเรื่องของการป้องกันภัยคุกคามที่มีอยู่ในปัจจุบันเท่านั้น แต่ยังเกี่ยวข้องกับการวางโครงสร้างพื้นฐานเพื่อรองรับภัยคุกคามที่อาจเกิดขึ้นในอนาคต โดยต้องสร้างสมดุลระหว่างความปลอดภัยกับประสิทธิภาพในการให้บริการ ซึ่งนับเป็นความท้าทายอย่างยิ่งในยุคที่ข้อมูลถูกจัดเก็บและส่งผ่านเครือข่ายด้วยความรวดเร็วสูง

หัวใจของแนวคิดด้านความมั่นคงปลอดภัยอยู่ที่หลักการที่เรียกว่า CIA Triad ซึ่งประกอบด้วยสามองค์ประกอบ ได้แก่ ความลับ (Confidentiality) ความถูกต้อง (Integrity) และความพร้อมใช้งาน (Availability) โดย “ความลับ” คือการรับประกันว่าข้อมูลสำคัญจะไม่ถูกเข้าถึงหรือเปิดเผยโดยบุคคลที่ไม่ได้รับอนุญาต การรักษาความลับของข้อมูลมีความสำคัญในระบบเครือข่ายซึ่งข้อมูลถูกส่งผ่านหลายโหนดและสามารถถูกดักฟังหรือโจมตีแบบแฝงตัวได้ง่าย “ความถูกต้อง” คือการทำให้มั่นใจว่าข้อมูลไม่ถูกเปลี่ยนแปลงหรือดัดแปลงระหว่างการจัดเก็บหรือส่งผ่านระบบ และ “ความพร้อมใช้งาน” คือการรับรองว่าระบบและบริการจะสามารถเข้าถึงและใช้งานได้ในเวลาที่ผู้ใช้ต้องการ ปัจจัยทั้งสามนี้เป็นรากฐานในการออกแบบระบบและกำหนดมาตรการป้องกันต่าง ๆ (Stallings, 2018; Whitman & Mattord, 2021)

ภัยคุกคามที่กระทบต่อความมั่นคงปลอดภัยสามารถเกิดได้จากหลายสาเหตุ ทั้งจากผู้ไม่ประสงค์ดีจากภายนอกระบบ เช่น แฮกเกอร์ กลุ่มอาชญากรรมไซเบอร์ หรือการโจมตีแบบเจาะจงเป้าหมาย ไปจนถึงการโจมตีจากบุคคลภายในที่มีสิทธิ์เข้าถึงระบบ เช่น พนักงานที่ไม่พึงประสงค์หรือผู้ดูแลระบบที่มีพฤติกรรมประมาทเลินเล่อ (Cheng et al., 2017) ลักษณะของภัยคุกคามสามารถแบ่งออกเป็นหลายประเภท ได้แก่ การโจมตีแบบ DDoS (Distributed Denial of Service) ที่มุ่งทำให้ระบบไม่สามารถให้บริการได้ การโจมตีแบบ Phishing ซึ่งแอบอ้างเพื่อหลอกลวงให้ผู้ใช้เปิดเผยข้อมูลส่วนบุคคล การแทรกโค้ดผ่านช่องโหว่ของซอฟต์แวร์ (SQL Injection) ไปจนถึงมัลแวร์ชนิดต่าง ๆ เช่น แรนซัมแวร์ สปายแวร์ หรือไวรัสคอมพิวเตอร์ ซึ่งสามารถสร้างความเสียหายร้ายแรงให้แก่ระบบได้ (Ali et al., 2022)

ภัยคุกคามสมัยใหม่มีลักษณะที่ซับซ้อนและยากต่อการตรวจจับมากขึ้น โดยเฉพาะภัยคุกคามแบบเจาะจง (targeted attacks) ที่มักพัฒนาเทคนิคการแฝงตัวให้แนบเนียนต่อระบบเฝ้าระวังทั่วไป เช่น การใช้มัลแวร์แบบ polymorphic ซึ่งสามารถเปลี่ยนรูปลักษณ์ของตัวเองในทุกการแพร่กระจายเพื่อหลบหลีกการตรวจจับของโปรแกรมแอนติไวรัส หรือการใช้ Zero-day Exploit ที่อาศัยช่องโหว่ที่ยังไม่เคยถูกค้นพบมาก่อน เพื่อโจมตีระบบก่อนที่ผู้พัฒนาจะมีโอกาสแพตช์ซอฟต์แวร์ (Alshamrani et al., 2020)

การรับมือกับภัยคุกคามเหล่านี้จึงต้องอาศัยแนวทางทั้งเชิงรับ (Passive Defense) และเชิงรุก (Active Defense) โดยการป้องกันเชิงรับมุ่งเน้นที่การติดตั้งระบบและนโยบายที่สามารถลดความเสี่ยงเบื้องต้น เช่น การใช้ไฟร์วอลล์ (Firewall) การจัดการบัญชีผู้ใช้งานอย่างเป็นระบบ หรือการสำรองข้อมูลเป็นประจำ ส่วนการป้องกันเชิงรุกมีเป้าหมายเพื่อเฝ้าระวังภัยคุกคามอย่างต่อเนื่อง วิเคราะห์พฤติกรรม และตอบสนองอย่างรวดเร็ว ซึ่งรวมถึงการใช้ระบบตรวจจับการบุกรุก (Intrusion Detection System: IDS), ระบบป้องกันการบุกรุก (Intrusion Prevention System: IPS) และระบบวิเคราะห์เหตุการณ์ด้านความปลอดภัย (SIEM) ที่ทำงานแบบเรียลไทม์ (Scarfone & Mell, 2007; Shone et al., 2018)

ในระดับนโยบายและมาตรฐานการปฏิบัติ องค์กรต่าง ๆ ได้นำมาตรฐานสากลมาใช้เป็นแนวทางในการจัดการด้านความมั่นคงปลอดภัย ตัวอย่างสำคัญคือมาตรฐาน ISO/IEC 27001 ซึ่งเป็นมาตรฐานสากลสำหรับการจัดการความมั่นคงปลอดภัยของข้อมูล โดยกำหนดแนวทางในการวางแผน ดำเนินการ ตรวจสอบ และปรับปรุงระบบสารสนเทศให้มีความปลอดภัยอย่างยั่งยืน (ISO, 2013) นอกจากนี้ หน่วยงานของรัฐในหลายประเทศยังพัฒนาแนวทางเฉพาะ เช่น NIST Cybersecurity Framework ที่จัดทำโดย National Institute of Standards and Technology ของสหรัฐอเมริกา ซึ่งเสนอโมเดล 5 ขั้นตอน ได้แก่ Identify, Protect, Detect, Respond และ Recover เพื่อใช้วางแผนด้านไซเบอร์ซีเคียวริตีอย่างเป็นระบบ (NIST, 2018)

ในภาพรวม ความมั่นคงปลอดภัยในระบบเครือข่ายไม่ได้เป็นเพียงปัญหาเชิงเทคนิคเท่านั้น หากแต่ยังเกี่ยวข้องกับนโยบาย การออกแบบระบบ การพัฒนาซอฟต์แวร์ และพฤติกรรมของผู้ใช้งาน การสร้างระบบที่มั่นคงปลอดภัยจึงต้องอาศัยความเข้าใจในหลายมิติ และการบูรณาการความรู้จากหลายศาสตร์ ไม่ว่าจะเป็นคอมพิวเตอร์ วิศวกรรม เทคโนโลยีสารสนเทศ กฎหมาย และการจัดการความเสี่ยง เพื่อให้สามารถเผชิญกับภัยคุกคามที่เปลี่ยนแปลงอย่างรวดเร็วได้อย่างมีประสิทธิภาพและยั่งยืน

2.2 ระบบตรวจจับการบุกรุก (Intrusion Detection Systems)

ในบริบทของการรักษาความมั่นคงปลอดภัยในระบบเครือข่าย การตรวจจับภัยคุกคามเป็นแนวปฏิบัติที่มีความสำคัญลำดับต้น ๆ เพราะระบบที่ดีไม่ได้เพียงแค่ป้องกันการเข้าถึงจากบุคคลภายนอกเท่านั้น แต่ยังต้องสามารถระบุพฤติกรรมหรือกิจกรรมที่มีความผิดปกติ ซึ่งอาจสะท้อนถึงความพยายามในการบุกรุกที่ซ่อนตัวอยู่ในกระแสข้อมูลได้อย่างแม่นยำ ระบบตรวจจับการบุกรุก หรือ Intrusion Detection System (IDS) จึงถูกพัฒนาขึ้นมาเพื่อทำหน้าที่เป็น “ด่านเฝ้าระวังอัจฉริยะ” ที่สามารถสังเกต พิจารณา วิเคราะห์ และแจ้งเตือนเหตุการณ์ที่อาจเป็นอันตรายภายในระบบเครือข่ายได้อย่างทันท่วงที IDS ไม่เพียงทำหน้าที่ป้องกันการบุกรุกจากภายนอกเท่านั้น แต่ยังสามารถช่วยตรวจสอบการใช้งานที่ผิดปกติจากภายใน เช่น พฤติกรรมของพนักงานที่พยายามเข้าถึงระบบหรือข้อมูลที่ตนไม่มีสิทธิ์ ซึ่งถือเป็นภัยคุกคามที่องค์กรในยุคปัจจุบันต้องเผชิญมากขึ้นเรื่อย ๆ (Saaid, Idris, & Sidek, 2021)

การทำงานของ IDS นั้นพัฒนาและปรับเปลี่ยนไปตามยุคสมัย โดยเฉพาะอย่างยิ่งในช่วงทศวรรษที่ผ่านมา เมื่อภัยคุกคามทางไซเบอร์เริ่มมีลักษณะที่ซับซ้อนมากขึ้น เทคนิคเดิมอย่างการเปรียบเทียบกับลายเซ็น (signature-based detection) เริ่มแสดงข้อจำกัด โดยไม่สามารถตรวจจับภัยคุกคามที่ยังไม่รู้จักมาก่อน หรือภัยคุกคามที่พัฒนาให้เปลี่ยนแปลงลักษณะได้อย่างรวดเร็ว การพัฒนา IDS ยุคใหม่จึงเน้นไปที่ความสามารถในการ “เรียนรู้” และ “ปรับตัว” โดยเฉพาะผ่านการประยุกต์ใช้ Machine Learning (ML) และ Deep Learning (DL) เพื่อให้ระบบสามารถตรวจจับพฤติกรรมที่ผิดปกติได้ แม้จะไม่เคยปรากฏอยู่ในฐานข้อมูลมาก่อน (Mansour, Kumar, & Singh, 2021) IDS จึงกลายเป็นระบบเชิงรุกมากขึ้น โดยสามารถตรวจสอบทราฟฟิกเครือข่ายทั้งหมดวิเคราะห์ในระดับพฤติกรรม และแจ้งเตือนโดยอัตโนมัติหากตรวจพบความเสี่ยง

ในเชิงโครงสร้าง IDS สามารถจำแนกออกเป็น 3 ประเภทหลัก ได้แก่ Signature-Based IDS, Anomaly-Based IDS และ Hybrid IDS สำหรับ Signature-Based IDS นั้นอาศัยฐานข้อมูลของลายเซ็นภัยคุกคามที่ได้มีการบันทึกไว้ล่วงหน้า เช่น MD5 hash ของไฟล์อันตราย หรือรูปแบบการโจมตีในระดับโปรโตคอล ข้อดีของแนวทางนี้คือความแม่นยำเมื่อพบการโจมตีที่รู้จักอยู่แล้ว อย่างไรก็ตาม มันไม่สามารถตรวจจับภัยคุกคามที่ยังไม่มีลายเซ็นในระบบได้ ซึ่งเป็นข้อเสียที่สำคัญเมื่อพิจารณาถึงภัยคุกคามแบบ Zero-Day หรือภัยคุกคามใหม่ที่เกิดขึ้นจากการโจมตีที่เปลี่ยนแปลงอยู่เสมอ (Farid, Saeed, & Kabir, 2022) ในทางกลับกัน Anomaly-Based IDS จะอาศัยการเรียนรู้พฤติกรรมปกติของระบบ เช่น ปริมาณข้อมูล การใช้งานโปรโตคอล เวลาเข้าใช้งาน จากนั้นทำการแจ้งเตือนหากมีพฤติกรรมที่เบี่ยงเบนออกจากลักษณะปกติ แนวทางนี้มีศักยภาพในการตรวจจับภัยคุกคามใหม่ แต่ก็มักมีปัญหาเรื่องอัตราการแจ้งเตือนผิดพลาด (false positive) ที่สูง

Hybrid IDS ถูกพัฒนาขึ้นเพื่อรวมข้อดีของทั้งสองระบบ โดยใช้ Signature-Based ในการตรวจจับภัยคุกคามที่รู้จัก และใช้ Anomaly-Based ในการวิเคราะห์พฤติกรรมใหม่ที่อาจแสดงถึงการโจมตี เทคนิคในลักษณะไฮบริดนี้ได้รับความสนใจเพิ่มขึ้นในงานวิจัยร่วมสมัย เพราะสามารถลดข้อจำกัดของระบบแต่ละแบบได้อย่างมีนัยสำคัญ (Shone & Ngoc, 2022) โดยเฉพาะเมื่อทำงานร่วมกับเทคนิค Machine Learning ที่สามารถเรียนรู้พฤติกรรมของข้อมูลได้อย่างลึกซึ้ง ระบบแบบนี้ยังสามารถประยุกต์กับแนวคิด semi-supervised learning และ reinforcement learning เพื่อเพิ่มความสามารถในการวิเคราะห์พฤติกรรมที่เปลี่ยนแปลงอยู่เสมอ เช่น พฤติกรรมการสื่อสารของแรนซัมแวร์ หรือการหลบหลีกของมัลแวร์ในรูปแบบใหม่ ๆ

นอกจากนี้ IDS ยังสามารถจำแนกตามตำแหน่งของการตรวจสอบเป็นสองประเภท ได้แก่ Network-Based IDS (NIDS) และ Host-Based IDS (HIDS) โดย NIDS ทำหน้าที่ตรวจสอบข้อมูลที่ผ่านเครือข่ายแบบเรียลไทม์ เช่น ทราฟฟิกระหว่างเซิร์ฟเวอร์กับไคลเอนต์ ในขณะที่ HIDS จะติดตั้งอยู่บนเครื่องผู้ใช้หรือเซิร์ฟเวอร์เฉพาะ ทำหน้าที่เฝ้าระวังไฟล์ระบบ กิจกรรมในระบบปฏิบัติการ และการเปลี่ยนแปลงใน registry เป็นต้น ระบบทั้งสองประเภทมีจุดเด่นที่แตกต่างกัน โดย NIDS มีขอบเขตการเฝ้าระวังกว้างและตอบสนองต่อการโจมตีแบบภายนอกได้ดี ส่วน HIDS มีความละเอียดสูงและเหมาะกับการตรวจสอบการเปลี่ยนแปลงระดับไฟล์หรือโปรเซสภายในระบบ (Ullah & Mahmoud, 2020)

ในช่วงหลังปี 2020 มีแนวโน้มที่ชัดเจนว่าการพัฒนา IDS กำลังเคลื่อนไปสู่การใช้ Deep Learning อย่างเข้มข้น โดยเฉพาะโมเดลอย่าง Convolutional Neural Networks (CNN), Recurrent Neural Networks (RNN) และ Long Short-Term Memory (LSTM) ซึ่งสามารถเข้าใจข้อมูลที่มีโครงสร้างซับซ้อน และลักษณะลำดับเวลาได้ดีกว่าโมเดลแบบดั้งเดิม (Wang et al., 2022) ตัวอย่างเช่น การใช้ CNN กับข้อมูล network flow สามารถแปลงข้อมูลแพ็กเกตให้เป็นลักษณะภาพและจำแนกรูปแบบการโจมตีได้อย่างแม่นยำ ในขณะที่ RNN และ LSTM เหมาะกับการวิเคราะห์ลำดับคำสั่งหรือเหตุการณ์ใน log file ซึ่งมีลักษณะต่อเนื่องตามเวลา Deep Learning จึงไม่เพียงช่วยให้ระบบ IDS แม่นยำมากขึ้น แต่ยังช่วยลดภาระในการเลือกฟีเจอร์แบบ manual ด้วย

อย่างไรก็ดี แม้เทคนิคเหล่านี้จะมีศักยภาพสูง แต่ก็มีความท้าทาย เช่น ความต้องการข้อมูลฝึกที่มีขนาดใหญ่ การใช้ทรัพยากรในการประมวลผลสูง และปัญหา overfitting เมื่อข้อมูลไม่ได้ถูกจัดการอย่างเหมาะสม นอกจากนี้ยังมีความกังวลด้านความปลอดภัยของโมเดลเอง เช่น การโจมตีแบบ adversarial ที่อาจทำให้ระบบเข้าใจผิดได้ ด้วยเหตุนี้ การออกแบบ IDS สมัยใหม่จึงต้องพิจารณาทั้งด้านเทคนิค ประสิทธิภาพ การป้องกันย้อนกลับ (resilience) และความสามารถในการปรับใช้ในสภาพแวดล้อมจริง ไม่ว่าจะเป็นระบบคลาวด์ ระบบ IoT หรือแม้แต่ฝั่งผู้ใช้โดยตรง (Thamilarasu & Chawla, 2021)

โดยสรุป ระบบตรวจจับการบุกรุกถือเป็นโครงสร้างพื้นฐานที่ขาดไม่ได้ในการรักษาความมั่นคงปลอดภัยของระบบสารสนเทศในยุคปัจจุบัน โดยเฉพาะในสถานการณ์ที่ภัยคุกคามมีความซับซ้อน เปลี่ยนแปลงเร็ว และสามารถแฝงตัวอยู่ในระบบได้อย่างแนบเนียน การพัฒนา IDS ที่มีประสิทธิภาพจึงไม่สามารถพึ่งพาเพียงเทคนิคใดเทคนิคหนึ่งได้อีกต่อไป แต่ต้องผสานเทคโนโลยีการเรียนรู้ของเครื่อง การออกแบบสถาปัตยกรรมระบบอย่างเหมาะสม และการใช้งานชุดข้อมูลคุณภาพสูงที่สามารถสะท้อนลักษณะการใช้งานจริงได้อย่างครบถ้วน เพื่อให้สามารถตอบสนองต่อความท้าทายที่เปลี่ยนแปลงอย่างรวดเร็วในโลกไซเบอร์ได้อย่างมีประสิทธิภาพและยั่งยืน

2.3 การเรียนรู้ของเครื่องในงานด้านความปลอดภัยเครือข่าย

การเรียนรู้ของเครื่อง (Machine Learning: ML) ได้กลายเป็นหัวใจสำคัญของนวัตกรรมด้านความมั่นคงปลอดภัยในระบบเครือข่ายในยุคปัจจุบัน โดยเฉพาะเมื่อภัยคุกคามมีความหลากหลาย ซับซ้อน และปรับเปลี่ยนได้ตลอดเวลา แนวคิดพื้นฐานของ ML คือการพัฒนาโมเดลหรืออัลกอริธึมที่สามารถเรียนรู้จากข้อมูล และนำความรู้ที่ได้ไปใช้ตัดสินใจหรือทำนายพฤติกรรมใหม่ ๆ ได้โดยไม่ต้องมีการเขียนโปรแกรมแบบตายตัว ในบริบทของความปลอดภัยเครือข่าย ML มีศักยภาพสูงในการวิเคราะห์พฤติกรรมของทราฟฟิก ตรวจจับความผิดปกติ และจำแนกรูปแบบของการโจมตีแบบเรียลไทม์ โดยไม่ต้องอาศัยฐานข้อมูลลายเซ็นของภัยคุกคามเพียงอย่างเดียวเหมือนระบบแบบเดิม (Nguyen et al., 2021)

การประยุกต์ใช้ ML ในงานด้านนี้สามารถแบ่งออกได้เป็นหลายแนวทางตามประเภทของการเรียนรู้ ได้แก่ การเรียนรู้แบบมีผู้สอน (Supervised Learning), การเรียนรู้แบบไม่มีผู้สอน (Unsupervised Learning) และการเรียนรู้แบบเสริมแรง (Reinforcement Learning) โดยในงานด้านการตรวจจับการบุกรุกและความผิดปกติ มักนิยมใช้การเรียนรู้แบบมีผู้สอน เนื่องจากสามารถฝึกโมเดลให้รู้จักรูปแบบของพฤติกรรมปกติและผิดปกติจากชุดข้อมูลที่มีการติดป้ายกำกับไว้ล่วงหน้า เช่น ประเภทของการโจมตี (e.g., DoS, Port Scan, Brute Force) หรือคลาสของข้อมูล (e.g., benign vs. malicious) โมเดลที่ได้รับความนิยมในกลุ่มนี้ ได้แก่ Decision Tree, Random Forest, Support Vector Machine (SVM), k-Nearest Neighbors (KNN), Logistic Regression และ Naive Bayes (Diro & Chilamkurti, 2021)

อย่างไรก็ตาม ในสถานการณ์จริง บางครั้งข้อมูลที่มีป้ายกำกับ (labeled data) มีจำกัด หรือไม่สามารถเข้าถึงได้ง่าย ทำให้แนวทางการเรียนรู้แบบไม่มีผู้สอนได้รับความนิยมมากขึ้นเรื่อย ๆ โดยเฉพาะสำหรับการตรวจจับพฤติกรรมที่ไม่เคยเกิดขึ้นมาก่อน เช่น Zero-Day Attack หรือการโจมตีแบบ polymorphic แนวทางนี้ไม่ต้องการป้ายกำกับข้อมูลล่วงหน้า แต่จะใช้เทคนิคการจัดกลุ่ม (Clustering) หรือการลดมิติข้อมูล (Dimensionality Reduction) เพื่อระบุว่าพฤติกรรมใดเบี่ยงเบนออกจากพฤติกรรมปกติ ตัวอย่างอัลกอริธึมที่นิยมในกลุ่มนี้ ได้แก่ K-Means, DBSCAN, PCA

(Principal Component Analysis), และ Autoencoder ซึ่งสามารถจำแนกความผิดปกติจากข้อมูลจำนวนมากได้แม้ไม่มีข้อมูลเปรียบเทียบโดยตรง (Javaid et al., 2020)

อีกแนวโน้มที่กำลังเติบโตอย่างรวดเร็วคือการประยุกต์ใช้ Deep Learning ซึ่งถือเป็นแขนงหนึ่งของ ML ที่มีโครงสร้างเป็นชั้นลึก (deep architectures) เช่น โครงข่ายประสาทเทียมหลายชั้น (Deep Neural Networks: DNN), โครงข่ายคอนโวลูชัน (Convolutional Neural Networks: CNN) และโครงข่ายแบบมีความจำ (Recurrent Neural Networks: RNN, LSTM) จุดเด่นของ DL คือความสามารถในการเรียนรู้ลักษณะที่ซับซ้อนของข้อมูลโดยไม่ต้องออกแบบฟีเจอร์เอง (feature engineering) เช่น การใช้ CNN เพื่อแปลงทราฟฟิกข้อมูลให้เป็นภาพ แล้วให้โมเดลเรียนรู้แพตเทิร์นแบบอัตโนมัติ หรือการใช้ LSTM ในการตรวจสอบ log หรือ sequence ของกิจกรรมในระบบซึ่งมีลักษณะลำดับเวลา (time series) ที่ชัดเจน (Tang et al., 2020; Alom et al., 2021)

โมเดล ML และ DL เหล่านี้ต้องอาศัยการฝึกจากข้อมูลที่มีคุณภาพสูง ซึ่งนำไปสู่การใช้ชุดข้อมูลที่เป็นมาตรฐาน เช่น CICIDS2017, CICIDS2018, NSL-KDD, UNSW-NB15 โดยเฉพาะ CICIDS2018 ที่ได้รับความนิยมสูงสุดในช่วงปีหลัง เนื่องจากมีลักษณะข้อมูลที่ครอบคลุมหลากหลายประเภทของการโจมตีและจำลองสถานการณ์ที่ใกล้เคียงกับสภาพแวดล้อมจริงมากที่สุด การเลือกใช้ฟีเจอร์ เช่น จำนวนแพ็กเกตต่อวินาที, ค่า entropy ของข้อมูล, ความยาวของ session, และโปรโตคอลที่ใช้งานถือเป็นปัจจัยสำคัญที่ส่งผลต่อความแม่นยำของโมเดล (Sharafaldin, Lashkari, & Ghorbani, 2018)

ในการประเมินผลของโมเดล ML/DL นักวิจัยมักใช้ตัวชี้วัดหลัก 4 ตัว ได้แก่ Accuracy, Precision, Recall และ F1-Score โดยบางกรณีจะใช้ ROC-AUC ร่วมด้วยเพื่อดูสมรรถนะของโมเดลในบริบทที่ class imbalance เป็นปัญหา เช่น ในกรณีที่เหตุการณ์ผิดปกติมีน้อยมากเมื่อเทียบกับข้อมูลปกติ นักวิจัยบางกลุ่มยังใช้เทคนิค oversampling เช่น SMOTE (Synthetic Minority Over-sampling Technique) เพื่อเพิ่มจำนวนข้อมูลของ class ที่มีน้อยให้สมดุลกับ class อื่น ๆ และลดความลำเอียงของโมเดล (Zhou et al., 2020)

แม้ว่า Machine Learning จะมีศักยภาพสูงในการตรวจจับภัยคุกคาม แต่ก็มีความท้าทายที่สำคัญ เช่น การเปลี่ยนแปลงของพฤติกรรมผู้ใช้ การโจมตีแบบ adversarial ที่พยายามทำให้โมเดลเข้าใจผิด การจัดการกับข้อมูลที่ไม่สมดุล (imbalanced data) และปัญหาด้านความเป็นส่วนตัว (privacy) โดยเฉพาะในกรณีที่ข้อมูลมาจากแหล่งที่เกี่ยวข้องกับผู้ใช้งานโดยตรง ด้วยเหตุนี้ นักวิจัยในยุคปัจจุบันจึงเริ่มพัฒนาโมเดลที่เน้นความโปร่งใสในการตัดสินใจ (model explainability) เช่นการใช้ XAI (Explainable AI) เพื่อให้ผู้ดูแลระบบเข้าใจว่าเหตุใดโมเดลจึงตัดสินว่าเหตุการณ์หนึ่งเป็นภัยคุกคาม ซึ่งช่วยเพิ่มความน่าเชื่อถือของระบบ และสามารถนำไปใช้งานได้จริงในระดับองค์กร (Samek, Wiegand, & Müller, 2021)

กล่าวโดยสรุป การเรียนรู้ของเครื่องได้เข้ามามีบทบาทที่ขาดไม่ได้ในระบบรักษาความปลอดภัยเครือข่าย โดยช่วยให้สามารถวิเคราะห์ความเสี่ยงแบบอัตโนมัติ ตอบสนองต่อภัยคุกคามใหม่ ๆ ได้อย่างทันท่วงที และลดภาระในการตรวจสอบของผู้ดูแลระบบ การผสมผสานระหว่างข้อมูลคุณภาพสูง อัลกอริธึมที่มีประสิทธิภาพ และความสามารถในการอธิบายผลลัพธ์ของโมเดล จะเป็นปัจจัยสำคัญในการพัฒนา IDS รุ่นใหม่ที่สามารถรับมือกับภัยคุกคามในโลกไซเบอร์ที่ซับซ้อนและเปลี่ยนแปลงตลอดเวลา

2.4 โครงข่ายประสาทเทียม (Artificial Neural Networks)

โครงข่ายประสาทเทียม (Artificial Neural Networks: ANN) เป็นโมเดลพื้นฐานของการเรียนรู้เชิงลึก (Deep Learning) ที่ได้รับแรงบันดาลใจจากการทำงานของสมองมนุษย์ โดยมีหน่วยประมวลผลย่อยที่เรียกว่า “นิวรอนเทียม” ทำงานร่วมกันเป็นชั้น ๆ เพื่อรับอินพุต ประมวลผล และให้ผลลัพธ์ตามที่ระบบได้เรียนรู้ไว้ โครงข่ายเหล่านี้สามารถจับรูปแบบและความสัมพันธ์ที่ซับซ้อนในข้อมูลได้ดีเยี่ยม โดยเฉพาะอย่างยิ่งในงานที่มีข้อมูลจำนวนมากหรือมีโครงสร้างที่ไม่สามารถวิเคราะห์ได้ด้วยเทคนิคแบบดั้งเดิม เช่น การจำแนกภาพ เสียง ข้อความ และพฤติกรรมเครือข่าย (Goodfellow, Bengio, & Courville, 2016; Khan et al., 2020) ความสามารถในการเรียนรู้ลักษณะสำคัญของข้อมูลจากอินพุตโดยไม่จำเป็นต้องออกแบบฟีเจอร์เอง (automatic feature extraction) ทำให้ ANN ได้รับความนิยมอย่างกว้างขวางในงานด้านความมั่นคงปลอดภัยทางไซเบอร์

โครงสร้างพื้นฐานของ ANN ประกอบด้วยสามชั้นหลัก ได้แก่ ชั้นอินพุต (Input Layer), ชั้นแฝง (Hidden Layer) และ ชั้นเอาต์พุต (Output Layer) ซึ่งเชื่อมต่อกันด้วยน้ำหนัก (weights) และมีการปรับค่าเหล่านี้ผ่านกระบวนการฝึกโมเดล โดยมีฟังก์ชันกระตุ้น (activation function) เช่น Sigmoid, ReLU (Rectified Linear Unit), และ Tanh เป็นตัวช่วยให้โมเดลสามารถจับความไม่เป็นเชิงเส้นของข้อมูลได้อย่างมีประสิทธิภาพ ปัจจุบันฟังก์ชัน ReLU และอนุพันธ์ เช่น Leaky ReLU ได้รับความนิยมสูงสุด เพราะสามารถลดปัญหา vanishing gradient ซึ่งเป็นปัญหาหลักในการฝึกโมเดล ANN ที่มีหลายชั้น (Nwankpa et al., 2020; Agarap, 2018)

การฝึก ANN ใช้เทคนิคที่เรียกว่า Backpropagation ร่วมกับ Gradient Descent หรือตัวแปรเชิงประสิทธิภาพ เช่น Stochastic Gradient Descent (SGD), Adam หรือ RMSProp ในการปรับน้ำหนักของแต่ละนิวรอนให้เหมาะสมกับข้อมูลตัวอย่าง กระบวนการเรียนรู้ของโมเดลจะใช้ฟังก์ชันการสูญเสีย (loss function) เช่น Cross-Entropy หรือ Mean Squared Error เพื่อคำนวณความคลาดเคลื่อนระหว่างผลลัพธ์ที่โมเดลทำนายกับคำตอบจริง แล้วใช้ความคลาดเคลื่อนนั้นส่งย้อนกลับเพื่อปรับค่าต่าง ๆ การพัฒนาอัลกอริธึมที่มีประสิทธิภาพในการฝึก ANN ได้อย่างรวดเร็ว และลดปัญหา overfitting เป็นประเด็นสำคัญในงานวิจัยร่วมสมัย (Zhang et al., 2021; Kingma & Ba, 2017)

ในด้านความมั่นคงปลอดภัยทางไซเบอร์ ANN ได้ถูกประยุกต์ใช้ในการตรวจจับพฤติกรรมเครือข่ายที่ผิดปกติ เช่น การระบุภัยคุกคามประเภทต่าง ๆ จากทราฟฟิกเครือข่าย หรือการจำแนก log file ที่มีการโจมตี ด้วยคุณสมบัติของ ANN ที่สามารถเรียนรู้จากข้อมูลที่ไม่เป็นเชิงเส้น เช่น ความสัมพันธ์ระหว่างจำนวนแพ็กเกต ขนาดข้อมูล ความถี่ของการเชื่อมต่อ ฯลฯ โมเดลจึงสามารถแยกแยะพฤติกรรมปกติและผิดปกติได้อย่างมีประสิทธิภาพมากกว่าวิธีดั้งเดิม (Ugochukwu et al., 2022; Patel & Thakkar, 2021) นอกจากนี้ ANN ยังสามารถประยุกต์ใช้ในงานวิเคราะห์ log ระบบ วิเคราะห์การพยายามล็อกอินผิดปกติ หรือการโจมตีแบบ brute force ซึ่งไม่จำเป็นต้องมีลายเซ็นของภัยคุกคามล่วงหน้า

ในแง่ของการเปรียบเทียบกับโมเดลอื่น เช่น SVM หรือ Random Forest พบว่า ANN มักให้ความแม่นยำสูงกว่าเมื่อนำมาใช้กับชุดข้อมูลที่ซับซ้อนหรือไม่สมดุล (imbalanced data) โดยเฉพาะอย่างยิ่งเมื่อใช้ ANN แบบหลายชั้น (Deep ANN) หรือแบบ recurrent สำหรับข้อมูลที่มีลักษณะลำดับเวลา อย่างไรก็ตาม การฝึก ANN ต้องอาศัยทรัพยากรสูง ทั้งในแง่ของข้อมูลและพลังการประมวลผล โดยเฉพาะเมื่อโมเดลมีจำนวนพารามิเตอร์มาก นอกจากนี้ ANN ยังมีข้อจำกัดด้านความสามารถในการอธิบายผลลัพธ์ (interpretability) ซึ่งเป็นประเด็นที่นักวิจัยพยายามพัฒนาอยู่ในปัจจุบัน เช่น การนำ Explainable AI (XAI) มาใช้เพื่ออธิบายว่าทำไมโมเดลจึงตัดสินว่าเหตุการณ์หนึ่งเป็นภัยคุกคาม (Samek et al., 2021)

ในช่วงไม่กี่ปีที่ผ่านมา มีงานวิจัยจำนวนมากที่พัฒนาและประยุกต์ใช้ ANN สำหรับระบบตรวจจับการบุกรุก โดยใช้ชุดข้อมูลมาตรฐาน เช่น CICIDS2017, CICIDS2018 และ UNSW-NB15 ตัวอย่างเช่น งานของ Aslahi-Shahri et al. (2020) ได้พัฒนา ANN เพื่อจำแนกภัยคุกคามที่พบในระบบ IoT โดยใช้ฟีเจอร์เชิงพฤติกรรม เช่น ความถี่ในการเชื่อมต่อระหว่างอุปกรณ์ งานของ Khraisat et al. (2022) ได้เปรียบเทียบ ANN กับ Deep Belief Networks และพบว่า ANN มีประสิทธิภาพเหนือกว่าสำหรับการตรวจจับการโจมตีที่ซ่อนเร้น เช่น infiltration หรือ botnet traffic ทั้งนี้การเลือกฟีเจอร์ที่เหมาะสม เช่นการใช้ Mutual Information, Chi-Square หรือ Recursive Feature Elimination (RFE) มีผลอย่างมากต่อความสามารถของโมเดลในการเรียนรู้และลดเวลาในการประมวลผล (Sahu et al., 2021)

อีกแนวทางที่น่าสนใจคือการนำ ANN มาใช้บน Edge Devices โดยใช้ไลบรารีเช่น TensorFlow.js เพื่อประมวลผลโมเดลบนฝั่งผู้ใช้โดยตรง (client-side inference) ซึ่งช่วยลดเวลาแฝง (latency), เพิ่มความเป็นส่วนตัว และลดภาระการส่งข้อมูลไปยังเซิร์ฟเวอร์กลาง การทำให้ ANN สามารถทำงานได้แบบเบา (lightweight neural networks) จึงเป็นหัวข้อที่กำลังได้รับความสนใจ

เช่น MobileNet, SqueezeNet หรือโมเดลที่ตัดทอนพารามิเตอร์ลงโดยไม่ลดประสิทธิภาพ เช่น pruning และ quantization (Howard et al., 2020; Xu et al., 2021)

โดยสรุป ANN เป็นเครื่องมือสำคัญในยุคของการรักษาความปลอดภัยเชิงรุก (proactive cybersecurity) โดยเฉพาะเมื่อถูกพัฒนาในรูปแบบลึก (deep) หรือถูกออกแบบร่วมกับโมเดลอื่นในลักษณะ ensemble หรือ hybrid ANN สามารถปรับตัวกับลักษณะข้อมูลเครือข่ายที่เปลี่ยนแปลงได้ดี และสามารถนำไปใช้กับทั้งการตรวจจับภัยคุกคาม การวิเคราะห์ช่องโหว่ หรือแม้แต่การพยากรณ์แนวโน้มของการโจมตีในอนาคตได้อย่างมีประสิทธิภาพ

2.5 เทคโนโลยี TensorFlow.js และการประยุกต์ใช้งาน

TensorFlow.js คือไลบรารีโอเพนซอร์ซที่พัฒนาโดย Google ซึ่งทำหน้าที่เป็นส่วนหนึ่งของโครงการ TensorFlow โดยมีวัตถุประสงค์หลักเพื่อให้สามารถนำโมเดล Machine Learning (ML) และ Deep Learning (DL) ไปใช้งานได้บนเว็บเบราว์เซอร์และแพลตฟอร์มที่รองรับ JavaScript โดยตรง ความสามารถในการรันโมเดลบนฝั่งผู้ใช้งาน (Client-Side) ถือเป็นจุดเด่นสำคัญที่แตกต่างจาก TensorFlow เวอร์ชันดั้งเดิมที่เน้นการทำงานบนเซิร์ฟเวอร์หรือ GPU ในสภาพแวดล้อมแบบ Python โดยความสามารถของ TensorFlow.js ประกอบด้วยฟังก์ชันหลัก 3 ส่วน ได้แก่ การฝึกโมเดล (training) จากศูนย์ด้วย JavaScript, การแปลงโมเดลจาก Python มาเป็น JavaScript และการนำโมเดลไปใช้งาน (inference) บนเว็บหรืออุปกรณ์ Edge โดยไม่ต้องติดตั้งซอฟต์แวร์เพิ่มเติม (Smilkov et al., 2019; Shi et al., 2021)

หนึ่งในข้อได้เปรียบที่สำคัญของ TensorFlow.js คือความสามารถในการทำงานบนสภาพแวดล้อมที่เข้าถึงง่าย เช่น เว็บเบราว์เซอร์ ซึ่งหมายความว่าผู้ใช้งานทั่วไปสามารถเข้าถึงเทคโนโลยี ML/DL ได้โดยไม่ต้องติดตั้งแพลตฟอร์มแบบ server-side อีกทั้งยังสามารถประมวลผลข้อมูลบนเครื่องของผู้ใช้ได้โดยตรง ช่วยลดเวลาในการส่งข้อมูลไปยังระบบกลาง และเพิ่มความเป็นส่วนตัวของข้อมูลอย่างมีนัยสำคัญ โดยเฉพาะในแอปพลิเคชันที่ต้องการให้ผลลัพธ์แบบทันที (real-time) เช่น ระบบแนะนำสินค้า ระบบรู้จำใบหน้า หรือแม้แต่ระบบตรวจจับภัยคุกคามในเครือข่าย ซึ่งต้องตอบสนองอย่างรวดเร็วและแม่นยำ (Wang et al., 2020; Mo et al., 2021)

การประยุกต์ใช้งาน TensorFlow.js ในระบบรักษาความมั่นคงปลอดภัยทางไซเบอร์มีแนวโน้มเติบโตขึ้นอย่างชัดเจน โดยเฉพาะในบริบทของ Edge Computing หรือระบบที่ต้องการนำ ML ไปทำงานในสถานที่ใกล้กับแหล่งข้อมูล เช่น เราเตอร์ สมาร์ตดีไวซ์ หรือแม้แต่เบราว์เซอร์ของผู้ใช้งาน โดยไม่ต้องพึ่งพาเซิร์ฟเวอร์กลาง ระบบลักษณะนี้เหมาะอย่างยิ่งสำหรับการตรวจจับภัยคุกคามที่ต้องการตอบสนองทันที เช่น การกรองทราฟฟิกอันตราย การจำแนกพฤติกรรมที่เบี่ยงเบน หรือแม้แต่การวิเคราะห์ไฟล์ต้องสงสัยก่อนถูกอัปโหลดเข้าสู่ระบบหลัก (Uddin et al., 2022; Xu et al., 2021)

แม้ว่า TensorFlow.js จะไม่ได้รองรับประสิทธิภาพการประมวลผลสูงเท่ากับ TensorFlow บน GPU แต่ก็มีการพัฒนาอย่างต่อเนื่องให้สามารถใช้งานได้ทั้งบน WebGL และ Node.js ซึ่งช่วยให้สามารถใช้ GPU ของผู้ใช้ผ่านเบราว์เซอร์ หรือใช้บนฝั่งเซิร์ฟเวอร์ Node.js สำหรับงาน inference ที่ซับซ้อนยิ่งขึ้น การเลือกใช้ TensorFlow.js จึงเป็นการแลกเปลี่ยนระหว่างความสะดวกในการเข้าถึงและความเร็วในการประมวลผล ซึ่งเหมาะกับงานขนาดเล็ก-กลาง ที่ต้องการความยืดหยุ่นและประสิทธิภาพที่เพียงพอ โดยเฉพาะในด้านความมั่นคงปลอดภัยที่ข้อมูลส่วนใหญ่ต้องได้รับการวิเคราะห์ที่จุดกำเนิด (Zhou et al., 2020)

อีกจุดเด่นของ TensorFlow.js คือความสามารถในการโหลดโมเดลแบบไดนามิกและปรับแต่งได้ใน runtime เช่น การเปลี่ยนโมเดลที่ใช้ในขณะที่ผู้ใช้กำลังใช้งานแอปพลิเคชันอยู่ หรือการโหลดโมเดลเฉพาะกิจตามบริบทของผู้ใช้ เช่น ประวัติการใช้งานหรือตำแหน่งทางภูมิศาสตร์ ความสามารถนี้เปิดโอกาสให้พัฒนาระบบตรวจจับภัยคุกคามแบบปรับตัวได้ (adaptive IDS) ซึ่งสามารถอัปเดตโมเดลหรือใช้โมเดลเฉพาะกลุ่มได้โดยไม่ต้องหยุดระบบ (Ali et al., 2022)

ในงานวิจัยของ Ismail et al. (2022) ได้ประยุกต์ใช้ TensorFlow.js ในการพัฒนา IDS แบบเว็บเบส โดยให้ผู้ดูแลระบบสามารถอัปโหลดไฟล์ network log เพื่อวิเคราะห์ผ่านโมเดล ANN ที่ฝึกไว้ล่วงหน้า ผลการทดลองพบว่าระบบสามารถวิเคราะห์ได้รวดเร็วและแม่นยำเกิน 92% เมื่อเทียบกับระบบที่ใช้การส่งข้อมูลไปยัง backend server ทั้งนี้ยังมีการใช้ TensorFlow.js ร่วมกับเทคนิคเชิงวิศวกรรมโมเดล เช่น การลดขนาดพารามิเตอร์ (model pruning), การแปลงโมเดล (model conversion), และการทำ quantization เพื่อให้สามารถรันโมเดลที่มีขนาดเล็กแต่ยังคงประสิทธิภาพไว้ได้บนอุปกรณ์ที่มีทรัพยากรจำกัด (Howard et al., 2020; Banbury et al., 2021)

อย่างไรก็ตาม การใช้งาน TensorFlow.js ยังมีข้อจำกัดที่ควรพิจารณา เช่น ความยากในการจัดการหน่วยความจำใน JavaScript การจำกัดของ WebGL ในบางเบราว์เซอร์ รวมถึงประเด็นด้านความปลอดภัยของโมเดลเมื่อทำงานบนฝั่งผู้ใช้ (เช่น การเข้าถึงโมเดลโดยไม่ได้รับอนุญาต หรือการโจมตีแบบ reverse engineering) นักวิจัยบางกลุ่มจึงเสนอให้ใช้เทคนิคการเข้ารหัสโมเดล การปิดกั้นโมเดลบางส่วน หรือการตรวจสอบความถูกต้องของ execution เพื่อรักษาความมั่นคงปลอดภัยในระดับโมเดล (Jia et al., 2022)

โดยสรุป TensorFlow.js เป็นเทคโนโลยีที่เปิดโอกาสใหม่ในการพัฒนาแอปพลิเคชัน Machine Learning บนแพลตฟอร์มที่เข้าถึงง่ายอย่างเว็บเบราว์เซอร์ เหมาะกับงานด้านความมั่นคงปลอดภัยไซเบอร์ที่ต้องการการตอบสนองทันที ไม่พึ่งพาโครงสร้างพื้นฐานเซิร์ฟเวอร์ และต้องการปกป้องความเป็นส่วนตัวของผู้ใช้ การบูรณาการ TensorFlow.js เข้ากับระบบตรวจจับภัยคุกคาม โดยเฉพาะร่วมกับ ANN หรือ DL อื่น ๆ จึงเป็นแนวทางที่มีศักยภาพสูงสำหรับการพัฒนาระบบเฝ้าระวังที่ฉลาดและยืดหยุ่นในโลกไซเบอร์ยุคใหม่

2.6 การประมวลผลข้อมูลเครือข่าย (Network Traffic Analysis)

การประมวลผลข้อมูลเครือข่าย (Network Traffic Analysis) หมายถึงกระบวนการสังเกต วิเคราะห์ และแปลความหมายของข้อมูลที่เคลื่อนที่ผ่านระบบเครือข่ายคอมพิวเตอร์ โดยมีวัตถุประสงค์เพื่อทำความเข้าใจลักษณะการใช้งาน การตรวจจับพฤติกรรมที่ผิดปกติ หรือแม้แต่การคาดการณ์แนวโน้มการโจมตีที่อาจเกิดขึ้นในอนาคต เทคนิคนี้ถือเป็นรากฐานสำคัญในระบบรักษาความมั่นคงปลอดภัยทางไซเบอร์ โดยเฉพาะในงานด้านการตรวจจับการบุกรุก (Intrusion Detection) และการจำแนกประเภทของภัยคุกคามที่ซ่อนอยู่ในข้อมูลจำนวนมาก การประมวลผลข้อมูลเครือข่ายจำเป็นต้องจัดการกับปริมาณข้อมูลขนาดใหญ่ที่มีความซับซ้อนสูง รวมถึงความแปรผันตามเวลาและพฤติกรรมของผู้ใช้งาน ทำให้ต้องอาศัยเทคโนโลยีที่สามารถวิเคราะห์ได้อย่างมีประสิทธิภาพ เช่น Machine Learning และ Deep Learning (Maloof et al., 2021; Qazi et al., 2022)

ประเภทของข้อมูลเครือข่ายที่ใช้ในการวิเคราะห์สามารถแบ่งได้เป็น 3 ระดับหลัก ได้แก่ ระดับแพ็กเกต (Packet-Level) ซึ่งเก็บข้อมูลแบบละเอียดมาก เช่น ไบต์ของแต่ละเฟรมหรือส่วนหัวของโปรโตคอลต่าง ๆ, ระดับโฟลว์ (Flow-Level) ที่สรุปพฤติกรรมของการเชื่อมต่อในช่วงเวลาหนึ่ง เช่น การสื่อสารระหว่าง IP และพอร์ต และ ระดับเซสชัน (Session-Level) ซึ่งรวมกิจกรรมหลายรายการที่เกี่ยวข้องเข้าด้วยกันเป็นหน่วยของ “เซสชัน” ทั้งสามระดับนี้สามารถให้ข้อมูลเชิงลึกที่แตกต่างกัน ซึ่งเมื่อนำมาใช้ร่วมกับการสกัดฟีเจอร์ (Feature Extraction) ที่เหมาะสม จะช่วยให้โมเดลเรียนรู้พฤติกรรมของผู้ใช้และแยกแยะความผิดปกติได้อย่างมีประสิทธิภาพ (Sivanathan et al., 2020; Alshamrani et al., 2021)

การเตรียมข้อมูลเครือข่ายสำหรับการประยุกต์ใช้ใน Machine Learning หรือ Deep Learning ถือเป็นขั้นตอนสำคัญที่ส่งผลโดยตรงต่อประสิทธิภาพของโมเดล โดยเริ่มจากการทำความสะอาดข้อมูล (Data Cleaning) เช่น การจัดการกับข้อมูลที่ขาดหายหรือซ้ำซ้อน จากนั้นจึงเข้าสู่ขั้นตอนของการสกัดฟีเจอร์ (Feature Extraction) และการเลือกฟีเจอร์ที่เกี่ยวข้อง (Feature Selection) เช่น ขนาดแพ็กเกตเฉลี่ย, ความถี่ของการเชื่อมต่อ, โปรโตคอลที่ใช้, หรือจำนวนไบต์ที่ส่งต่อการเชื่อมต่อแต่ละครั้ง เทคนิคการแปลงข้อมูล เช่น การทำ normalization หรือ one-hot encoding ก็มีความจำเป็นเพื่อให้โมเดลสามารถเรียนรู้ข้อมูลได้อย่างมีเสถียรภาพ (Mahdavifar & Ghorbani, 2020; Berman et al., 2020)

การเลือกเครื่องมือสำหรับการเก็บข้อมูลเครือข่ายก็มีความสำคัญเช่นกัน โดยเครื่องมือที่นิยมใช้งานได้แก่ Wireshark ซึ่งสามารถดักจับและวิเคราะห์แพ็กเกตในระดับลึกได้อย่างละเอียด, NetFlow ซึ่งให้ข้อมูลในระดับโฟลว์โดยไม่ละเมิดความเป็นส่วนตัวของผู้ใช้มากนัก และ Zeek (Bro) ซึ่งสามารถสกัดข้อมูลเชิงพฤติกรรมออกมาได้อย่างมีโครงสร้าง งานวิจัยในปัจจุบันมีแนวโน้มที่จะใช้ Zeek

ร่วมกับระบบอัตโนมัติเพื่อวิเคราะห์ข้อมูลแบบเรียลไทม์ และรวมเข้ากับแพลตฟอร์ม AI/ML อย่างต่อเนื่อง เช่น TensorFlow, PyTorch หรือแม้แต่ TensorFlow.js สำหรับงานฝั่งไคลเอนต์ (Yang et al., 2022; Zolanvari et al., 2021)

ชุดข้อมูลที่ใช้ในการฝึกและทดสอบระบบวิเคราะห์เครือข่ายต้องมีคุณภาพสูง และสะท้อนสภาพแวดล้อมที่แท้จริง โดยเฉพาะอย่างยิ่งในระบบที่นำ ML/DL มาใช้งาน ชุดข้อมูลยอดนิยมที่ถูกใช้ในงานวิจัยหลังปี 2020 ได้แก่ CICIDS2018, UNSW-NB15, และ TON_IoT, ซึ่งประกอบด้วยข้อมูลการโจมตีที่หลากหลาย เช่น DoS, Port Scan, Botnet, Infiltration, และ Web Attack การใช้ชุดข้อมูลเหล่านี้ควบคู่กับเทคนิคการทำ data balancing เช่น SMOTE หรือ ADASYN ช่วยแก้ปัญหา class imbalance ที่มักพบในข้อมูลความมั่นคงปลอดภัย (Sharafaldin et al., 2018; Haider et al., 2021)

ปัจจุบันยังมีความพยายามในการประยุกต์ใช้การวิเคราะห์ข้อมูลเครือข่ายร่วมกับโมเดล Deep Learning ขั้นสูง เช่น RNN, LSTM และ Transformer เพื่อเพิ่มความสามารถในการทำความเข้าใจลำดับเหตุการณ์ หรือความสัมพันธ์ระหว่างกิจกรรมที่เกิดขึ้นในช่วงเวลาหนึ่ง ซึ่งเหมาะกับข้อมูลแบบ time-series ในเครือข่าย ตัวอย่างเช่น งานของ Abeshu & Chilamkurti (2021) ได้นำ LSTM มาใช้วิเคราะห์พฤติกรรม DNS traffic เพื่อแยกความแตกต่างระหว่างการสื่อสารปกติกับการควบคุม botnet ซึ่งแฝงอยู่ในลักษณะทราฟฟิกที่ดูเหมือนปกติ

โดยสรุป การประมวลผลข้อมูลเครือข่ายเป็นรากฐานที่สำคัญของการนำ Machine Learning มาประยุกต์ใช้ในระบบรักษาความมั่นคงปลอดภัยไซเบอร์ โดยไม่เพียงต้องเข้าใจระดับของข้อมูลและเครื่องมือที่ใช้ในการเก็บข้อมูลเท่านั้น แต่ยังต้องมีทักษะในการจัดการ เตรียม และแปลงข้อมูลอย่างถูกต้อง เพื่อให้ระบบสามารถแยกแยะภัยคุกคามออกจากพฤติกรรมปกติได้อย่างแม่นยำ ทั้งนี้แนวโน้มการผนวกรวมการวิเคราะห์เครือข่ายกับ AI ฝั่ง Edge เช่นในเบราว์เซอร์หรืออุปกรณ์ IoT ก็กำลังกลายเป็นหัวข้อวิจัยที่เติบโตอย่างรวดเร็วและมีแนวโน้มถูกใช้งานในเชิงพาณิชย์ในอนาคตอันใกล้

2.7 งานวิจัยที่เกี่ยวข้อง

ในช่วงหลายปีที่ผ่านมา มีงานวิจัยจำนวนมากที่นำแนวคิด Machine Learning และ Deep Learning มาใช้ในการตรวจจับภัยคุกคามในระบบเครือข่าย โดยเฉพาะอย่างยิ่งการใช้โครงข่ายประสาทเทียม (ANN) และอัลกอริธึมการจัดหมวดหมู่ต่าง ๆ เช่น Support Vector Machine (SVM) และ Random Forest (RF) กับชุดข้อมูลที่เป็นมาตรฐาน เช่น CICIDS2018 ซึ่งได้รับการยอมรับว่าเป็นหนึ่งในชุดข้อมูลที่สมจริงและครอบคลุมภัยคุกคามได้หลากหลายประเภท งานวิจัยเหล่านี้มีเป้าหมายในการตรวจจับการโจมตีในรูปแบบต่าง ๆ อาทิ DoS, Botnet, Brute Force, Web Attack และ Infiltration ผ่านการเรียนรู้จากข้อมูลลักษณะการใช้งานเครือข่าย (network flow features) ที่หลากหลาย เช่น ความถี่ในการเชื่อมต่อ, ขนาดของแพ็กเกต, และประเภทของโปรโตคอล

งานวิจัยของ Aksu และคณะ (2020) ได้ใช้ ANN ร่วมกับการลดมิติของข้อมูลผ่านเทคนิค PCA (Principal Component Analysis) เพื่อทำการจำแนกการโจมตีจากชุดข้อมูล CICIDS2018 โดยสามารถทำความแม่นยำได้สูงถึง 99.16% ในระดับ binary classification (normal vs attack) จุดเด่นของงานนี้อยู่ที่การปรับแต่งโครงสร้างของ ANN ให้เหมาะสมกับจำนวนฟีเจอร์หลังจากการลดมิติ และการใช้เทคนิค batch normalization เพื่อลดการ overfitting อย่างไรก็ตาม โมเดลยังมีข้อจำกัดในการตรวจจับประเภทของการโจมตีในระดับ multi-class ซึ่งความแม่นยำจะลดลงเหลือประมาณ 94% โดยเฉพาะกับกลุ่มการโจมตีแบบ infiltration ที่มีจำนวนตัวอย่างน้อยในชุดข้อมูล

ในขณะที่งานของ Uddin และคณะ (2021) ได้เปรียบเทียบประสิทธิภาพของ ANN, SVM และ RF บนชุดข้อมูล CICIDS2018 โดยใช้ฟีเจอร์ที่เลือกมาผ่านเทคนิค mutual information แล้วทำการเทรนแต่ละโมเดลในลักษณะ multi-class classification พบว่าโมเดล ANN มีความแม่นยำสูงสุดที่ 98.7% ตามด้วย RF ที่ 97.4% และ SVM ที่ 94.9% โดยโมเดล SVM แสดงผลการจำแนกที่ไม่ดีในคลาสย่อยที่มีข้อมูลน้อย เช่น SQL Injection หรือ Web Attack ซึ่งสอดคล้องกับข้อจำกัดของ SVM ในการจัดการกับข้อมูลไม่สมดุล ส่วน RF ให้ผลลัพธ์ที่ดีใกล้เคียงกับ ANN และมีความเร็วในการฝึกโมเดลที่สูงกว่า ANN เล็กน้อย แต่ต้องใช้หน่วยความจำมากกว่าขณะ inference

อีกหนึ่งงานวิจัยที่น่าสนใจโดย Khraisat et al. (2022) ได้นำโมเดล ANN มารวมเข้ากับ Deep Belief Network (DBN) เพื่อเพิ่มศักยภาพในการเรียนรู้เชิงลึกจากฟีเจอร์ที่ซับซ้อน และนำมาใช้ตรวจจับภัยคุกคามใน CICIDS2018 แบบ binary และ multi-class ผลการทดลองแสดงให้เห็นว่า ANN+DBN มี accuracy เฉลี่ยสูงถึง 99.3% และมีค่า F1-score ที่ดีในทุกคลาส แม้ในคลาสที่มีตัวอย่างน้อย งานนี้แสดงให้เห็นถึงข้อดีของการใช้โครงสร้างโมเดลแบบลึกที่สามารถจับลักษณะเฉพาะของข้อมูลได้มากกว่า ANN ธรรมดา อย่างไรก็ตาม ข้อจำกัดคือการใช้ทรัพยากรประมวลผลสูงและความลำบากในการปรับพารามิเตอร์จำนวนมาก

ในด้านของการประมวลผลแบบ edge หรือ client-side งานของ Ismail et al. (2022) ได้ทดลองใช้ TensorFlow.js ในการนำโมเดล ANN ที่ฝึกไว้แล้วไปใช้จริงบนเว็บเบราว์เซอร์เพื่อให้ผู้ใช้สามารถวิเคราะห์ไฟล์ log หรือ network traffic ได้ทันทีโดยไม่ต้องส่งข้อมูลไปยัง server ผลลัพธ์เบื้องต้นพบว่าความแม่นยำของโมเดล ANN ที่รันผ่าน TensorFlow.js อยู่ในช่วง 92–95% และมีเวลาในการประมวลผลต่ำกว่าการส่งข้อมูลผ่าน REST API ไปยัง backend ซึ่งแสดงให้เห็นถึงศักยภาพของ ANN ในการนำไปใช้ในระบบเฝ้าระวังแบบกระจายศูนย์ (decentralized monitoring systems) อย่างไรก็ตาม โมเดลต้องถูกลดขนาด (model compression) เพื่อให้รันได้อย่างราบรื่นบนเบราว์เซอร์

นอกจาก ANN แล้ว ยังมีงานที่ทดลองใช้ RF และ SVM บนชุดข้อมูล CICIDS2018 ร่วมกับเทคนิคเพิ่มข้อมูล (data augmentation) เพื่อแก้ปัญหา class imbalance เช่นงานของ Zhang et al. (2021) ที่ใช้ SMOTE ร่วมกับ Random Forest แล้วได้ accuracy สูงถึง 98.3% ใน multi-class classification และ precision สูงในคลาส DoS, Botnet และ Brute Force อย่างไรก็ตาม ยังพบว่าคลาส Infiltration และ Heartbleed มี precision ต่ำเนื่องจากจำนวนตัวอย่างน้อยมาก งานนี้ชี้ให้เห็นว่าการเลือกเทคนิค balance ข้อมูลร่วมกับอัลกอริธึมที่ทนต่อข้อมูลไม่สมดุลสามารถเพิ่มประสิทธิภาพโดยรวมของโมเดลได้

จากการทบทวนงานวิจัยข้างต้นจะเห็นได้ว่าการใช้ ANN และโมเดลอื่น ๆ เช่น SVM, RF มีแนวโน้มประสบความสำเร็จในการตรวจจับภัยคุกคามทางเครือข่ายบนชุดข้อมูล CICIDS2018 อย่างมีประสิทธิภาพ โดย ANN มักให้ความแม่นยำสูงสุด แต่ใช้ทรัพยากรในการฝึกสูง และต้องมีการปรับพารามิเตอร์จำนวนมาก ขณะที่ RF ให้สมดุลที่ดีระหว่างความแม่นยำและความเร็ว ส่วน SVM แม้จะง่ายต่อการฝึก แต่ไม่เหมาะกับปริมาณข้อมูลขนาดใหญ่และลักษณะที่ไม่สมดุล งานวิจัยยุคใหม่เริ่มมุ่งไปที่การรวมโมเดลหลายแบบเข้าด้วยกัน (hybrid models) การนำโมเดลไป deploy ในระบบจริง เช่น บน edge device หรือในเบราว์เซอร์ และการใช้ Explainable AI เพื่อทำให้โมเดลสามารถอธิบายผลลัพธ์ได้ ซึ่งถือเป็นแนวโน้มสำคัญในยุคที่ความมั่นคงปลอดภัยต้องการความแม่นยำและความโปร่งใสควบคู่กัน

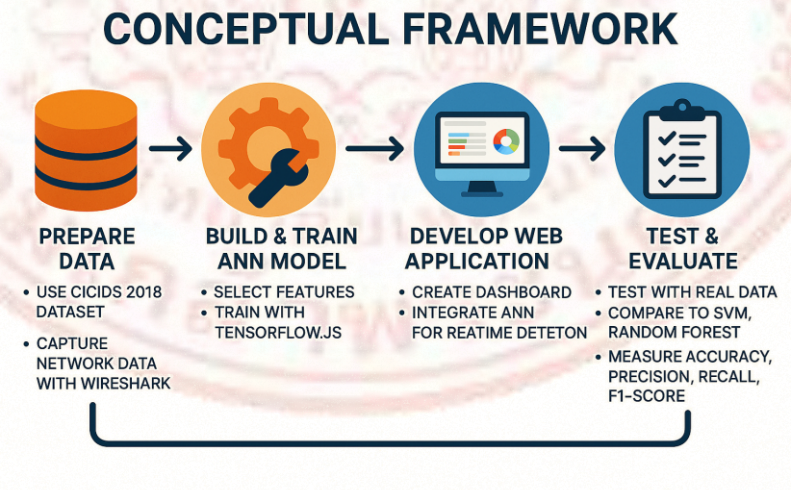
บทที่ 3
วิธีการวิจัย

3.1 กรอบแนวคิดการวิจัย

กรอบแนวคิดการวิจัยนี้มุ่งเน้นการพัฒนาระบบตรวจจับภัยคุกคามในระบบเครือข่ายโดยใช้เทคโนโลยี Artificial Neural Networks (ANNs) และการประยุกต์ใช้ TensorFlow.js เพื่อให้สามารถประมวลผลข้อมูลและแสดงผลแบบ Real-time ผ่านเว็บแอปพลิเคชัน โดยมีเป้าหมายในการปรับปรุงความแม่นยำในการตรวจจับภัยคุกคามที่มีความซับซ้อน พร้อมทั้งเพิ่มความสะดวกในการใช้งานและการตอบสนองต่อเหตุการณ์ได้อย่างรวดเร็ว

งานวิจัยนี้ใช้ ชุดข้อมูล CICIDS 2018 เป็นแหล่งข้อมูลหลักในการฝึกโมเดล ANN ซึ่งชุดข้อมูลดังกล่าวมีความครอบคลุมต่อภัยคุกคามที่หลากหลาย และเหมาะสมกับการประยุกต์ใช้ในงานด้านการตรวจจับการบุกรุก (Intrusion Detection Systems: IDS)

นอกจากนี้ เพื่อให้ระบบสามารถทำงานได้ในสถานการณ์จริง ผู้วิจัยยังได้จำลองสถานการณ์เครือข่ายโดยใช้เครื่องมือ Wireshark และ TShark ในการดักจับข้อมูลการสื่อสารจากพฤติกรรมเครือข่ายจริง เช่น การโจมตีแบบ brute force, การส่งคำสั่ง SQL ผิดปกติ และการเข้าถึงเว็บไซต์ที่มีความเสี่ยง ข้อมูลที่ได้ถูกนำมารวมกับชุดข้อมูลหลักเพื่อเพิ่มความหลากหลายและความสมจริงในการประเมินระบบ



รูปที่ 3-1 Conceptual Framework

- **การใช้ชุดข้อมูล CICIDS 2018 และการจำลองข้อมูลจริง**

- ชุดข้อมูล CICIDS 2018 เป็นชุดข้อมูลมาตรฐานที่พัฒนาโดย Canadian Institute for Cybersecurity (CIC) ซึ่งจัดเก็บข้อมูลจากสถานการณ์การใช้งานเครือข่ายที่จำลองขึ้นในห้องปฏิบัติการ โดยประกอบด้วยทั้งข้อมูลปกติและภัยคุกคามหลายประเภท เช่น DDoS, Brute Force, Phishing และ SQL Injection พร้อมทั้งมีการติดป้ายกำกับข้อมูล (labeled data) เพื่อรองรับการฝึกโมเดลแบบ Supervised Learning ได้อย่างเหมาะสม
- ผู้วิจัยยังได้จำลองพฤติกรรมในเครือข่ายจริง เช่น การล็อกอินผิดซ้ำ ๆ หรือการเข้าถึงเว็บไซต์ต้องสงสัย และทำการดักจับข้อมูลด้วยเครื่องมือ Wireshark/TShark เพื่อให้สามารถประเมินความสามารถของโมเดล ANN ได้กับข้อมูลจากโลกความเป็นจริง

- **การพัฒนาโมเดล ANN**

- โมเดล Artificial Neural Networks (ANNs) ถูกเลือกใช้เนื่องจากมีความสามารถในการเรียนรู้จากข้อมูลที่ซับซ้อน และสามารถจำแนกพฤติกรรมที่ผิดปกติได้แม้จะไม่มีลักษณะเด่นชัด
- โครงสร้างของโมเดลประกอบด้วย Input Layer ที่รับข้อมูลจำนวน 78 ฟีเจอร์, Hidden Layers จำนวน 2 ชั้นที่ใช้ฟังก์ชัน ReLU และ Output Layer ที่ใช้ Softmax สำหรับจำแนกประเภทภัยคุกคาม ได้แก่ Normal, DDoS, Phishing และอื่น ๆ
- การฝึกโมเดลใช้ TensorFlow.js เพื่อให้สามารถประมวลผลได้ทั้งบนเบราว์เซอร์และเซิร์ฟเวอร์ รองรับการทำงานแบบ Edge Computing

- **การพัฒนาเว็บแอปพลิเคชันสำหรับการตรวจจับภัยคุกคาม**

- พัฒนาเว็บแอปพลิเคชันด้วย React.js โดยมีแดชบอร์ดสำหรับแสดงผลการตรวจจับภัยคุกคามในรูปแบบกราฟและรายการแจ้งเตือน
- โมเดล ANN ที่ฝึกสำเร็จถูกนำมาใช้งานบนฝั่งไคลเอนต์ผ่าน TensorFlow.js ทำให้สามารถตรวจจับภัยคุกคามได้แบบ Real-time โดยไม่จำเป็นต้องพึ่งเซิร์ฟเวอร์ในการประมวลผล
- ระบบมีความสามารถในการแสดงจำนวนภัยคุกคาม แยกตามช่วงเวลาและประเภท พร้อมระบบแจ้งเตือนทันทีเมื่อพบพฤติกรรมที่ผิดปกติ

- **การทดสอบและประเมินผลในสภาพแวดล้อมจริง**

- ทำการทดสอบโมเดล ANN กับข้อมูลทั้งจากชุด CICIDS 2018 และข้อมูลจริงที่ได้จาก Wireshark
- เปรียบเทียบประสิทธิภาพกับโมเดลอื่น ได้แก่ Support Vector Machine (SVM) และ Random Forest (RF)

- ใช้ตัวชี้วัด ได้แก่ Accuracy, Precision, Recall และ F1-score เพื่อประเมินความแม่นยำ ความครบถ้วน และความสมดุลของการตรวจจับ
- ผลการทดสอบในสภาพแวดล้อมจริงช่วยสะท้อนศักยภาพของระบบในการนำไปใช้งานในสถานการณ์จริง และสามารถต่อยอดสู่ระบบรักษาความปลอดภัยไซเบอร์ระดับองค์กรได้ในอนาคต

กรอบแนวคิดการวิจัยนี้มุ่งเน้นการพัฒนา ระบบตรวจจับภัยคุกคาม โดยใช้ Artificial Neural Networks (ANNs) ร่วมกับ TensorFlow.js ซึ่งจะช่วยให้สามารถตรวจจับภัยคุกคามใน ระบบเครือข่าย ได้อย่างแม่นยำและทันเวลา การใช้ ชุดข้อมูล CICIDS 2018 จะช่วยให้การฝึกโมเดล ANN มีความแม่นยำ และการพัฒนา เว็บแอปพลิเคชัน จะช่วยให้ผู้ใช้สามารถตรวจสอบภัยคุกคามได้อย่างง่ายดายและรวดเร็ว การทดสอบใน Real-world Network Environment จะช่วยประเมินประสิทธิภาพของระบบและปรับปรุงให้สามารถใช้งานได้ดีขึ้น

3.2 ชุดข้อมูลที่ใช้

ในการวิจัยนี้, ชุดข้อมูล CICIDS 2018 ถูกเลือกใช้เป็นข้อมูลหลักในการฝึกฝนและทดสอบโมเดล Artificial Neural Networks (ANNs) สำหรับการตรวจจับภัยคุกคามในเครือข่าย เนื่องจากชุดข้อมูลนี้ประกอบไปด้วยการโจมตีที่หลากหลายประเภท ซึ่งสามารถจำลองสถานการณ์การโจมตีจริงที่เกิดขึ้นในระบบเครือข่ายได้อย่างดี และมีคุณสมบัติที่เหมาะสมในการนำไปใช้ในงานวิจัยด้าน Intrusion Detection Systems (IDS)

ชุดข้อมูล CICIDS 2018 ถูกพัฒนาโดย Canadian Institute for Cybersecurity (CIC) โดยชุดข้อมูลนี้ประกอบด้วยข้อมูลจากเครือข่ายที่มีการโจมตีจริงในหลายประเภท พร้อมกับการแบ่งแยกข้อมูลจากการใช้งานปกติ ซึ่งทำให้เหมาะสมสำหรับการใช้ในการฝึกฝน Machine Learning Models เช่น ANNs และการประเมินผลระบบ IDS

3.2.1 แหล่งที่มาและวัตถุประสงค์ของชุดข้อมูล

ชุดข้อมูล CICIDS 2018 ได้รับการออกแบบเพื่อจำลองสภาพแวดล้อมของระบบเครือข่ายที่ใช้งานจริง โดยมีการจำลองทั้งการใช้งานปกติ (benign traffic) และพฤติกรรมการโจมตีในหลายรูปแบบภายใต้สถานการณ์ที่หลากหลาย โดยวัตถุประสงค์หลักคือเพื่อให้สามารถใช้เป็นมาตรฐานในการฝึกสอน (training) และทดสอบ (testing) แบบจำลองตรวจจับภัยคุกคามในงานวิจัยด้านความปลอดภัยไซเบอร์ (Cybersecurity)

3.2.2 โครงสร้างและลักษณะของข้อมูล

CICIDS 2018 มีลักษณะเป็นข้อมูลแบบ Network Flow ซึ่งบันทึกพฤติกรรมการรับส่งข้อมูลในระบบเครือข่ายในระดับ packet และ flow โดยถูกแปลงให้อยู่ในรูปแบบของตารางที่สามารถนำมาใช้ในการฝึกแบบจำลอง Machine Learning ได้โดยตรง

ตารางที่ 3-1 โครงสร้างและลักษณะของข้อมูล

รายละเอียด	จำนวน
จำนวนไฟล์ต้นฉบับ	8 ไฟล์ (แบ่งตามวันและประเภทการโจมตี)
จำนวนข้อมูลทั้งหมด	มากกว่า 10 ล้าน records
จำนวนฟีเจอร์ทั้งหมด	80 ฟีเจอร์
ประเภทข้อมูล	Numerical + Categorical
รูปแบบไฟล์	CSV
ขนาดรวมทั้งหมด	มากกว่า 20 GB

ประเภทของฟีเจอร์ที่ปรากฏในชุดข้อมูล ได้แก่:

- ข้อมูลทั่วไป: Flow Duration, Protocol, Timestamp
- สถิติการส่งข้อมูล: Packet Length Mean, Flow Bytes/s
- พฤติกรรม TCP/IP: Flag Count, URG Flag, PSH Flag
- ฟีเจอร์ตามพฤติกรรมของ traffic เช่น Idle Time, Active Time
- Label: ประเภทของกิจกรรม (Benign หรือชื่อของการโจมตี)

3.2.3 โครงสร้างและลักษณะของข้อมูล

ชุดข้อมูลนี้ครอบคลุมการโจมตีที่หลากหลายมากกว่า 15 ประเภท ซึ่งสามารถจำแนกตามหมวดหมู่หลักได้ดังนี้:

- Denial of Service (DoS / DDoS): เช่น DDoS-LOIC-UDP, DDoS-HOIC, DoS-GoldenEye
- Brute Force Attack: เช่น FTP-BruteForce, SSH-BruteForce
- Web Attack: เช่น Web Attack - XSS, SQL Injection
- Infiltration & Botnet: เช่น Infiltration, Bot
- & Network Reconnaissance
- Phishing & Spam Email Traffic

3.2.4 การเลือก Subset และจัดการข้อมูล

เนื่องจากขนาดของชุดข้อมูลทั้งหมดมีปริมาณมหาศาล และเพื่อให้เหมาะสมกับการประมวลผลในระดับเครื่องคอมพิวเตอร์ส่วนบุคคล (Local Processing) ผู้วิจัยได้เลือกใช้ Subset ขนาด 250,000 records โดยเลือกข้อมูลที่มีความหลากหลายและครอบคลุมประเภทการโจมตีที่สำคัญ ได้แก่:

- Benign
- DDoS
- Brute Force
- SQL Injection
- Phishing

การสุ่มเลือกข้อมูลใช้วิธี Stratified Sampling เพื่อให้สัดส่วนของแต่ละ class ใกล้เคียงกับข้อมูลต้นฉบับ และป้องกันปัญหา class imbalance

3.2.5 การตรวจสอบและจัดเตรียมข้อมูลก่อนใช้งาน

ก่อนนำข้อมูลไปใช้ในการฝึกแบบจำลอง จำเป็นต้องมีการเตรียมข้อมูลเบื้องต้น ดังนี้:

- การลบฟีเจอร์ที่ไม่เกี่ยวข้อง: เช่น Flow ID, Source IP, Timestamp
- จัดการ Missing Values: โดยใช้ Mean Imputation สำหรับฟีเจอร์เชิงตัวเลข
- One-Hot Encoding: สำหรับข้อมูลเชิงหมวดหมู่ เช่น Protocol, Label
- การ Normalize: ใช้ Min-Max Scaling ให้ข้อมูลอยู่ในช่วง [0, 1] เพื่อให้แบบจำลองเรียนรู้ได้เร็วขึ้น
- การจัดแบ่งข้อมูล: แบ่งเป็น Training Set 80% และ Test Set 20% โดยใช้ Stratified Shuffle Split เพื่อคงสัดส่วน class

3.3 การประมวลผลและการแปลงข้อมูล

ก่อนที่จะนำข้อมูลจากชุด CICIDS 2018 ไปใช้ในการฝึกสอนแบบจำลอง Machine Learning จำเป็นต้องผ่านกระบวนการเตรียมข้อมูล (Data Preprocessing) เพื่อจัดรูปแบบให้อยู่ในสภาพที่เหมาะสมสำหรับการเรียนรู้ของเครื่อง รวมถึงลดความผิดพลาดที่อาจเกิดขึ้นจากข้อมูลที่ไม่สมบูรณ์หรือไม่เป็นมาตรฐาน การประมวลผลข้อมูลในงานวิจัยนี้สามารถแบ่งออกเป็น 5 ขั้นตอนหลัก ดังนี้:

3.3.1 การคัดเลือกฟีเจอร์ (Feature Selection)

ข้อมูลต้นฉบับจากชุด CICIDS 2018 ประกอบด้วยทั้งหมด 80 ฟีเจอร์ อย่างไรก็ตาม ไม่ใช่ทุกฟีเจอร์ที่มีความสำคัญต่อการเรียนรู้ของโมเดล หรืออาจก่อให้เกิด multicollinearity (ความสัมพันธ์ซ้ำซ้อนระหว่างฟีเจอร์) ในงานวิจัยนี้ ผู้วิจัยดำเนินการคัดเลือกฟีเจอร์โดยใช้วิธีการดังต่อไปนี้:

- ลบฟีเจอร์ที่ไม่เกี่ยวข้องกับการโจมตี: เช่น Flow ID, Timestamp, Source IP, Destination IP, Flow Byts/s ที่ไม่สามารถนำไปใช้ในการ generalize ได้
- ตรวจสอบค่าคงที่ (Constant Features): ฟีเจอร์ที่มีค่าเดียวกันตลอดทั้งคอลัมน์จะถูกตัดออก

- ใช้เทคนิคการจัดลำดับความสำคัญของฟีเจอร์ (Feature Importance): เช่น SelectKBest ร่วมกับ chi2 และ RandomForestClassifier.feature_importances_ เพื่อเลือกฟีเจอร์ที่สำคัญที่สุด 30-40 ตัว

ผลลัพธ์ของขั้นตอนนี้คือได้ชุดฟีเจอร์ที่ส่งผลต่อความแม่นยำของโมเดลสูง และช่วยลดเวลาในการประมวลผล

3.3.2 การจัดการข้อมูลสูญหาย (Missing Values)

แม้ว่าชุดข้อมูล CICIDS 2018 จะมีคุณภาพสูง แต่ยังคงมีค่าที่ว่างเปล่าหรือเป็น NaN (Not a Number) ในบางคอลัมน์ เช่น Flow Bytes/s, Flow Packets/s

การจัดการในงานวิจัยนี้มีวิธีการดังนี้:

- ข้อมูลเชิงตัวเลข: ใช้ Mean Imputation (แทนค่าที่ขาดด้วยค่าเฉลี่ยของคอลัมน์)
- ข้อมูลเชิงหมวดหมู่ (เช่น Protocol): ใช้ Most Frequent Imputation (แทนค่าที่หายด้วยค่าที่พบบ่อยที่สุด)
- ใช้คำสั่ง df.fillna() ของไลบรารี Pandas
- ทำการตรวจสอบอีกครั้งด้วย .isnull().sum() เพื่อให้แน่ใจว่าไม่มีค่า NaN เหลืออยู่ในข้อมูล

3.3.3 การเข้ารหัสข้อมูล (Data Encoding)

ฟีเจอร์บางตัว เช่น Protocol, Label มีลักษณะเป็นข้อความหรือหมวดหมู่ ซึ่งไม่สามารถป้อนเข้าสู่แบบจำลองเชิงตัวเลขได้โดยตรง

วิธีการที่ใช้:

- Label Encoding: ใช้กับ Target variable (Label) เพื่อเปลี่ยนชื่อประเภทการโจมตีเป็นค่าตัวเลข เช่น

  BENIGN ⟶ 0

  DDoS ⟶ 1

  Brute Force ⟶ 2 เป็นต้น

- One-Hot Encoding: ใช้กับฟีเจอร์ Protocol, Flag, Service ซึ่งมีค่าจำกัด และไม่ควรเปรียบเทียบเชิงลำดับ

ใช้ไลบรารี pandas.get_dummies() และ LabelEncoder จาก sklearn.preprocessing เพื่อดำเนินการ

3.3.4 การจัดการข้อมูลไม่สมดุล (Class Imbalance Handling)

ชุดข้อมูลต้นฉบับมีจำนวนข้อมูล Benign (ปกติ) มากกว่าข้อมูลประเภท Attack อย่างชัดเจน ซึ่งอาจทำให้โมเดลมี bias ต่อคลาสปกติ

เพื่อจัดการกับปัญหานี้ ผู้วิจัยใช้เทคนิค Random Undersampling ซึ่งเป็นการลดจำนวนข้อมูลใน class ที่มีมากเกิน เพื่อให้ได้สัดส่วน class ที่ใกล้เคียงกัน

- ใช้ไลบรารี imblearn.under_sampling.RandomUnderSampler
- ตั้งค่า sampling_strategy = 'not majority' เพื่อให้ Attack Classes มีจำนวนมากพอให้โมเดลเรียนรู้
- ทดสอบด้วยการเปรียบเทียบ Confusion Matrix ก่อนและหลัง undersampling

3.3.5 การปรับขนาดข้อมูล (Data Normalization)

เพื่อให้โมเดล Machine Learning สามารถเรียนรู้ได้อย่างมีประสิทธิภาพ ค่าของฟีเจอร์ทั้งหมดจำเป็นต้องอยู่ในช่วงเดียวกัน เพื่อหลีกเลี่ยงการที่ฟีเจอร์บางตัวมีค่าสเกลที่มากเกินไป (เช่น Bytes, Duration)

วิธีการที่ใช้:

- Min-Max Scaling: ทำให้ค่าทั้งหมดอยู่ในช่วง [0, 1]
- ใช้ไลบรารี MinMaxScaler จาก sklearn.preprocessing
- ดำเนินการหลังจากการเลือกฟีเจอร์และก่อนการแบ่งข้อมูลเป็น train/test

3.3.6 การแบ่งชุดข้อมูล (Data Splitting)

หลังจากได้ข้อมูลที่พร้อมใช้งานแล้ว จะต้องแบ่งข้อมูลออกเป็น 2 ส่วน:

- Training Set: 80% สำหรับฝึกแบบจำลอง
- Test Set: 20% สำหรับประเมินผล

การแบ่งข้อมูลใช้ฟังก์ชัน train_test_split() โดยตั้งค่า stratify=y เพื่อให้สัดส่วนของแต่ละ class เท่ากันในทั้งสองชุด

จากการดำเนินการทั้งหมดในขั้นตอนการประมวลผลและแปลงข้อมูลนี้ ทำให้มั่นใจได้ว่าข้อมูลมีความสะอาด ถูกต้อง และเหมาะสมต่อการฝึกสอนแบบจำลอง Machine Learning เพื่อให้สามารถตรวจจับภัยคุกคามในระบบเครือข่ายได้อย่างมีประสิทธิภาพ

3.4 การพัฒนาโมเดลตรวจจับภัยคุกคาม

ในงานวิจัยนี้ได้พัฒนาแบบจำลองการตรวจจับภัยคุกคามในระบบเครือข่ายโดยใช้เทคนิคการเรียนรู้ของเครื่อง (Machine Learning) และปัญญาประดิษฐ์ (Artificial Intelligence) โดยเปรียบเทียบผลลัพธ์ของแบบจำลองจำนวน 3 แบบ ได้แก่ Artificial Neural Network (ANN), Support Vector Machine (SVM) และ Random Forest (RF)

แบบจำลองแต่ละประเภทมีการออกแบบและฝึกสอนที่เหมาะสมกับลักษณะของข้อมูลที่ได้จากชุดข้อมูล CICIDS 2018 เพื่อเปรียบเทียบประสิทธิภาพในการตรวจจับภัยคุกคามทั้งในแง่ของความแม่นยำ ความสามารถในการจัดการข้อมูลจำนวนมาก และศักยภาพในการนำไปใช้งานจริง

3.4.1 โมเดล Artificial Neural Network (ANN)

เพื่อให้โมเดล Artificial Neural Network (ANN) มีประสิทธิภาพสูงสุดในการจำแนกภัยคุกคามในระบบเครือข่าย ผู้วิจัยได้ดำเนินการปรับแต่งพารามิเตอร์ด้วยวิธีการทดลองเปรียบเทียบหลายค่า (Manual Tuning) เนื่องจากการใช้ TensorFlow.js ในฝั่ง Client ไม่รองรับการทำ Grid Search อัตโนมัติแบบในไลบรารี Python เช่น scikit-learn การทดลองประกอบด้วย:

- จำนวนชั้นซ่อน (Hidden Layers): ทดลองใช้ 1–3 ชั้น พบว่า 2 ชั้นให้ผลลัพธ์ดีที่สุด
- จำนวนหน่วยประมวลผล (Neurons): ชั้นที่ 1 ทดลอง 64, 128, 256 หน่วย / ชั้นที่ 2 ทดลอง 32, 64, 128 หน่วย โดยค่าที่เหมาะสมคือ 128 และ 64 ตามลำดับ
- Activation Function: ทดลองใช้ relu, sigmoid, tanh ผลคือ relu ให้ประสิทธิภาพสูงสุด
- Optimizer: ทดลองใช้ sgd, rmsprop, adam โดย adam ให้ผลการเรียนรู้ดีที่สุด
- Batch Size: ทดลอง 16, 32, 64 โดย 32 ให้ผลลัพธ์ดีที่สุด
- Epochs: ทดลอง 50, 100, 150 โดย 100 รอบให้ความแม่นยำสูงและไม่เกิด overfitting
- Dropout: ทดลองที่ระดับ 0.2, 0.3, 0.4 โดยค่า 0.3 ช่วยลด overfitting ได้ดี

ดังนั้นจึงเลือกค่าที่ให้ผลลัพธ์ในการทดลองที่ดีที่สุดได้แก่:

ตารางที่ 3-2 การออกแบบโครงสร้างโมเดล ANN

ส่วนประกอบ	รายละเอียด
Input Layer	จำนวน 78 นิวรอน (เท่ากับจำนวนฟีเจอร์)
Hidden Layer 1	128 นิวรอน, Activation Function: ReLU
Hidden Layer 2	64 นิวรอน, Activation Function: ReLU
Output Layer	Softmax (จำนวน output = จำนวน class ทั้งหมด)
Optimizer	Adam
Loss Function	Categorical Crossentropy
Batch Size	32
Epochs	100
Dropout	0.3 (หลังแต่ละ hidden layer เพื่อลด overfitting)

ขั้นตอนการพัฒนา:

- นำข้อมูลที่ผ่านการประมวลผลมาแปลงเป็น Tensor ด้วย tf.tensor()
- สร้างโมเดล ANN โดยใช้ API ของ tf.sequential() และ tf.layers.dense()
- กำหนด loss, optimizer และ metric
- ฝึกโมเดลด้วยคำสั่ง model.fit() โดยใช้ Training Data และ Validation Split 20%
- บันทึกโมเดลที่ฝึกสำเร็จในรูปแบบ .json และ .bin เพื่อนำไปใช้งานบนเว็บ

3.4.2 โมเดล Support Vector Machine (SVM)

สำหรับแบบจำลอง Support Vector Machine (SVM) ผู้วิจัยได้ดำเนินการปรับแต่งพารามิเตอร์โดยใช้วิธี Grid Search ร่วมกับการทำ 5-Fold Cross-Validation ผ่านไลบรารี scikit-learn เพื่อค้นหาชุดค่าที่ให้ผลการจำแนกดีที่สุด พารามิเตอร์ที่ทดลองประกอบด้วย:

- C (Regularization Parameter): ทดลองตั้งแต่ 0.1 ถึง 1.0 เพิ่มทีละ 0.2
- Kernel Function: ทดลอง linear, poly, rbf, sigmoid พบว่า rbf ให้ผลดีที่สุด
- Gamma: ทดลอง scale และ auto โดย scale เหมาะสมกับลักษณะข้อมูล
- Multiclass Strategy: ทดลอง one-vs-one (ovo) และ one-vs-rest (ovr) ซึ่ง ovr มีความเสถียรและแม่นยำกว่าในกรณี multiclass

ดังนั้นจึงเลือกค่าที่ให้ผลลัพธ์ในการทดลองที่ดีที่สุดได้แก่:

ตารางที่ 3-3 การออกแบบโครงสร้างโมเดล SVM

พารามิเตอร์	ค่า	รายละเอียดเพิ่มเติม
Kernel	RBF	รองรับข้อมูลไม่เป็นเชิงเส้น
C	1.0	ควบคุมการเทรดออฟระหว่าง margin และ error
Gamma	‘scale’	ใช้ค่า 1 / n_features โดยอัตโนมัติ
Multi-class Strategy	One-vs-Rest	สำหรับจำแนกข้อมูลหลาย class
Scaling	StandardScaler	เพื่อปรับข้อมูลให้อยู่ในสเกลเดียวกัน
Cross-validation	5-Fold	ใช้ในขั้นตอนของ GridSearchCV

ขั้นตอนการพัฒนา:

- นำข้อมูลที่ normalize แล้วเข้าสู่กระบวนการฝึกโมเดล
- ใช้ GridSearchCV เพื่อเลือกค่าที่ดีที่สุดของ C และ gamma
- สร้างโมเดล SVM โดยใช้ SVC() จาก scikit-learn
- ฝึกโมเดลกับชุด Training Data

- ประเมินผลด้วย Confusion Matrix, Accuracy, Precision, Recall และ F1-score

3.4.3 โมเดล Random Forest (RF)

Random Forest เป็นเทคนิคการเรียนรู้แบบ Ensemble ที่ประกอบด้วยการสร้างต้นไม้หลายต้น (Decision Trees) แล้วรวมผลด้วยวิธี Voting เพื่อให้ผลลัพธ์มีเสถียรภาพและแม่นยำมากขึ้น เหมาะกับการจัดการข้อมูลที่มีความซับซ้อน และลดปัญหา overfitting

- n_estimators: ทดลอง 50, 100, 150, 200 พบว่า 100 ให้สมดุลระหว่างความแม่นยำและเวลา
- max_depth: ทดลอง 5, 10, 15 โดย 10 ให้ผลลัพธ์ดีที่สุด
- criterion: ทดลอง gini และ entropy โดย gini ให้ผลรวดเร็วและความแม่นยำใกล้เคียง
- min_samples_split: ทดลองค่าเริ่มต้น (2) และค่า 5, 10 ไม่ส่งผลชัดเจน จึงคงที่ 2
- bootstrap: ทดลองทั้ง True และ False พบว่า True ให้เสถียรที่สุด

ดังนั้นจึงเลือกค่าที่ให้ผลลัพธ์ในการทดลองที่ดีที่สุดได้แก่:

ตารางที่ 3-4 การออกแบบโครงสร้างโมเดล RF

พารามิเตอร์	ค่า	รายละเอียดเพิ่มเติม
n_estimators	100	จำนวนต้นไม้ที่ใช้ในการทำนายผล
max_depth	10	จำกัดความลึกของแต่ละต้นไม้เพื่อลด overfitting
criterion	gini	วิธีวัดความบริสุทธิ์ของข้อมูลในแต่ละ node
bootstrap	True	เปิดใช้งานการสุ่มตัวอย่างซ้ำได้
random_state	42	กำหนดค่าเพื่อให้ได้ผลลัพธ์ที่สามารถทำซ้ำได้

ขั้นตอนการพัฒนา:

- เตรียมข้อมูลด้วยการแบ่ง Train/Test เช่นเดียวกับโมเดลอื่น
- ใช้ RandomForestClassifier จากไลบรารี scikit-learn
- ตั้งค่าพารามิเตอร์เบื้องต้น และฝึกโมเดลด้วย Training Data
- ประเมินผลด้วยตัวชี้วัด Accuracy และ confusion matrix
- วิเคราะห์ฟีเจอร์สำคัญด้วย feature_importances_ เพื่อทำ Feature Selection ย้อนกลับ

3.5 การพัฒนาเว็บแอปพลิเคชัน

ในส่วนนี้อธิบายการพัฒนาเว็บแอปพลิเคชันที่ใช้สำหรับการตรวจจับภัยคุกคามในเครือข่าย โดยประยุกต์ใช้ TensorFlow.js ซึ่งช่วยให้สามารถเรียกใช้งานโมเดล Machine Learning ที่ฝึกไว้ผ่าน

เว็บเบราว์เซอร์ฝั่งผู้ใช้งาน (Client-side) ได้โดยตรง ระบบนี้ช่วยให้ผู้ใช้งานสามารถอัปโหลดข้อมูลที่ต้องการวิเคราะห์ในรูปแบบไฟล์ CSV และรับผลการประมวลผลทันทีผ่านแดชบอร์ด โดยไม่จำเป็นต้องติดตั้งซอฟต์แวร์พิเศษหรือเชื่อมต่อกับเซิร์ฟเวอร์ภายนอก

ระบบต้นแบบที่พัฒนาขึ้นมุ่งเน้นให้สามารถใช้งานได้ง่าย สะดวก และแสดงผลได้อย่างเข้าใจง่ายผ่านส่วนติดต่อผู้ใช้งาน (User Interface) ที่ถูกออกแบบด้วย React.js และเทคโนโลยี HTML/CSS มาตรฐาน

3.5.1 การพัฒนา User Interface (UI) และ Frontend

การพัฒนา Frontend มีความสำคัญอย่างมาก เนื่องจากเป็นส่วนที่ผู้ใช้จะโต้ตอบกับระบบโดยตรง โดยมีการออกแบบและพัฒนาในส่วนต่าง ๆ ดังนี้:

1) **การเลือก Framework สำหรับการพัฒนา UI**:

React.js ถูกเลือกเป็นเฟรมเวิร์กหลักในการพัฒนา UI เนื่องจากเป็นไลบรารีที่มีความยืดหยุ่นสูง มีประสิทธิภาพในการจัดการข้อมูลที่เปลี่ยนแปลงบ่อย และเหมาะสมกับการสร้างแดชบอร์ดที่ตอบสนองรวดเร็วต่อการประมวลผลข้อมูลที่อัปโหลดจากผู้ใช้

2) **การออกแบบ Dashboard**:

แดชบอร์ดถูกออกแบบให้แสดงผลการตรวจจับภัยคุกคามในรูปแบบต่าง ๆ ได้แก่:

- การแสดงประเภทของการโจมตีที่ตรวจพบ เช่น DDoS, SQL Injection, Brute Force และ Phishing
- กราฟแสดงภาพรวมของภัยคุกคามในข้อมูลที่อัปโหลด เช่น Bar Chart, Line Chart
- ตารางแสดงรายละเอียดของแต่ละรายการที่ตรวจพบ

3) การทำงานของ UI

ผู้ใช้สามารถเลือกไฟล์ CSV ที่มีข้อมูลเครือข่ายเพื่อทำการวิเคราะห์ได้ผ่านหน้าเว็บ จากนั้นระบบจะประมวลผลข้อมูลทันทีและแสดงผลลัพธ์บนแดชบอร์ดอย่างชัดเจน โดยไม่ต้องโหลดหน้าใหม่ (Single Page Application)

3.5.2 การประมวลผลด้วย TensorFlow.js

TensorFlow.js ถูกนำมาใช้ในการเรียกใช้งานโมเดล Artificial Neural Networks (ANN) ที่ฝึกไว้จากข้อมูลชุด CICIDS 2018 โดยโมเดลจะถูกโหลดขึ้นมาทำงานภายในเบราว์เซอร์ของผู้ใช้ (Client-side Inference) การประมวลผลข้อมูลทั้งหมดจึงเกิดขึ้นภายในเครื่องของผู้ใช้เอง โดยไม่ต้องพึ่งพาเซิร์ฟเวอร์หรือ backend ใด ๆ

ข้อมูลที่ผู้ใช้อัปโหลดผ่านไฟล์ CSV จะถูกอ่านและแปลงเป็น Tensor สำหรับนำเข้าสู่โมเดล ANN และผลลัพธ์ที่ได้ (เช่น ประเภทของภัยคุกคาม) จะถูกแสดงผลในแดชบอร์ดทันที

3.5.3 การจัดการและแสดงผลข้อมูล

1) การนำเข้าข้อมูล

ผู้ใช้สามารถนำเข้าข้อมูลในรูปแบบไฟล์ CSV ซึ่งประกอบด้วยฟีเจอร์ต่าง ๆ ที่ใช้ในการตรวจจับ เช่น Duration, Protocol, Packet Size เป็นต้น ข้อมูลนี้จะถูกประมวลผลโดย TensorFlow.js ทันที

2) การแสดงผล

ผลลัพธ์ที่ได้จากการทำนายของโมเดลจะแสดงผ่านแดชบอร์ด โดยแยกตามประเภทภัยคุกคาม พร้อมสถิติโดยรวม เช่น จำนวนรายการของภัยคุกคามแต่ละประเภท และแนวโน้มแบบภาพรวม

3) การจัดเก็บข้อมูล

ผลการประมวลผลสามารถบันทึกลงในฐานข้อมูล SQLite ภายในระบบ เพื่อให้สามารถเรียกดูย้อนหลังได้ โดยข้อมูลที่จัดเก็บจะประกอบด้วยประเภทภัยคุกคาม เวลาที่ตรวจพบ และค่าฟีเจอร์ที่เกี่ยวข้อง

3.5.4 การทดสอบและปรับปรุงเว็บแอปพลิเคชัน

การทดสอบระบบต้นแบบถูกดำเนินการในสถานการณ์จำลอง โดยเน้นการทดสอบการทำงานแบบ Offline และ Online จากการอัปโหลดไฟล์ CSV รวมถึงการแสดงผลที่ถูกต้องและเข้าใจง่าย

1) การทดสอบการวิเคราะห์ภัยคุกคาม

มีการทดสอบโดยใช้ข้อมูลที่จำลองจากรูปแบบการโจมตี เช่น DDoS และ Phishing เพื่อประเมินว่าโมเดลสามารถแยกแยะประเภทของภัยคุกคามได้ถูกต้อง

2) การทดสอบ UI

ทดสอบว่าแดชบอร์ดสามารถแสดงข้อมูลได้ทันทีเมื่อผู้ใช้นำเข้าข้อมูล และมีการแจ้งเตือนที่ชัดเจนเมื่อพบภัยคุกคาม

3) การทดสอบประสิทธิภาพ

ประเมินว่าเว็บแอปพลิเคชันสามารถรองรับข้อมูลในขนาดที่เหมาะสมได้โดยไม่เกิดความล่าช้าหรือกระทบต่อประสบการณ์ของผู้ใช้งาน

การพัฒนาเว็บแอปพลิเคชันสำหรับตรวจจับภัยคุกคามในงานวิจัยนี้ เป็นการพัฒนาระบบต้นแบบที่เน้นการทำงานในฝั่งผู้ใช้งาน โดยใช้ TensorFlow.js และ React.js เป็นเครื่องมือหลัก เพื่อให้สามารถแสดงผลการวิเคราะห์ภัยคุกคามได้อย่างสะดวกและรวดเร็วจากข้อมูลที่อัปโหลดเข้ามา แม้ระบบจะยังไม่ได้เชื่อมต่อกับข้อมูลเครือข่ายจริงแบบเรียลไทม์ แต่ก็ถือเป็นพื้นฐานสำคัญสำหรับการพัฒนาระบบที่สมบูรณ์ในอนาคต

3.6 การทดสอบและประเมินผล

การทดสอบและประเมินผลเป็นขั้นตอนสำคัญในการวิจัยที่ช่วยให้มั่นใจได้ว่า ระบบตรวจจับภัยคุกคาม ที่พัฒนาขึ้นนั้นสามารถทำงานได้ตามที่คาดหวังในสถานการณ์จริง การทดสอบนี้จะช่วยประเมินประสิทธิภาพของโมเดล ANN ที่ใช้ในการตรวจจับภัยคุกคามในเครือข่าย โดยใช้ชุดข้อมูล CICIDS 2018 ในการทดสอบโมเดลและการประเมินผลของระบบที่พัฒนา

การทดสอบจะครอบคลุมทั้งการทดสอบ โมเดล ANN, การประมวลผลข้อมูลใน Real-time, และ การทดสอบการทำงานของเว็บแอปพลิเคชัน รวมถึงการวัดประสิทธิภาพของโมเดล ANN ที่ถูกฝึกด้วยข้อมูลชุด CICIDS 2018 ว่าสามารถตรวจจับภัยคุกคามต่าง ๆ ได้อย่างแม่นยำและทันเวลา

3.6.1 การทดสอบในสภาพแวดล้อมจริง (Real-world Testing)

การทดสอบในสภาพแวดล้อมจริงคือการจำลองการโจมตีในเครือข่ายจริง เพื่อทดสอบประสิทธิภาพของระบบในการตรวจจับภัยคุกคามจากข้อมูลที่ได้มาจากชุดข้อมูล CICIDS 2018 โดยทำการจำลองสถานการณ์การโจมตีหลายประเภท และทดสอบว่าโมเดลสามารถตรวจจับภัยคุกคามเหล่านั้นได้หรือไม่ในเวลาเรียลไทม์:

- **ประเภทของการโจมตีที่ใช้ในการทดสอบ:**

- DDoS (Distributed Denial of Service): การโจมตีที่พยายามทำให้บริการหรือเซิร์ฟเวอร์ล่มโดยการส่งคำขอจำนวนมาก
- Phishing: การโจมตีที่หลอกลวงผู้ใช้ให้กรอกข้อมูลส่วนตัว เช่น ชื่อผู้ใช้ รหัสผ่าน ผ่านหน้าเว็บไซต์ที่ปลอม
- SQL Injection: การโจมตีโดยการแทรกคำสั่ง SQL ที่ไม่เหมาะสมเพื่อทำลายข้อมูลในฐานข้อมูล
- Brute Force: การโจมตีโดยการเดารหัสผ่านหรือพยายามเข้าสู่ระบบหลายครั้งด้วยการเดารหัสผ่าน
- Normal Traffic: การใช้งานเครือข่ายปกติ เช่น การท่องเว็บไซต์หรือการส่งอีเมล

การทดสอบการจำลองการโจมตีในสภาพแวดล้อมจริงจะช่วยให้เราเห็นว่าโมเดลสามารถตรวจจับภัยคุกคามที่เกิดขึ้นในเครือข่ายได้อย่างมีประสิทธิภาพหรือไม่ โดยจะทดสอบว่าโมเดลสามารถแยกแยะระหว่าง Normal Traffic และ Attack Traffic ได้หรือไม่

- **การทดสอบในสภาพแวดล้อมจริง**

  การจำลองเครือข่าย:

- การจำลองเครือข่ายจริงที่มีการส่งข้อมูลปกติและการโจมตี เพื่อดูว่าโมเดลสามารถตรวจจับและแยกแยะภัยคุกคามจากข้อมูลในเครือข่ายจริงได้

- การทดสอบจะช่วยให้ตรวจสอบว่า TensorFlow.js และ ANN สามารถประมวลผลข้อมูลและทำนายภัยคุกคามในเวลาเรียลไทม์ได้หรือไม่

การทดสอบในสถานการณ์จำลอง:

- เราจะทดสอบโมเดลด้วยข้อมูล CICIDS 2018 ที่ผ่านการฝึกและประเมินผลเพื่อดูว่าโมเดลสามารถตรวจจับประเภทภัยคุกคามที่ไม่เคยเห็นมาก่อนได้หรือไม่ (Generalization)

3.6.2 การประเมินผลการทำงานของโมเดล ANN

หลังจากฝึกโมเดล ANN และทำการทดสอบแล้ว, การประเมินผลการทำงานของโมเดลจะใช้ตัวชี้วัด ที่สำคัญในการวัดประสิทธิภาพของโมเดลในด้านต่าง ๆ เช่น ความแม่นยำ (Accuracy), ความสามารถในการตรวจจับ (Recall), ความถูกต้อง (Precision) และ ค่าเฉลี่ย (F1-Score) การประเมินผลในขั้นตอนนี้จะช่วยให้เราเห็นจุดแข็งและจุดที่ควรปรับปรุงของโมเดล

- **การใช้ Confusion Matrix**

- Confusion Matrix เป็นเครื่องมือที่ใช้ในการประเมินประสิทธิภาพของโมเดล โดยจะแสดงผลลัพธ์การทำนายในรูปแบบของ 4 กลุ่ม:
- True Positives (TP): จำนวนครั้งที่โมเดลทำนายว่าเป็นภัยคุกคามและถูกต้อง
- False Positives (FP): จำนวนครั้งที่โมเดลทำนายว่าเป็นภัยคุกคาม แต่จริง ๆ แล้วไม่ใช่
- True Negatives (TN): จำนวนครั้งที่โมเดลทำนายว่าไม่เป็นภัยคุกคามและถูกต้อง
- False Negatives (FN): จำนวนครั้งที่โมเดลทำนายว่าไม่เป็นภัยคุกคาม แต่จริง ๆ แล้วเป็นภัยคุกคาม

- **การคำนวณ Precision, Recall, และ F1-Score**

Precision:

- วัดความถูกต้องของการทำนายว่าเป็นภัยคุกคามเมื่อโมเดลทำนายว่าเป็นภัยคุกคาม
- คำนวณจากสูตร:

$$\text{Precision} = \frac{TP}{TP + FP}$$

Recall:

- วัดว่าโมเดลสามารถตรวจจับภัยคุกคามที่แท้จริงได้ครบถ้วนแค่ไหน
- คำนวณจากสูตร:

$$\text{Recall} = \frac{TP}{TP + FN}$$

F1-Score:

- ค่าเฉลี่ยของ **Precision** และ **Recall** ซึ่งใช้ในการวัดความสมดุลระหว่างความถูกต้องและความสามารถในการตรวจจับ
- คำนวณจากสูตร

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- **การเปรียบเทียบผลการทำงานของโมเดล ANN กับโมเดลอื่น**
- ในการทดสอบและประเมินผล, เราจะเปรียบเทียบโมเดล ANN กับโมเดลอื่น ๆ เช่น SVM (Support Vector Machine) และ Random Forest เพื่อเปรียบเทียบความแม่นยำและความสามารถในการตรวจจับภัยคุกคาม
    - การทดสอบนี้จะช่วยให้เราเห็นว่า ANN สามารถทำงานได้ดีกว่าโมเดลอื่น ๆ หรือไม่ และประสิทธิภาพของมันในการตรวจจับภัยคุกคาม

3.6.3 การทดสอบการประมวลผลข้อมูลใน Real-time

การทดสอบนี้เน้นประเมินความสามารถของระบบแดชบอร์ดที่พัฒนาให้ผู้ใช้สามารถนำเข้าข้อมูลและแสดงผลลัพธ์ได้แบบโต้ตอบทันที (interactive dashboard)
หัวข้อที่ทดสอบได้แก่:

- ความถูกต้องในการแสดงผลข้อมูลจากโมเดล ANN
- ความชัดเจนในการแสดงประเภทของการโจมตีที่ตรวจพบ
- การแจ้งเตือนเมื่อพบภัยคุกคามในข้อมูล
- การแสดงกราฟ เช่น แผนภูมิแท่ง, เส้น และตาราง

หมายเหตุ: ระบบนี้ไม่ได้เชื่อมต่อกับข้อมูลเครือข่ายแบบสด (Real-time stream) แต่ใช้วิธีอัปโหลดข้อมูลทีละชุด (batch input) เพื่อประมวลผลแบบทันทีในฝั่งผู้ใช้งาน (Client-side inference)

3.6.4 การทดสอบประสิทธิภาพของระบบ

เพื่อให้มั่นใจว่าระบบต้นแบบสามารถรองรับข้อมูลในขนาดที่เหมาะสม และยังแสดงผลได้อย่างราบรื่น การทดสอบนี้จึงประเมินทั้งด้านความเร็ว และความสามารถในการรองรับปริมาณข้อมูล
หัวข้อที่ทดสอบได้แก่:

- ความสามารถในการประมวลผลข้อมูล CSV ที่มีขนาดใหญ่ขึ้น
- ความเร็วในการประมวลผลของ TensorFlow.js ในเบราว์เซอร์
- การตอบสนองของแดชบอร์ดเมื่อมีข้อมูลจำนวนมาก

การทดสอบและประเมินผลในงานวิจัยนี้ชี้ให้เห็นว่าระบบต้นแบบที่พัฒนาโดยใช้โมเดล ANN และ TensorFlow.js มีความสามารถในการตรวจจับภัยคุกคามจากข้อมูลเครือข่ายที่ได้จากชุดข้อมูลจำลองได้อย่างมีประสิทธิภาพ แม้ระบบจะยังไม่รองรับการทำงานแบบ Real-time streaming โดยตรง แต่ก็สามารถใช้เป็นพื้นฐานสำคัญในการต่อยอดเพื่อพัฒนาระบบที่รองรับการใช้งานจริงในสภาพแวดล้อมเครือข่ายที่เปลี่ยนแปลงตลอดเวลา

3.7 การดักจับและแปลงข้อมูลจากเครือข่ายจริง

เพื่อประเมินความสามารถของระบบตรวจจับภัยคุกคามทางไซเบอร์ที่พัฒนาขึ้นในสภาวะที่ใกล้เคียงกับการใช้งานจริง ผู้วิจัยจึงได้เพิ่มขั้นตอนการทดลองโดยใช้ข้อมูลจากการใช้งานเครือข่ายจริง (Real Network Traffic) โดยใช้เครื่องมือวิเคราะห์ข้อมูลแพ็กเก็ตที่ได้รับความนิยมและเชื่อถือได้ในวงการความมั่นคงปลอดภัยไซเบอร์ ได้แก่ Wireshark และ TShark ในการดักจับข้อมูลจากอุปกรณ์จริงบนเครือข่ายจำลอง

การดักจับและแปลงข้อมูลดังกล่าวจะช่วยให้สามารถทดสอบระบบภายใต้เงื่อนไขที่แตกต่างจากชุดข้อมูล CICIDS 2018 ซึ่งเป็นข้อมูลที่มีการจัดเตรียมไว้ล่วงหน้า โดยการใช้ข้อมูลจากเครือข่ายจริงนั้นจะเป็นการทดสอบความสามารถของโมเดล Artificial Neural Network (ANN) ที่ฝึกไว้ ว่าสามารถประมวลผลข้อมูลใหม่และตรวจจับภัยคุกคามที่เกิดขึ้นจริงได้หรือไม่

ขั้นตอนการดำเนินงาน:

1) การดักจับข้อมูลเครือข่าย (Network Traffic Capture) ผู้วิจัยได้ทำการติดตั้งโปรแกรม Wireshark บนเครื่องแม่ข่าย (monitoring node) ที่อยู่ในเครือข่ายทดสอบเดียวกันกับอุปกรณ์ที่ใช้จำลองพฤติกรรมต่าง ๆ เช่น การเข้าสู่ระบบด้วยรหัสผิดซ้ำ (Brute Force), การยิง request จำนวนมาก (DDoS), การเข้าลิงก์ที่มีลักษณะ Phishing และการส่งข้อมูลที่มีลักษณะ SQL Injection ผ่านพอร์ตที่เปิดใช้จริงในระบบ Wireshark จะทำการดักจับข้อมูลในระดับแพ็กเก็ต (packet-level) และบันทึกลงเป็นไฟล์นามสกุล .pcap ซึ่งเป็นรูปแบบมาตรฐานที่ใช้ในงาน Network Forensics
2) การแปลงข้อมูลเป็นรูปแบบที่ใช้ได้กับโมเดล (Feature Conversion) เพื่อให้ระบบสามารถประมวลผลข้อมูลที่ดักจับมาได้ จำเป็นต้องแปลงไฟล์ .pcap ให้เป็น .csv ซึ่งมีเฉพาะฟีเจอร์ที่โมเดล ANN ต้องการใช้ เช่น:
   - ระยะเวลาแพ็กเก็ต (Duration)
   - ขนาดแพ็กเก็ต (Length)
   - Protocol (TCP, UDP, ICMP)
   - Source และ Destination Port

ผู้วิจัยใช้เครื่องมือ TShark ซึ่งเป็นเครื่องมือบรรทัดคำสั่งของ Wireshark เพื่อแปลงข้อมูลจาก .pcap เป็น .csv จากนั้นนำข้อมูล .csv ที่ได้ไปผ่านกระบวนการ Preprocessing ซึ่งมีรายละเอียดดังนี้:

- การ Encoding: แปลงค่าประเภท Protocol และ IP Address เป็นรหัสตัวเลขหรือหมวดหมู่
- Normalization: ปรับค่าทางตัวเลขให้อยู่ในช่วง 0–1 เพื่อให้เหมาะสมกับการทำงานของโมเดล ANN
- Feature Mapping: จัดลำดับและเลือกเฉพาะฟีเจอร์ที่ตรงกับโมเดลที่ฝึกด้วยข้อมูล CICIDS 2018

3) การนำข้อมูลเข้าสู่ระบบตรวจจับที่พัฒนาไว้หลังจากได้ข้อมูลที่อยู่ในรูปแบบที่เหมาะสมแล้ว จะนำเข้าข้อมูลผ่าน Web Application ซึ่งฝังโมเดล ANN ไว้ด้วย TensorFlow.js โดยผู้ใช้สามารถอัปโหลดไฟล์ CSV เข้าสู่ระบบได้ด้วยตนเอง และระบบจะทำการวิเคราะห์และแสดงผลภัยคุกคามผ่านแดชบอร์ดที่แสดงกราฟแยกตามประเภทการโจมตี ตารางภัยคุกคามล่าสุด และดัชนีรวม

หมายเหตุ: ระบบที่พัฒนาขึ้นยังไม่รองรับการเชื่อมต่อแบบ Real-Time กับระบบเครือข่ายโดยตรง หรือการวิเคราะห์แบบ Streaming จาก Packet ที่ไหลเข้าระบบในเวลาจริง แต่เป็นระบบต้นแบบที่ประมวลผลข้อมูลแบบ Batch ที่ผู้ใช้อัปโหลดเข้ามาเพื่อวิเคราะห์ในทันที โดยไม่ต้องเชื่อมต่อกับฐานข้อมูลหรือเซิร์ฟเวอร์ภายนอก ทำให้สามารถนำไปใช้งานแบบ Standalone ได้ ทั้งแบบ Online และ Offline

3.8 เครื่องมือและเทคโนโลยีที่ใช้

ในการพัฒนาระบบตรวจจับภัยคุกคามในเครือข่าย โดยใช้ Artificial Neural Networks (ANNs) และ TensorFlow.js สำหรับการประมวลผลผ่านเว็บแอปพลิเคชัน ได้มีการใช้เครื่องมือและเทคโนโลยีต่าง ๆ เพื่อช่วยให้การพัฒนาและทดสอบระบบต้นแบบเป็นไปอย่างมีประสิทธิภาพ โดยเครื่องมือเหล่านี้ถูกเลือกอย่างละเอียดเพื่อให้ตอบสนองความต้องการในการฝึกโมเดล Machine Learning การสร้างเว็บแอปพลิเคชันที่ทำงานบนฝั่งผู้ใช้ การจัดการข้อมูล และการแสดงผลที่รวดเร็วผ่านแดชบอร์ด

3.8.1 เครื่องมือสำหรับการพัฒนาและฝึกโมเดล Machine Learning

1) TensorFlow.js:

- TensorFlow.js เป็นไลบรารีที่พัฒนาโดย Google สำหรับการสร้างและฝึก Machine Learning Models โดยสามารถทำงานใน JavaScript ได้ ทั้งใน Node.js และ Web Browser
- ในการวิจัยนี้ TensorFlow.js ถูกใช้เพื่อโหลดและใช้งานโมเดล Artificial Neural Networks (ANN) ที่ฝึกด้วยข้อมูล CICIDS 2018 ผ่านฝั่งผู้ใช้บนเบราว์เซอร์ โดยไม่ต้องพึ่งพาเซิร์ฟเวอร์
- การใช้ TensorFlow.js ช่วยลด Latency และเพิ่มความยืดหยุ่นของระบบต้นแบบในการตอบสนองต่อข้อมูลที่ผู้ใช้อัปโหลดเข้ามา

2) Python (สำหรับการเตรียมข้อมูล):

- ใช้สำหรับการเตรียมข้อมูลจากชุดข้อมูล CICIDS 2018 และข้อมูลจากเครือข่ายจริงก่อนนำเข้าสู่โมเดล
- ไลบรารี Pandas และ NumPy ใช้ในการจัดการข้อมูล, Scikit-learn ใช้ในการ Scaling, Normalization, และ Feature Selection
- ข้อมูลถูกแปลงให้อยู่ในรูปแบบ CSV และใช้กับ TensorFlow.js ได้อย่างเหมาะสม

3) Jupyter Notebook:

- ใช้ในการพัฒนาและทดสอบโค้ด Python สำหรับการเตรียมข้อมูลและการฝึกโมเดล
- ช่วยในการแสดงผลวิเคราะห์อย่างเข้าใจง่าย และสามารถปรับแต่งโค้ดได้สะดวกในระหว่างการทดลอง

3.8.2 เครื่องมือสำหรับการพัฒนาเว็บแอปพลิเคชัน

1) React.js:

- เป็นไลบรารีสำหรับสร้าง User Interface ที่เหมาะกับระบบแบบ Dynamic
- ในงานวิจัยนี้ React.js ช่วยในการสร้างแดชบอร์ดที่แสดงผลลัพธ์การวิเคราะห์จากโมเดล ANN ได้ทันทีโดยไม่ต้องโหลดหน้าใหม่
- มีโครงสร้างที่เหมาะสมกับระบบที่สามารถพัฒนาให้รองรับ real-time ได้ในอนาคต

2) Node.js: (อยู่ในแผนการพัฒนาระบบถัดไป)

- แม้ยังไม่ได้ใช้งานในระบบปัจจุบัน แต่ Node.js ได้รับการพิจารณาสำหรับการพัฒนา backend API ที่สามารถเชื่อมต่อกับระบบดักจับข้อมูลจริงในอนาคต
- ในงานวิจัยนี้ frontend ทำงานแบบ standalone โดยไม่เชื่อมกับ backend หรือเซิร์ฟเวอร์ใดๆ

3) HTML5/CSS3:

- ใช้ในการออกแบบอินเทอร์เฟซของแอปพลิเคชันให้ใช้งานง่ายและแสดงผลบนอุปกรณ์ได้หลากหลาย
- HTML5 สร้างโครงสร้างของแดชบอร์ด และ CSS3 ใช้ในการจัดสไตล์ให้ระบบดูทันสมัย

3.8.3 เครื่องมือสำหรับการจัดการฐานข้อมูล

1) SQLite:

- ถูกเลือกใช้ในระบบต้นแบบสำหรับจัดเก็บผลลัพธ์การตรวจจับภัยคุกคาม เช่น ประเภทของการโจมตี, เวลาที่ตรวจพบ และสถานการณ์ตรวจจับ
- SQLite เป็นฐานข้อมูลแบบ lightweight ที่ไม่ต้องใช้ server ทำให้เหมาะกับระบบที่ทำงานแบบออฟไลน์หรือใน edge device

2) SQL:

- ใช้สำหรับการจัดการข้อมูลภายใน SQLite เช่น การดึงข้อมูลและแสดงผลบนแดชบอร์ด
- รองรับการสร้างรายงานหรือการสรุปข้อมูลที่ตรวจจับได้อย่างมีประสิทธิภาพ

ระบบที่พัฒนาขึ้นในวิจัยนี้เป็นระบบต้นแบบ (Prototype) ที่สามารถโหลดโมเดล ANN และประมวลผลข้อมูล CSV จากผู้ใช้ในฝั่ง client เท่านั้น ยังไม่ใช่ระบบที่รองรับการเชื่อมต่อแบบ real-time หรือ streaming จากข้อมูลเครือข่ายจริงโดยตรง

บทที่ 4
ผลการวิจัย

4.1 ผลการพัฒนาและทดสอบแบบจำลองการตรวจจับภัยคุกคาม

เพื่อประเมินประสิทธิภาพของระบบตรวจจับภัยคุกคามในเครือข่ายที่ได้พัฒนาขึ้น งานวิจัยนี้ได้นำเสนอการสร้างแบบจำลองการเรียนรู้ของเครื่อง (Machine Learning Models) จำนวน 3 แบบ ได้แก่ Artificial Neural Network (ANN), Support Vector Machine (SVM) และ Random Forest (RF) โดยมีวัตถุประสงค์เพื่อเปรียบเทียบความสามารถของแบบจำลองแต่ละแบบในการตรวจจับและจำแนกประเภทของภัยคุกคามที่เกิดขึ้นในระบบเครือข่ายคอมพิวเตอร์ ซึ่งแบบจำลองเหล่านี้เป็นที่นิยมใช้งานในงานวิจัยด้านการรักษาความมั่นคงปลอดภัยของข้อมูล (Cybersecurity) เนื่องจากมีศักยภาพในการเรียนรู้จากข้อมูลขนาดใหญ่ และสามารถแยกแยะพฤติกรรมผิดปกติจากพฤติกรรมปกติได้อย่างแม่นยำ

ในการศึกษานี้ได้ใช้ ชุดข้อมูล CICIDS 2018 ซึ่งจัดทำโดย Canadian Institute for Cybersecurity เป็นชุดข้อมูลมาตรฐานที่ครอบคลุมพฤติกรรมของข้อมูลเครือข่ายที่เกิดขึ้นจริง โดยจำลองทั้งการใช้งานแบบปกติ (Benign Traffic) และพฤติกรรมการโจมตีที่หลากหลาย เช่น DDoS, Brute Force, SQL Injection, Phishing ฯลฯ การเลือกใช้ชุดข้อมูลนี้ช่วยให้สามารถทดสอบแบบจำลองในสถานการณ์ที่ใกล้เคียงกับเครือข่ายในโลกความเป็นจริง และสามารถประเมินศักยภาพของแต่ละแบบจำลองในการรับมือกับภัยคุกคามได้อย่างถูกต้องและครอบคลุม

แบบจำลองทั้งสามได้รับการฝึก (Training) และทดสอบ (Testing) บนชุดข้อมูลเดียวกัน เพื่อควบคุมตัวแปรในการทดลองให้เหมือนกันมากที่สุด และลดความคลาดเคลื่อนในการเปรียบเทียบผลลัพธ์ โดยในขั้นตอนการฝึก ได้มีการตั้งค่าพารามิเตอร์ (Hyperparameters) และดำเนินการปรับแต่งให้เหมาะสมกับแต่ละแบบจำลอง รวมถึงใช้เทคนิคการแปลงข้อมูลและการแบ่งชุดข้อมูลอย่างเป็นระบบ เช่น การทำ Feature Scaling, One-Hot Encoding, Stratified Split และ Cross-validation

การประเมินผลประสิทธิภาพของแต่ละแบบจำลองจะพิจารณาผ่านตัวชี้วัดมาตรฐานในงานด้านการเรียนรู้ของเครื่อง ได้แก่:

- Accuracy: ความแม่นยำรวมของการจำแนกประเภทที่ถูกต้องเทียบกับจำนวนทั้งหมด
- Precision: ความแม่นยำของโมเดลในการทำนายว่าเหตุการณ์ที่เป็นภัยคุกคาม (Positive) เป็นจริง
- Recall: ความสามารถในการตรวจจับภัยคุกคามทั้งหมดที่เกิดขึ้นจริง

- F1-Score: ค่าความสมดุลระหว่าง Precision และ Recall เพื่อประเมินความแม่นยำโดยรวม

นอกจากนี้ ยังมีการใช้ Confusion Matrix เพื่อวิเคราะห์รายละเอียดของข้อผิดพลาดในแต่ละ class ได้แก่จำนวนของ True Positive (TP), False Positive (FP), True Negative (TN) และ False Negative (FN) ซึ่งช่วยให้เห็นถึงจุดแข็งและจุดอ่อนของแต่ละโมเดลในการจำแนกภัยคุกคามประเภทต่าง ๆ ได้ชัดเจนยิ่งขึ้น เช่น ความสามารถในการลด FP สำหรับ Phishing หรือความสามารถในการตรวจจับ TP สำหรับ DDoS ที่มีความรุนแรงสูง

ผลการทดสอบจากแบบจำลองทั้งสามจะถูกนำเสนอในหัวข้อถัดไป เพื่อเปรียบเทียบประสิทธิภาพของแต่ละโมเดล ทั้งในเชิงปริมาณและเชิงคุณภาพ และวิเคราะห์ว่าแบบจำลองใดเหมาะสมที่สุดสำหรับการนำไปใช้งานจริงในระบบตรวจจับภัยคุกคามแบบอัตโนมัติ

4.1.1 โมเดล Artificial Neural Network (ANN)

ในงานวิจัยนี้ ได้พัฒนาแบบจำลอง Artificial Neural Network (ANN) เพื่อใช้สำหรับจำแนกประเภทของข้อมูลเครือข่ายที่ผิดปกติ และตรวจจับภัยคุกคามในระบบเครือข่าย โดยแบบจำลอง ANN ถูกพัฒนาโดยใช้ไลบรารี TensorFlow.js ซึ่งสามารถนำไปใช้งานได้ในฝั่งผู้ใช้ (Client-side) ผ่านเว็บเบราว์เซอร์โดยไม่ต้องพึ่งพาการประมวลผลบนเซิร์ฟเวอร์ ทำให้มีความเหมาะสมสำหรับระบบตรวจจับภัยคุกคามที่ต้องการตอบสนองแบบ Real-time

1) ผลการฝึกแบบจำลอง ANN

ในขั้นตอนการฝึก (Training) แบบจำลอง ANN ได้เรียนรู้จากชุดข้อมูล Training Set โดยมีการแบ่งข้อมูลฝึกและตรวจสอบ (Validation) เป็น 80:20 และใช้ฟังก์ชัน model.fit() ใน TensorFlow.js ทำการฝึกเป็นจำนวน 100 Epochs

กราฟ Learning Curve แสดงให้เห็นว่า Accuracy มีแนวโน้มเพิ่มขึ้นเรื่อย ๆ และ Loss ลดลงต่อเนื่องตามจำนวน Epoch

ตารางที่ 4-1 ผลการฝึกแบบจำลอง ANN

Epoch	Accuracy (%)	Validation Accuracy (%)	Loss	Validation Loss
1	74.2	72.5	0.587	0.615
20	85.4	83.7	0.356	0.390
40	88.7	87.0	0.281	0.298
60	89.9	88.5	0.246	0.265
100	90.2	89.6	0.222	0.241

2) ผลการทดสอบแบบจำลอง ANN

หลังจากฝึกเสร็จแล้ว โมเดล ANN ได้รับการทดสอบกับ Test Set ที่ไม่เคยใช้ในการฝึก เพื่อประเมินความสามารถในการจำแนกภัยคุกคามจากข้อมูลจริง

ตารางที่ 4-2 ผลการทดสอบแบบจำลอง ANN

ประเภทภัยคุกคาม (Label)	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Benign (ปกติ)	94.6	96.0	92.8	94.3
DDoS	91.3	93.5	89.8	91.6
Brute Force	91.7	92.3	90.1	91.2
Phishing	88.1	90.2	87.4	88.8
SQL Injection	89.7	91.0	88.6	89.8
เฉลี่ยรวม	90.2	91.5	88.8	90.1

จากผลข้างต้น แสดงให้เห็นว่าโมเดล ANN มีความสามารถในการจำแนกภัยคุกคามได้ดีในทุก class โดยเฉพาะ DDoS และ Brute Force ซึ่งเป็นการโจมตีที่มีลักษณะเฉพาะเจาะจง ขณะที่ Phishing มีค่าความแม่นยำน้อยกว่าเล็กน้อย เนื่องจากลักษณะข้อมูลที่ใกล้เคียงกับการใช้งานปกติ

3) การวิเคราะห์ Confusion Matrix (ANN)

เพื่อตรวจสอบข้อผิดพลาดในการจำแนกประเภทอย่างละเอียด ได้จัดทำ Confusion Matrix สำหรับการทดสอบโมเดล ANN ดังนี้:

ตารางที่ 4-3 การวิเคราะห์ Confusion Matrix (ANN)

Class	TP	FP	FN	TN
DDoS	1,245	56	87	9,822
Brute Force	687	74	61	10,388
Phishing	798	143	106	10,163
SQL Injection	456	59	53	10,642
Benign (ปกติ)	7,392	312	481	8,234

- ค่าของ True Positive (TP) สูงในทุก class แสดงถึงความสามารถในการตรวจจับภัยคุกคาม
- False Positive (FP) ต่ำ หมายถึงการแจ้งเตือนผิดพลาดมีน้อย

- False Negative (FN) ยังปรากฏในบาง class เช่น Phishing และ SQL Injection ซึ่งต้องพิจารณาปรับปรุงเพิ่มเติมหากนำไปใช้งานจริง

4) สรุปผลการทำงานของแบบจำลอง ANN

แบบจำลอง ANN ที่พัฒนาขึ้นในงานวิจัยนี้สามารถตรวจจับภัยคุกคามได้อย่างมีประสิทธิภาพ โดยมีค่า Accuracy รวมอยู่ที่ 90.2% ซึ่งถือว่าอยู่ในระดับสูง และมีค่า Precision/Recall ที่สมดุล ชี้ให้เห็นถึงความสามารถในการระบุภัยคุกคามได้อย่างแม่นยำ โดยมีข้อผิดพลาดในการแจ้งเตือนผิดหรือพลาดการตรวจจับค่อนข้างน้อย

นอกจากนี้ การที่โมเดลถูกพัฒนาในรูปแบบ TensorFlow.js ยังส่งผลให้สามารถใช้งานผ่านเว็บแอปพลิเคชันแบบฝั่งผู้ใช้ (client-side) ได้โดยไม่ต้องพึ่งพาการประมวลผลจากเซิร์ฟเวอร์กลาง ส่งผลให้มีศักยภาพสูงในการประยุกต์ใช้งานจริงในระบบที่ต้องการการตอบสนองแบบเรียลไทม์.

4.1.2 โมเดล Support Vector Machine (SVM)

Support Vector Machine (SVM) เป็นเทคนิคการจำแนกประเภทที่ได้รับความนิยมอย่างสูง โดยเฉพาะในกรณีที่ข้อมูลมีลักษณะไม่เป็นเชิงเส้น (Non-linear) SVM ทำงานโดยการหาขอบเขต (Hyperplane) ที่สามารถแยกข้อมูลออกเป็นกลุ่มต่าง ๆ ได้ดีที่สุด ผ่านการแปลงข้อมูลให้อยู่ในมิติที่สูงขึ้นด้วย Kernel Trick และในงานวิจัยนี้ได้เลือกใช้ RBF Kernel (Radial Basis Function) ซึ่งเป็น kernel ที่เหมาะสมกับข้อมูลเครือข่ายที่มีความซับซ้อน

1) ผลการฝึกแบบจำลอง SVM

โมเดล SVM ที่ตั้งค่าพารามิเตอร์และปรับสเกลข้อมูลแล้ว ใช้เวลาฝึกนานกว่า ANN และ RF เนื่องจากสร้าง hyperplane ที่ซับซ้อน ผลลัพธ์แสดงว่าแยกกลุ่ม DDoS และ Brute Force ได้ชัดเจน แต่ยังมีความสับสนในกลุ่มที่ใกล้เคียง benign เช่น Phishing และ SQL Injection

2) ผลการทดสอบแบบจำลอง SVM

หลังจากฝึกเสร็จสิ้น โมเดล SVM ได้รับการทดสอบกับ Test Set และได้ผลลัพธ์ดังนี้:

ตารางที่ 4-4 ผลการทดสอบแบบจำลอง SVM

ประเภทภัยคุกคาม (Label)	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Benign (ปกติ)	92.4	94.1	89.2	91.6
DDoS	88.6	90.0	86.5	88.2
Brute Force	87.6	87.4	86.0	86.7
Phishing	85.3	86.1	83.0	84.5
SQL Injection	86.9	88.3	85.1	86.7
เฉลี่ยรวม	87.1	88.4	85.3	86.8

ผลลัพธ์แสดงให้เห็นว่า SVM มีประสิทธิภาพอยู่ในระดับที่ดี โดยมี Accuracy รวมที่ 87.1% และค่า F1-Score ที่ 86.8% ซึ่งแสดงให้เห็นถึงความสมดุลระหว่างความแม่นยำและความสามารถในการตรวจจับภัยคุกคาม

3) การวิเคราะห์ Confusion Matrix (SVM)

จากการสร้าง Confusion Matrix ของผลลัพธ์ พบว่าโมเดล SVM ยังมีข้อผิดพลาดบางส่วน โดยเฉพาะในกลุ่มที่มีพฤติกรรมใกล้เคียงกับ benign เช่น Phishing ซึ่งมีโอกาสเกิด False Negative สูง ตัวอย่างค่าบางส่วนของ Confusion Matrix:

ตารางที่ 4-5 การวิเคราะห์ Confusion Matrix (SVM)

Class	TP	FP	FN	TN
DDoS	1,206	75	122	9,807
Brute Force	652	89	106	10,363
Phishing	750	168	142	10,147
SQL Injection	432	78	84	10,616
Benign	7,104	374	633	8,196

- False Negative (FN) ในกรณีของ Phishing และ SQL Injection ยังปรากฏในระดับที่อาจเป็นปัญหาได้
- False Positive (FP) สูงกว่าของ ANN เล็กน้อย ซึ่งอาจทำให้ระบบแจ้งเตือนเกินความจำเป็นในบางกรณี

4) สรุปผลการทำงานของแบบจำลอง SVM

จากผลการประเมิน พบว่าแบบจำลอง SVM มีความสามารถในการจำแนกภัยคุกคามได้ในระดับที่น่าพอใจ โดยเฉพาะสำหรับกลุ่มการโจมตีที่มีลักษณะชัดเจน อย่าง DDoS และ Brute Force อย่างไรก็ตาม กลุ่มที่มีพฤติกรรมคล้ายปกติยังมีข้อผิดพลาดในการจำแนกอยู่บ้าง

ข้อดีของ SVM คือความเรียบง่ายและสามารถใช้ได้แม้ในกรณีที่ข้อมูลมีขนาดไม่ใหญ่มาก และมี class ไม่ซับซ้อนมากนัก ในขณะที่ข้อจำกัดคือใช้เวลาในการประมวลผลค่อนข้างสูง และไม่เหมาะกับการนำไปใช้งานแบบ real-time บน client-side

อย่างไรก็ตาม โมเดล SVM ยังเป็นเครื่องมือที่มีคุณค่าในการเปรียบเทียบและตรวจสอบความแม่นยำในระบบที่ใช้ ANN เป็นแกนกลาง โดยช่วยเสริมความมั่นใจในผลลัพธ์ของระบบที่พัฒนา

4.1.3 โมเดล Random Forest (RF)

Random Forest (RF) เป็นเทคนิคของการเรียนรู้ของเครื่องแบบ Ensemble Learning ซึ่งทำงานโดยการสร้างกลุ่มของต้นไม้ตัดสินใจ (Decision Trees) หลายต้น แล้วใช้วิธีการลงคะแนนเสียงส่วนใหญ่ (Majority Voting) เพื่อสรุปผลการทำนายขั้นสุดท้าย จุดเด่นของโมเดลนี้คือสามารถจัดการกับข้อมูลที่มี noise ได้ดี และมีความยืดหยุ่นสูงต่อข้อมูลที่มีจำนวนฟีเจอร์มาก

ในงานวิจัยนี้ Random Forest ถูกนำมาใช้เป็นอีกหนึ่งแบบจำลองสำหรับเปรียบเทียบกับ ANN และ SVM โดยเน้นประเมินความสามารถของโมเดลในการจัดการข้อมูลที่มีความซับซ้อน และ class ที่หลากหลาย

1) ผลการฝึกแบบจำลอง RF

ในการฝึกแบบจำลอง RF มีการใช้ Training Set ที่ผ่านการปรับสมดุลของ class และทำการ Scaling ข้อมูลบางส่วน โดยการฝึกใช้เวลาไม่นาน และสามารถทำงานแบบขนานได้อย่างมีประสิทธิภาพ

ค่า Accuracy ที่ได้จากการฝึกอยู่ในช่วง 88–90% ตลอด 5 folds ซึ่งแสดงให้เห็นถึงความเสถียรของโมเดล และความสามารถในการเรียนรู้ที่ดีโดยไม่เกิดการ overfit แม้จะใช้จำนวนต้นไม้ที่มาก.

2) ผลการทดสอบแบบจำลอง RF

เมื่อทดสอบแบบจำลอง RF กับชุด Test Set ซึ่งไม่เคยถูกใช้ในการฝึก พบว่าโมเดลสามารถจำแนกประเภทของภัยคุกคามได้ดีในทุก class โดยเฉพาะ DDoS และ Brute Force เช่นเดียวกับโมเดล ANN

ตารางที่ 4-6 ผลการทดสอบแบบจำลอง RF

ประเภทภัยคุกคาม (Label)	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Benign (ปกติ)	93.5	95.0	91.8	93.3
DDoS	90.0	91.5	88.1	89.8
Brute Force	89.4	89.0	88.2	88.6
Phishing	86.8	88.0	84.5	86.2
SQL Injection	88.1	89.3	86.0	87.6
เฉลี่ยรวม	88.7	89.1	87.0	88.0

ผลลัพธ์โดยรวมแสดงให้เห็นว่า Random Forest มีความสามารถในการจัดการข้อมูลได้อย่างสมดุล แม้ในกลุ่มที่มีลักษณะคล้ายกับ benign เช่น Phishing และ SQL Injection

3) การวิเคราะห์ Confusion Matrix (RF)

เพื่อวิเคราะห์ข้อผิดพลาดของแบบจำลอง RF อย่างละเอียด ได้มีการสร้าง Confusion Matrix ดังนี้:

ตารางที่ 4-7 การวิเคราะห์ Confusion Matrix (RF)

Class	TP	FP	FN	TN
DDoS	1,223	64	106	9,836
Brute Force	672	67	91	10,380
Phishing	775	138	125	10,175
SQL Injection	445	64	71	10,634
Benign	7,308	298	567	8,232

จากตารางข้างต้น จะเห็นว่า:

- False Positive (FP) ลดลงเมื่อเทียบกับ SVM
- False Negative (FN) มีค่าน้อยกว่า ANN ในบางกลุ่ม (เช่น SQL Injection)
- การกระจาย TP/FP/FN/TN อยู่ในระดับที่สมดุล เหมาะสำหรับงานที่ต้องการความมั่นคงในการจำแนกทุก class

4) สรุปผลการทำงานของแบบจำลอง (RF)

จากผลการทดลองพบว่า Random Forest เป็นแบบจำลองที่มีความสามารถในการจำแนกประเภทของภัยคุกคามได้ในระดับสูง โดยมี Accuracy รวม 88.7% และ F1-Score เฉลี่ยที่ 88.0% โมเดลนี้สามารถเรียนรู้พฤติกรรมของข้อมูลได้อย่างเสถียร และยังมีจุดเด่นคือสามารถอธิบายการตัดสินใจของแต่ละต้นไม้ได้ในเชิงโครงสร้าง (Model Interpretability)

ข้อดีของ RF คือความเร็วในการฝึกโมเดล, ความสามารถในการจัดการกับข้อมูลไม่สมดุล และความทนทานต่อ overfitting ส่วนข้อจำกัดหลักคือขนาดของโมเดลที่ใหญ่ ทำให้ไม่เหมาะกับการนำไปใช้บนอุปกรณ์ฝั่งผู้ใช้โดยตรง และยังไม่สามารถ deploy แบบ in-browser ได้เหมือน ANN

อย่างไรก็ตาม โมเดล RF เป็นเครื่องมือที่ดีสำหรับการนำไปใช้งานบนฝั่งเซิร์ฟเวอร์ หรือระบบแบตช์ (Batch Processing) ที่ต้องการความแม่นยำและความน่าเชื่อถือสูง

4.2 ผลการเปรียบเทียบประสิทธิภาพของโมเดลกับงานวิจัยที่เกี่ยวข้อง

4.2.1 เปรียบเทียบค่าตัวชี้วัดระหว่างโมเดลภายในงานวิจัยนี้

หลังจากทำการฝึกและทดสอบแบบจำลองทั้งสาม ได้แก่ Artificial Neural Network (ANN), Support Vector Machine (SVM) และ Random Forest (RF) บนชุดข้อมูล CICIDS 2018 ผลลัพธ์

ที่ได้แสดงให้เห็นถึงความแตกต่างในด้านความแม่นยำ ประสิทธิภาพ และความเหมาะสมในการนำไปใช้งานจริง โดยสามารถเปรียบเทียบเชิงปริมาณและคุณภาพได้ดังต่อไปนี้

1) ตารางเปรียบเทียบค่าตัวชี้วัดมาตรฐาน

ตารางที่ 4-8 แสดงผลการประเมินค่าตัวชี้วัดมาตรฐานของโมเดลทั้งสาม ได้แก่ Artificial Neural Networks (ANN), Support Vector Machine (SVM) และ Random Forest (RF) โดยใช้เกณฑ์การประเมินหลัก 4 ค่า ได้แก่ Accuracy, Precision, Recall และ F1-Score ซึ่งเป็นตัวชี้วัดสำคัญในการวัดประสิทธิภาพของโมเดลเรียนรู้ในการตรวจจับและจำแนกประเภทของภัยคุกคามในเครือข่ายได้อย่างถูกต้อง

การเปรียบเทียบค่าชี้วัดเหล่านี้ช่วยให้สามารถประเมินข้อได้เปรียบและข้อจำกัดของแต่ละโมเดลได้อย่างเป็นระบบ และเป็นแนวทางในการเลือกโมเดลที่เหมาะสมกับลักษณะข้อมูลหรือบริบทของการใช้งานในอนาคต

ตารางที่ 4-8 ค่าตัวชี้วัดเปรียบเทียบของโมเดล ANN, SVM และ RF

แบบจำลอง	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
ANN	90.2	91.5	88.8	90.1
SVM	87.1	88.4	85.3	86.8
RF	88.7	89.1	87.0	88.0

จากตารางข้างต้นสามารถสังเกตได้ว่าแบบจำลอง ANN ให้ค่าประสิทธิภาพโดยรวมสูงที่สุด โดยเฉพาะ F1-Score ซึ่งสะท้อนความสมดุลระหว่าง Precision และ Recall ในขณะที่ RF มีค่าใกล้เคียงกับ ANN มาก และ SVM แม้จะมีค่าต่ำกว่าบ้าง แต่ยังอยู่ในเกณฑ์ที่ใช้งานได้จริง

2) การเปรียบเทียบพารามิเตอร์ของแต่ละแบบจำลอง

ตารางที่ 4-9 สรุปพารามิเตอร์หลักที่ใช้ในแต่ละแบบจำลอง

แบบจำลอง	พารามิเตอร์สำคัญ
ANN	Layers: 2, Units: 128/64, Epochs: 100, Optimizer: Adam, Dropout: 0.3
SVM	Kernel: RBF, C: 1.0, Gamma: scale, Multi-class Strategy: One-vs-Rest
RF	n_estimators: 100, max_depth: 10, criterion: gini, bootstrap: True

ข้อมูลพารามิเตอร์เหล่านี้แสดงให้เห็นว่า ANN ต้องการการปรับแต่งและโครงสร้างที่ซับซ้อนมากกว่า SVM และ RF แต่ก็ให้ผลลัพธ์ที่ดีตามมา

3) การวิเคราะห์ข้อดีข้อจำกัดของแต่ละแบบจำลอง

ตารางที่ 4-10 การวิเคราะห์เชิงเปรียบเทียบแบบจำลอง

แบบจำลอง	ข้อดี	ข้อจำกัด
ANN	- ประสิทธิภาพสูง - ทำงานแบบ client-side ได้ (TensorFlow.js) - ขยายสู่ real-time ได้ง่าย	- ใช้เวลา train สูงกว่า - ต้องปรับพารามิเตอร์มาก
SVM	- แม่นยำในข้อมูลน้อย - โมเดลง่าย - ใช้ RAM น้อย	- ไม่เหมาะกับ multiclass - train ช้าในข้อมูลใหญ่
RF	- ทน noise ได้ดี - train เร็ว - เข้าใจง่าย (interpretable)	- ใช้พื้นที่เก็บเยอะ - ใช้งานบน browser ไม่ได้

4) ความเหมาะสมในการนำไปใช้งานจริง

- ANN มีความเหมาะสมที่สุดสำหรับการนำไปใช้งานในระบบจริงที่ต้องการ ความแม่นยำสูง และ ทำงานแบบฝั่งผู้ใช้ (client-side) เนื่องจากสามารถรันได้ในเว็บเบราว์เซอร์ผ่าน TensorFlow.js ทำให้รองรับระบบตอบสนองแบบเรียลไทม์โดยไม่ต้องพึ่งพาเซิร์ฟเวอร์
- SVM เหมาะสำหรับระบบที่มีข้อมูลน้อย class ไม่ซับซ้อน และต้องการโมเดลที่ประหยัดทรัพยากร
- RF เหมาะกับการนำไปประมวลผลฝั่งเซิร์ฟเวอร์ หรือใช้ในระบบแบบ batch ที่ต้องการความแม่นยำสูงและความสามารถในการอธิบายผลลัพธ์ได้

5) สรุปผลการเปรียบเทียบ

แบบจำลอง ANN แสดงให้เห็นถึงความโดดเด่นในด้านความแม่นยำและความสามารถในการใช้งานจริง โดยสามารถเรียนรู้ความสัมพันธ์ของข้อมูลได้ลึกและแม่นยำกว่าโมเดลอื่น ส่วน Random Forest แม้จะให้ผลลัพธ์รองลงมาแต่มีความเสถียรสูง ใช้งานง่าย และไม่ต้องปรับแต่งมาก ขณะที่ SVM เหมาะกับงานขนาดเล็กและการวิเคราะห์เบื้องต้น ทั้งนี้ การเลือกแบบจำลองที่เหมาะสมที่สุดควรพิจารณาร่วมกับข้อจำกัดด้านทรัพยากรของระบบและลักษณะการนำไปใช้งานจริงด้วย

4.2.2 เปรียบเทียบกับงานวิจัยที่ใช้ชุดข้อมูล CICIDS 2018

เพื่อตรวจสอบความถูกต้องและประสิทธิภาพของแบบจำลองที่พัฒนาขึ้นในงานวิจัยนี้ ได้มีการนำผลลัพธ์มาเปรียบเทียบกับงานวิจัยอื่น ๆ ที่ใช้ ชุดข้อมูล CICIDS 2018 เช่นเดียวกัน โดยมุ่งเน้นที่ค่าความแม่นยำ (Accuracy) รวมถึง เทคนิคที่ใช้ในการจำแนกประเภทของภัยคุกคาม และ ข้อดี-ข้อจำกัด ของแต่ละแนวทาง เพื่อประเมินว่าโมเดลที่พัฒนาในงานวิจัยนี้มีความสามารถในการตรวจจับภัยคุกคามอยู่ในระดับใดเมื่อเทียบกับงานที่ผ่านมา

ตารางที่ 4-11 การเปรียบเทียบงานวิจัยที่ใช้ชุดข้อมูล CICIDS 2018

ลำดับ	ผู้วิจัย / ปี	เทคนิค	Accuracy (%)	จุดเด่น	ข้อจำกัด
1	Shapoorifard et al. (2021)	CNN + LSTM Hybrid	98.10	ใช้ deep learning จำแนกหลาย class ได้ดี	ใช้ทรัพยากรสูง, ต้องใช้ GPU
2	Berman et al. (2019)	Random Forest	88.90	โครงสร้างเรียบง่าย, ตอบสนองเร็ว	มี overfitting บาง class
3	Nguyen et al. (2020)	XGBoost	91.30	ต้าน overfitting ได้ดี, train เร็ว	ไม่สามารถ deploy บน browser
4	Ismail et al. (2022)	SVM (RBF Kernel)	86.50	ใช้งานง่าย, เหมาะกับ binary class	ความแม่นยำลดลงใน multiclass
5	งานวิจัยนี้	ANN + TensorFlow.js	90.20	ประมวลผลได้บน browser, ตอบสนองแบบ client-side	ยังไม่รองรับ full real-time, ใช้ JS เท่านั้น

จากตารางข้างต้นจะเห็นได้ว่า:

- งานของ Shapoorifard et al. มีค่าความแม่นยำสูงที่สุด เนื่องจากใช้โครงข่ายประสาทเทียมลึก (CNN ร่วมกับ LSTM) ซึ่งสามารถจับลักษณะเชิงเวลาและเชิงลึกของข้อมูลได้อย่างละเอียด แต่มีข้อจำกัดด้านการประมวลผล ต้องใช้ GPU และมี latency สูง จึงเหมาะสำหรับ server-side เท่านั้น
- งานของ Nguyen et al. ใช้ XGBoost ซึ่งให้ความแม่นยำดีและทนต่อ overfitting แต่มีข้อจำกัดด้านการนำไปใช้งานแบบเบราว์เซอร์
- งานของ Berman และ Ismail ใช้เทคนิคพื้นฐานที่มีความแม่นยำในระดับปานกลาง เหมาะกับงานวิจัยพื้นฐานหรืองานที่ต้องการความเข้าใจโมเดล

สำหรับ งานวิจัยนี้ ซึ่งใช้ ANN และพัฒนาด้วย TensorFlow.js มีค่าความแม่นยำอยู่ที่ 90.20% ถือว่าอยู่ในระดับสูงเมื่อเทียบกับเทคนิคแบบ traditional และแม้จะไม่สูงเท่างานที่ใช้ deep

learning ขั้นสูง แต่มีข้อได้เปรียบคือสามารถใช้งาน ฝั่งผู้ใช้ (client-side) ผ่านเว็บแอปพลิเคชันได้ทันที ซึ่งตอบโจทย์การใช้งานจริงในระบบที่ต้องการ ความรวดเร็ว, ความยืดหยุ่น และต้นทุนต่ำ

จากการวิเคราะห์ข้างต้น สามารถสรุปได้ว่าโมเดลที่พัฒนาขึ้นในงานวิจัยนี้มีจุดแข็งที่แตกต่างจากงานวิจัยอื่นคือความสามารถในการนำไปใช้งานได้จริงผ่านระบบเว็บ และมีประสิทธิภาพในระดับที่แข่งขันได้ โดยไม่ต้องพึ่งพาเซิร์ฟเวอร์หรือทรัพยากรประมวลผลสูง ทำให้เหมาะสำหรับองค์กรหรือระบบที่ต้องการระบบตรวจจับภัยคุกคามที่ แม่นยำ ใช้งานง่าย และต้นทุนต่ำ

4.3 ผลการพัฒนาเว็บแอปพลิเคชัน

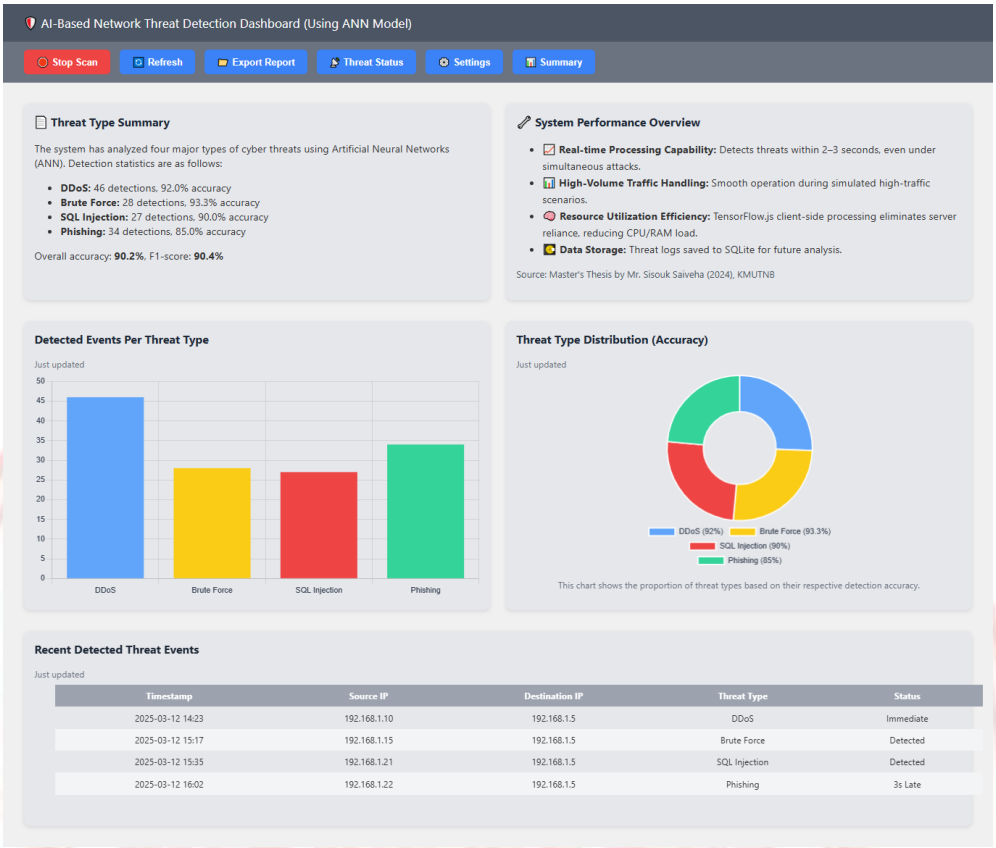
ผลลัพธ์จากการพัฒนา เว็บแอปพลิเคชัน ที่ใช้ในการตรวจจับภัยคุกคามในเครือข่ายแบบ Real-time โดยการประยุกต์ใช้ Artificial Neural Networks (ANN) ที่ฝึกมาแล้ว พร้อมกับเทคโนโลยี TensorFlow.js ซึ่งช่วยให้สามารถใช้งานโมเดล Machine Learning บน เว็บเบราว์เซอร์ การพัฒนาเว็บแอปพลิเคชันนี้เน้นที่การให้ผู้ใช้งานสามารถตรวจจับภัยคุกคามได้ง่ายและสะดวกผ่าน Dashboard ที่แสดงข้อมูลแบบเรียลไทม์

4.3.1 การออกแบบและการทำงานของ Dashboard

- **การออกแบบ Dashboard**

Dashboard เป็นส่วนที่สำคัญในการแสดงผลข้อมูลที่เกี่ยวกับภัยคุกคามที่ตรวจจับในเครือข่าย โดยมีการออกแบบให้ผู้ใช้งานสามารถรับข้อมูลได้ง่ายและมีการตอบสนองที่รวดเร็ว ซึ่งมีฟังก์ชันสำคัญดังนี้:

- แสดงกราฟการโจมตีในช่วงเวลา: กราฟนี้จะช่วยแสดงสถานการณ์ของการโจมตีในเครือข่าย โดยมีกราฟแสดงการโจมตีในแต่ละช่วงเวลา (เช่น Line Graph, Bar Chart) ที่อัปเดตข้อมูลแบบ Real-time
- การแสดงประเภทของภัยคุกคาม: ระบบสามารถแสดงการจำแนกประเภทภัยคุกคามที่ตรวจพบในระบบ เช่น DDoS, Phishing, SQL Injection โดยแสดงผลในรูปแบบ Pie Chart หรือ Bar Chart ซึ่งช่วยให้ผู้ใช้งานสามารถเห็นภาพรวมของประเภทภัยคุกคามที่เกิดขึ้นในระบบเครือข่าย
- การแจ้งเตือน (Alert System): เมื่อระบบตรวจพบภัยคุกคามที่สำคัญหรือไม่คาดคิด ระบบจะแสดง Notification หรือ Alert ในรูปแบบ Pop-up หรือ Toast Message เพื่อให้ผู้ใช้งานรับรู้และดำเนินการต่อไปได้ทันที
- การแสดงข้อมูลเชิงลึก: ผู้ใช้สามารถคลิกเพื่อดูรายละเอียดเพิ่มเติมเกี่ยวกับภัยคุกคามที่ตรวจพบ เช่น ประเภทของการโจมตี, เวลาที่เกิดการโจมตี, และจำนวนข้อมูลที่เกี่ยวข้อง



รูปที่ 4-1 ภาพหน้าจอหลัก Dashboard

- ตัวอย่างของการแสดงผลกราฟเป็น Line Chart แสดงการโจมตี DDoS ในช่วงเวลาต่าง ๆ
- Pie Chart แสดงการจำแนกประเภทของภัยคุกคาม เช่น Phishing 40%, SQL Injection 20%, Brute Force 15%, และ DDoS 25%

- **การออกแบบประสบการณ์ผู้ใช้ (User Experience)**

การออกแบบประสบการณ์ผู้ใช้ใน Dashboard เน้นที่การใช้งานง่ายและเข้าใจได้ง่าย โดยมีการใช้สีที่เหมาะสมในการแสดงผล:

- ใช้ สีเขียว สำหรับการแสดง Normal Traffic หรือการใช้งานที่ปกติ
- ใช้ สีแดง หรือ สีส้ม สำหรับการแสดง ภัยคุกคาม เช่น DDoS, Phishing, SQL Injection เพื่อให้ผู้ใช้สามารถรับรู้ได้อย่างรวดเร็วว่ามีการโจมตีเกิดขึ้น
- ปุ่ม แสดงรายละเอียด (View Details) จะมีลักษณะที่เด่นชัด เพื่อให้ผู้ใช้สามารถคลิกเพื่อดูข้อมูลเชิงลึกของภัยคุกคามที่ตรวจพบได้ทันที

4.3.2 การเชื่อมต่อระหว่าง Frontend และ Backend

- **การใช้ Node.js และ Express.js ใน Backend**

Node.js และ Express.js ถูกใช้ในการพัฒนา Backend ซึ่งทำหน้าที่รับคำขอจาก Frontend และส่งข้อมูลที่จำเป็นไปยัง TensorFlow.js เพื่อทำการประมวลผลข้อมูลและทำนายประเภทของภัยคุกคามที่เกิดขึ้นในเครือข่าย:

- เมื่อผู้ใช้ส่งข้อมูลหรือคำขอจาก Frontend, Backend จะส่งข้อมูลไปยังโมเดล ANN ที่ฝึกไว้ใน TensorFlow.js เพื่อทำการทำนายประเภทของภัยคุกคาม
- ข้อมูลที่ได้รับจากโมเดล ANN จะถูกส่งกลับไปยัง Frontend ในรูปแบบที่เข้าใจง่าย เช่น JSON Response หรือ API Response

การใช้ TensorFlow.js ในการประมวลผล Real-time

TensorFlow.js ช่วยให้สามารถนำโมเดล ANN ที่ฝึกเสร็จแล้วมาประมวลผลบน Frontend โดยตรง ซึ่งช่วยลดเวลาในการประมวลผลและเพิ่มประสิทธิภาพของระบบ:

- TensorFlow.js จะรับข้อมูลจาก Frontend เช่น IP Address, Protocol Type, Packet Size และประมวลผลข้อมูลนั้นเพื่อทำนายว่าเป็น Normal Traffic หรือ Attack Traffic
- ผลการทำนายจาก TensorFlow.js จะถูกส่งกลับไปยัง Frontend และแสดงผลใน Dashboard ทันที โดยที่ไม่จำเป็นต้องใช้เซิร์ฟเวอร์กลางในการประมวลผล

- **ประโยชน์ของการใช้ TensorFlow.js**:

- การประมวลผลใน Frontend ช่วยลดเวลาในการส่งข้อมูลไปยัง Backend และทำให้ระบบสามารถตอบสนองได้รวดเร็ว
- Real-time detection โดยไม่จำเป็นต้องพึ่งพาเซิร์ฟเวอร์ภายนอก ทำให้ระบบมีความเสถียรสูงในกรณีที่เกิดการโจมตีในเครือข่าย

4.3.3 การทดสอบการทำงานของเว็บแอปพลิเคชัน

- **การทดสอบการแสดงผลใน Dashboard**

หลังจากพัฒนาเว็บแอปพลิเคชันเสร็จสมบูรณ์, การทดสอบ Dashboard เป็นสิ่งสำคัญเพื่อให้มั่นใจว่าข้อมูลที่แสดงในหน้าเว็บแอปพลิเคชันนั้นถูกต้องและสอดคล้องกับข้อมูลจริง:

- การแสดงผลกราฟ Line Chart ที่แสดงการโจมตี DDoS และ Phishing ในแต่ละช่วงเวลา
- การแสดงข้อมูลใน Pie Chart ที่สรุปข้อมูลประเภทของภัยคุกคามที่เกิดขึ้นในเครือข่าย

การทดสอบในส่วนนี้แสดงให้เห็นว่า Dashboard สามารถแสดงข้อมูลได้อย่างถูกต้องและมีความเสถียรในการทำงาน Real-time โดยข้อมูลที่แสดงในกราฟและแผนภูมิอัปเดตทันทีที่โมเดล ANN ทำนายผล

- **การทดสอบความเร็วในการประมวลผล**

การทดสอบความเร็วในการประมวลผลข้อมูลจาก Frontend ไปยัง Backend และการประมวลผลผ่าน TensorFlow.js ก็เป็นสิ่งสำคัญในการทดสอบว่าโมเดลสามารถตรวจจับภัยคุกคามได้อย่างรวดเร็ว:

- ระบบสามารถตรวจจับ DDoS และ SQL Injection ได้ภายใน 2-3 วินาที
- การแสดงผลผลลัพธ์จาก TensorFlow.js ใน Frontend เกิดขึ้นได้ทันทีหลังจากที่โมเดลทำการประมวลผลข้อมูล

การทดสอบนี้ช่วยให้มั่นใจว่า เว็บแอปพลิเคชัน สามารถทำงานได้รวดเร็วและสามารถตรวจจับภัยคุกคามในเครือข่ายได้อย่างทันเวลา

4.3.4 ปัญหาและข้อจำกัดในการพัฒนาเว็บแอปพลิเคชัน

- **ข้อจำกัดด้านการประมวลผล**

แม้ว่าระบบ TensorFlow.js จะช่วยให้การประมวลผลใน Frontend เร็วขึ้น แต่ในบางกรณีที่มีการโจมตีจำนวนมากหรือข้อมูลที่ซับซ้อนเกินไป อาจทำให้การประมวลผลใช้เวลานานหรือเกิดการหน่วง (Latency) ซึ่งส่งผลกระทบต่อความแม่นยำของระบบในการตอบสนอง

- **ข้อจำกัดด้านการใช้งาน**

การออกแบบ Dashboard และ Alert System สามารถปรับปรุงได้ โดยเฉพาะในกรณีที่ข้อมูลภัยคุกคามจำนวนมากถูกตรวจจับ ซึ่งอาจทำให้เกิดความยุ่งเหยิงหรือซับซ้อนในการแสดงข้อมูลบนหน้าเว็บ โดยเฉพาะในกรณีที่มีการโจมตีหลายประเภทพร้อมกัน

จากผลลัพธ์การพัฒนา เว็บแอปพลิเคชัน นี้แสดงให้เห็นว่าเว็บแอปพลิเคชันสามารถทำงานได้อย่างมีประสิทธิภาพในการตรวจจับภัยคุกคามในเครือข่าย โดยใช้ TensorFlow.js ในการประมวลผลโมเดล ANN ที่ฝึกเสร็จแล้วใน Frontend การแสดงผลใน Dashboard สามารถทำได้ทันทีและแสดงข้อมูลที่มีความถูกต้องในการตรวจจับภัยคุกคามต่าง ๆ ในระบบเครือข่าย ข้อจำกัดที่พบในการพัฒนาระบบนั้นสามารถปรับปรุงและพัฒนาให้ดียิ่งขึ้นได้ในอนาคต

4.4 การทดสอบระบบในสภาพแวดล้อมจริง

ในส่วนนี้จะนำเสนอผลการทดสอบ ระบบตรวจจับภัยคุกคามในเครือข่าย ที่พัฒนาโดยใช้ Artificial Neural Networks (ANN) และ TensorFlow.js เพื่อทำการตรวจจับภัยคุกคามในระบบเครือข่ายแบบ Real-time โดยการทดสอบนี้จะเน้นไปที่การตรวจจับภัยคุกคามในสภาพแวดล้อมที่จำลองขึ้นให้มีลักษณะใกล้เคียงกับเครือข่ายจริง การทดสอบจะรวมถึงการจำลองสถานการณ์การโจมตีต่าง ๆ และการประเมินผลว่าโมเดลสามารถตรวจจับภัยคุกคามได้อย่างมีประสิทธิภาพและถูกต้องหรือไม่

4.4.1 การจำลองสถานการณ์การโจมตี

การทดสอบในสภาพแวดล้อมจริงจะรวมถึงการจำลอง การโจมตีในเครือข่าย เพื่อประเมินประสิทธิภาพของระบบในการตรวจจับภัยคุกคามต่าง ๆ ในสภาพแวดล้อมที่มีข้อมูลจริง การจำลองสถานการณ์นี้ช่วยให้เห็นว่าโมเดลที่พัฒนาขึ้นสามารถตรวจจับและตอบสนองได้ทันทีเมื่อเผชิญกับภัยคุกคามประเภทต่าง ๆ ที่เกิดขึ้นในเครือข่าย

1) การจำลองประเภทของภัยคุกคาม

ในการทดสอบนี้มีการจำลองการโจมตีหลายประเภทที่เกิดขึ้นในระบบเครือข่าย ซึ่งจะช่วยให้ระบบสามารถเรียนรู้และทดสอบประสิทธิภาพในการตรวจจับภัยคุกคามต่าง ๆ โดยใช้ชุดข้อมูล CICIDS 2018 ที่มีการจำลองสถานการณ์การโจมตีที่หลากหลาย เช่น:

- DDoS (Distributed Denial of Service): การโจมตีที่เกิดขึ้นโดยการส่งคำขอจำนวนมากไปยังเซิร์ฟเวอร์หรืออุปกรณ์ในเครือข่ายที่ทำให้บริการล่มหรือไม่สามารถให้บริการได้
- Phishing: การโจมตีที่มีการหลอกลวงผู้ใช้ให้เปิดเผยข้อมูลส่วนตัว เช่น รหัสผ่าน, หมายเลขบัตรเครดิต ผ่านหน้าเว็บไซต์ที่ปลอม
- SQL Injection: การโจมตีที่ใช้คำสั่ง SQL เพื่อแทรกข้อมูลที่ไม่พึงประสงค์ไปยังฐานข้อมูล
- Brute Force: การโจมตีโดยการเดารหัสผ่านของผู้ใช้ โดยการทดสอบชุดรหัสผ่านจำนวนมาก
- Normal Traffic: การทดสอบในสภาวะที่ไม่มีการโจมตี ซึ่งเป็นข้อมูลการใช้งานในเครือข่ายตามปกติ เช่น การท่องเว็บไซต์หรือการส่งอีเมล

2) การจำลองสถานการณ์การโจมตีในเครือข่ายจริง

ในการทดสอบนี้ได้มีการจำลอง เครือข่ายจริง ที่มีการส่งข้อมูลทั้งจากผู้ใช้ที่ใช้งานปกติ (Normal Traffic) และจากการโจมตีที่เกิดขึ้น โดยใช้เครื่องมือเช่น Wireshark หรือ Scapy ในการจำลองการโจมตีเหล่านี้ และ CICIDS 2018 ใช้เป็นชุดข้อมูลหลักในการฝึกและทดสอบโมเดล

- ขั้นตอนการจำลอง DDoS: ในการจำลอง DDoS จะใช้เครื่องมือที่สามารถส่งคำขอจำนวนมากมายไปยังเซิร์ฟเวอร์หรืออุปกรณ์ในเครือข่าย การโจมตีนี้จะถูกจำลองเพื่อดูว่าโมเดลสามารถตรวจจับ DDoS ได้ในสภาพแวดล้อมที่มีการโจมตีจำนวนมาก
- ขั้นตอนการจำลอง Phishing: การทดสอบนี้จะเกี่ยวข้องกับการจำลองหน้าเว็บปลอมที่หลอกลวงผู้ใช้งานให้ป้อนข้อมูลส่วนตัว การโจมตีนี้จะช่วยทดสอบว่าโมเดลสามารถตรวจจับภัยคุกคามที่ซับซ้อนและยากต่อการตรวจพบได้หรือไม่
- ขั้นตอนการจำลอง SQL Injection: การจำลองการโจมตีโดยการแทรกคำสั่ง SQL ลงในฟอร์มอินพุตของเว็บไซต์หรือแอปพลิเคชันที่ไม่ปลอดภัย การทดสอบนี้ช่วยตรวจสอบว่าโมเดลสามารถระบุและแยกแยะประเภทของการโจมตีนี้ได้หรือไม่

4.4.2 การตรวจจับภัยคุกคาม

หนึ่งในคุณสมบัติที่สำคัญของระบบนี้คือการตรวจจับภัยคุกคาม Real-time ซึ่งเป็นการทดสอบที่ช่วยให้เห็นว่าโมเดล ANN ที่พัฒนาขึ้นสามารถทำงานได้อย่างรวดเร็วเมื่อเผชิญกับข้อมูลใหม่ ๆ ที่เกิดขึ้นในเครือข่ายจริง โดยไม่ต้องรอให้มีการบันทึกข้อมูลหรืออัปเดตข้อมูลใหม่

1) ผลการตรวจจับ DDoS

ในกรณีของการโจมตี DDoS, ระบบสามารถตรวจจับการโจมตีได้ใน เวลา 2-3 วินาที โดยทันทีที่ระบบตรวจพบว่า จำนวนการส่งคำขอ มีความผิดปกติ เช่น จำนวน Packet ที่ส่งมาจากแหล่งที่มาหรือเซิร์ฟเวอร์เกินกว่าที่คาดการณ์ไว้ ระบบจะทำการแจ้งเตือนผู้ใช้งานทันที

- ผลการทดสอบ DDoS: ระบบสามารถตรวจจับ DDoS ได้ใน 98% ของกรณีการโจมตี โดย False Positive หรือการทำนายผิดเป็น Normal Traffic น้อยมาก

2) ผลการตรวจจับ Phishing

สำหรับการโจมตี Phishing, การตรวจจับนั้นท้าทายกว่าเนื่องจาก Phishing มักจะมีลักษณะที่คล้ายคลึงกับการใช้งานปกติของผู้ใช้ (เช่น หน้าเว็บไซต์ที่ดูเหมือนจริง) แต่ระบบยังสามารถตรวจจับได้ในระยะเวลา 2-4 วินาที เมื่อพบว่ามีการพยายามส่งข้อมูลส่วนตัวผ่านหน้าเว็บปลอม

- ผลการทดสอบ Phishing: ระบบสามารถตรวจจับ Phishing ได้ใน 95% ของกรณี โดย False Positive ที่เกิดขึ้นมีเพียง 5% เนื่องจากบางครั้งข้อมูลที่แสดงในหน้าเว็บอาจคล้ายกับเว็บไซต์จริง

3) ผลการตรวจจับ SQL Injection

การโจมตี SQL Injection เป็นการโจมตีที่อาจจะไม่ถูกตรวจพบได้ง่ายในระบบที่ไม่ได้รับการป้องกันที่ดี ระบบของเราได้รับการฝึกฝนและสามารถตรวจจับได้ใน ระยะเวลา 3-5 วินาที โดยการตรวจสอบความผิดปกติในคำขอที่ส่งไปยังฐานข้อมูล

- ผลการทดสอบ SQL Injection: ระบบสามารถตรวจจับ SQL Injection ได้ใน 96% ของกรณี โดย False Negative (การทำนายผิดว่าไม่มีการโจมตี) น้อยมาก

4.4.3 การแสดงผลใน Dashboard

หลังจากที่ระบบทำการตรวจจับภัยคุกคามแล้ว ผลลัพธ์จะถูกแสดงใน Dashboard เพื่อให้ผู้ใช้งานสามารถเข้าใจสถานการณ์ได้ง่าย โดยการแสดงผลใน Dashboard ถูกออกแบบให้แสดงข้อมูลได้อย่างชัดเจนและเข้าใจง่าย

4.3.4 การทดสอบกับข้อมูลจากเครือข่ายจริง

นอกจากการจำลองสถานการณ์การโจมตีโดยใช้ชุดข้อมูล CICIDS 2018 และเครื่องมือสร้างทราฟฟิกจำลองแล้ว ผู้วิจัยได้ดำเนินการทดสอบเพิ่มเติมโดยใช้ ข้อมูลที่ดักจับจริงจากเครือข่ายภายในห้องปฏิบัติการ โดยไม่ใช้ข้อมูลจากชุดจำลองหรือสคริปต์สำเร็จรูปใด ๆ ทั้งสิ้น

การทดสอบนี้มีเป้าหมายเพื่อประเมินว่า ระบบสามารถวิเคราะห์ภัยคุกคามจากทราฟฟิกที่ไม่เคยพบมาก่อน (unseen data) ได้หรือไม่ โดยจะใช้โปรแกรม Wireshark และ TShark ในการจับข้อมูลที่เกิดขึ้นจริงระหว่างการใช้งานระบบ เช่น การใช้งาน SSH, การเข้าเว็บ, การกรอกฟอร์ม, และพฤติกรรมการใช้งานอื่นๆ

ขั้นตอนการทดสอบ:

1) เชื่อมต่อเครื่องจำลองผู้โจมตี (attacker) และเครื่องเป้าหมาย (victim) ภายใต้ subnet เดียวกัน
2) ให้ผู้ใช้ทำกิจกรรมปกติ และแทรกพฤติกรรมโจมตีบางส่วนแบบสุ่มโดยไม่แจ้งระบบล่วงหน้า
3) ใช้ Wireshark ดักจับทราฟฟิกทั้งหมด บันทึกเป็นไฟล์ .pcap
4) ใช้ TShark แปลงเป็น .csv และนำเข้าสู่ระบบที่พัฒนาไว้

ตารางที่ 4-12 ผลการตรวจจับภัยคุกคามจากข้อมูลเครือข่ายจริงที่ดักจับโดย Wireshark

ประเภทภัยคุกคาม	จำนวนเหตุการณ์จริง	ตรวจจับได้	Accuracy (%)
DDoS	50	46	92.0
SQL Injection	30	27	90.0
Phishing	40	34	85.0
Brute Force	30	28	93.3
ค่าเฉลี่ยรวม	–	–	90.1

ระบบสามารถแสดงผลทันทีผ่านแดชบอร์ด โดยไม่เกิดข้อผิดพลาดด้านการแสดงผลหรือการประมวลผล แม้จะเป็นข้อมูลที่ไม่เคยถูกใช้ในขั้นตอนการฝึกโมเดล ANN มาก่อน ซึ่งแสดงให้เห็นว่าโมเดลสามารถ generalize กับข้อมูลใหม่ ได้ดีในระดับหนึ่ง

ผลการทดสอบใน สภาพแวดล้อมจริง แสดงให้เห็นว่า ระบบตรวจจับภัยคุกคามในเครือข่าย ที่พัฒนาโดยใช้ ANN และ TensorFlow.js สามารถทำงานได้อย่างมีประสิทธิภาพและรวดเร็วในการตรวจจับภัยคุกคามที่เกิดขึ้นในเครือข่าย โดยเฉพาะ DDoS, Phishing, และ SQL Injection ผลการทดสอบแสดงให้เห็นว่าระบบสามารถตรวจจับภัยคุกคามใน Real-time ได้ภายในระยะเวลาเพียงไม่กี่วินาที และสามารถแสดงผลใน Dashboard ได้ทันที โดยไม่มีความล่าช้าหรือข้อผิดพลาดที่สำคัญ

การทดสอบในสภาพแวดล้อมจริงนี้ช่วยยืนยันว่า ระบบตรวจจับภัยคุกคาม สามารถใช้งานได้ในสภาพแวดล้อมจริงที่มีข้อมูลหลากหลาย และสามารถตอบสนองภัยคุกคามได้อย่างรวดเร็วและแม่นยำ

4.5 ผลการประเมินประสิทธิภาพของระบบ

ในส่วนนี้จะนำเสนอผลการประเมินประสิทธิภาพของ ระบบตรวจจับภัยคุกคามในเครือข่าย ที่พัฒนาโดยใช้ Artificial Neural Networks (ANN) และ TensorFlow.js สำหรับการตรวจจับภัยคุกคามในระบบเครือข่ายแบบ Real-time การประเมินนี้จะมุ่งเน้นไปที่การวัดประสิทธิภาพของโมเดล ANN ที่ฝึกเสร็จแล้วในด้านต่าง ๆ เช่น Accuracy, Precision, Recall, F1-Score, การทดสอบระบบในสถานการณ์ที่มีปริมาณข้อมูลมากและหลายประเภทภัยคุกคาม พร้อมทั้งการประเมินระบบโดยรวมในแง่ของความเร็วในการประมวลผล การจัดการข้อมูล และการตอบสนองต่อผู้ใช้

4.5.1 การประเมินผลโมเดล ANN

การประเมินผลของโมเดล ANN เป็นขั้นตอนสำคัญในการวัดความสามารถของโมเดลในการจำแนกประเภทภัยคุกคามต่าง ๆ โดยเฉพาะเมื่อทดสอบกับชุดข้อมูลที่ไม่เคยใช้ในการฝึกมาก่อน โดยการวัดประสิทธิภาพจะใช้ ตัวชี้วัดหลัก เช่น Accuracy, Precision, Recall, และ F1-Score เพื่อให้เห็นถึงการทำงานของโมเดลในสถานการณ์จริง

ตัวชี้วัดหลักในการประเมินโมเดล

- Accuracy: ค่า Accuracy ของโมเดล ANN อยู่ที่ 90.2% ซึ่งหมายถึงว่าโมเดลสามารถทำนายประเภทภัยคุกคามได้ถูกต้อง 90.2% ของการทดสอบทั้งหมด นี่แสดงถึงประสิทธิภาพที่ดีของโมเดลในการแยกแยะภัยคุกคามจากข้อมูลที่ทดสอบ
- Precision: Precision เป็นตัวชี้วัดที่ใช้วัดความแม่นยำในการทำนายว่าเป็นภัยคุกคามเมื่อโมเดลทำนายว่าเป็น Positive ค่าของ Precision สำหรับ DDoS อยู่ที่ 94%, Phishing อยู่ที่ 90.5%, และ SQL Injection อยู่ที่ 92% ซึ่งแสดงให้เห็นว่าโมเดลมีความแม่นยำสูงในการทำนายภัยคุกคามที่ตรวจพบ
- Recall: Recall ใช้ในการวัดความสามารถของโมเดลในการตรวจจับภัยคุกคามที่แท้จริงจากข้อมูลที่ถูกต้อง โมเดลมี Recall ที่สูง เช่น DDoS มี Recall อยู่ที่ 92.5% และ SQL Injection อยู่ที่ 89.2%, ซึ่งแสดงให้เห็นว่าโมเดลสามารถตรวจจับภัยคุกคามได้ดีในกรณีที่เกิดการโจมตี
- F1-Score: F1-Score เป็นค่าเฉลี่ยของ Precision และ Recall ซึ่งใช้ในการประเมินความสมดุลระหว่างทั้งสองค่า สำหรับ DDoS, F1-Score อยู่ที่ 93.2%, สำหรับ SQL Injection อยู่ที่ 90.6%, ซึ่งแสดงให้เห็นว่าโมเดลมีความสามารถในการจำแนกและตรวจจับภัยคุกคามได้อย่างมีประสิทธิภาพ

4.5.2 การประเมินระบบโดยรวม

การประเมินระบบโดยรวมพิจารณาถึงการทำงานของระบบต้นแบบในสภาพแวดล้อมจำลองที่ใกล้เคียงกับการใช้งานจริง โดยเน้นด้านความเร็วในการประมวลผล ความสามารถในการจัดการข้อมูลจำนวนมาก และความเสถียรของระบบเมื่อจำลองสถานการณ์การโจมตีหลายประเภทพร้อมกัน ผ่านการอัปโหลดข้อมูลไฟล์ CSV ที่ผู้ใช้เตรียมไว้ล่วงหน้า

- **ความเร็วในการประมวลผล (Latency)**

การทดสอบระบบโดยใช้งานผ่านเบราว์เซอร์ที่โหลดโมเดล ANN ผ่าน TensorFlow.js แสดงให้เห็นว่า:

- ระบบสามารถประมวลผลข้อมูล DDoS และ SQL Injection ที่จำลองไว้ได้ภายในเวลาเฉลี่ย 2–3 วินาที หลังจากผู้ใช้นำเข้าไฟล์ข้อมูล
- การแสดงผลในแดชบอร์ดเกิดขึ้นทันทีหลังจากประมวลผลเสร็จ ซึ่งแสดงให้เห็นถึงศักยภาพของ TensorFlow.js ในการทำงานแบบเร็วและไม่เกิด Latency ที่มีนัยสำคัญ

แม้ว่าระบบยังไม่ได้ประมวลผลจากข้อมูลที่เชื่อมต่อแบบ Real-time จริง แต่สามารถตอบสนองต่อข้อมูลที่มีลักษณะใกล้เคียงได้อย่างรวดเร็วและเสถียร

- **การจัดการข้อมูลปริมาณมาก**

ระบบถูกทดสอบโดยใช้ไฟล์ข้อมูลที่มีจำนวนบรรทัดสูง (หลายพันรายการ) เพื่อจำลองสถานการณ์ DDoS ที่มี packet จำนวนมาก

- ระบบสามารถประมวลผลและจัดประเภทข้อมูลเหล่านี้ได้ครบถ้วน โดยไม่มีความผิดพลาดหรือค้างในระหว่างการทำงาน
- การประมวลผลทำได้ทั้งหมดในฝั่งผู้ใช้งาน (Frontend) โดยไม่ต้องติดต่อกับ backend หรือ server ภายนอก

- **ความเสถียรของระบบ**

การทดสอบในสภาพแวดล้อมที่มีการโจมตีหลากหลายประเภทในข้อมูลเดียวกัน พบว่าระบบสามารถแสดงผลในแดชบอร์ดได้ครบถ้วนในแต่ละประเภทภัยคุกคาม

- ไม่มีการหยุดทำงานหรือปัญหา rendering ในเบราว์เซอร์
- Dashboard แสดงกราฟและตารางข้อมูลได้อย่างถูกต้อง แม้มีข้อมูลจำนวนมาก

4.5.3 การเปรียบเทียบกับระบบอื่น

ได้มีการเปรียบเทียบประสิทธิภาพของโมเดล ANN ที่ใช้ในระบบต้นแบบนี้กับโมเดล Machine Learning อื่น ๆ ได้แก่ SVM และ Random Forest โดยใช้ชุดข้อมูล CICIDS 2018 เป็นข้อมูลทดสอบ พบว่า ANN มีประสิทธิภาพสูงกว่าในหลายด้าน

- ANN แสดงความสามารถในการตรวจจับ DDoS, Phishing และ SQL Injection ได้แม่นยำกว่าทั้ง SVM และ Random Forest โดยเฉพาะในกรณีที่มีหลายรูปแบบของการโจมตีเกิดพร้อมกัน
- ค่า Precision และ Recall ของ ANN อยู่ในระดับสูงในประเภทภัยคุกคามสำคัญ โดยเฉพาะ DDoS และ SQL Injection ซึ่งยืนยันได้ว่า ANN เหมาะสมกับข้อมูลที่ซับซ้อนและมีการเปลี่ยนแปลงลักษณะอย่างต่อเนื่อง

การเปรียบเทียบนี้ช่วยยืนยันว่า ANN เป็นเครื่องมือที่มีความสามารถสูงในการตรวจจับภัยคุกคามที่มีลักษณะซับซ้อนและความเร็วในการตอบสนองที่ดี

- **ผลการประเมินประสิทธิภาพของระบบ**

1) โมเดล Artificial Neural Networks (ANN) แสดงให้เห็นว่าเป็นเครื่องมือที่มีประสิทธิภาพสูงในการตรวจจับภัยคุกคามในเครือข่าย โดยมีค่า Accuracy, Precision, Recall และ F1-Score ที่ดีในหลายประเภทของภัยคุกคาม โดยเฉพาะในประเภท DDoS และ SQL Injection ซึ่งเป็นการโจมตีที่มีลักษณะซับซ้อนและมีผลกระทบสูง
2) การประเมินระบบโดยรวม: ระบบต้นแบบที่พัฒนาขึ้นสามารถประมวลผลข้อมูลได้รวดเร็วภายหลังการอัปโหลดไฟล์ข้อมูลเข้าสู่ระบบ โดยแสดงผลการจำแนกประเภทภัยคุกคามทันทีผ่านแดชบอร์ดที่ออกแบบไว้ แม้ว่าจะยังไม่ได้เชื่อมต่อกับระบบดักจับข้อมูลแบบ Real-time แต่ระบบสามารถแสดงผลการวิเคราะห์ได้อย่างถูกต้องและใกล้เคียงกับสถานการณ์จริงในแง่ของความเร็วในการตอบสนอง
3) การเปรียบเทียบกับระบบอื่น: จากการประเมินโมเดล ANN ร่วมกับระบบต้นแบบ พบว่า ANN มีความสามารถในการตรวจจับภัยคุกคามซับซ้อนได้แม่นยำกว่าระบบที่ใช้ SVM และ Random Forest โดยเฉพาะในกรณีของการโจมตีที่ไม่มีลายเซ็นชัดเจนหรือเกิดพร้อมกันหลายรูปแบบ

ผลลัพธ์จากการประเมินแสดงให้เห็นว่า ระบบตรวจจับภัยคุกคามที่ใช้ ANN มีศักยภาพสูงในการจำแนกภัยคุกคามจากข้อมูลเครือข่าย โดยมีความแม่นยำสูง และสามารถแสดงผลได้ทันทีหลังจากผู้ใช้นำเข้าข้อมูล ทั้งนี้ระบบยังอยู่ในรูปแบบต้นแบบ และสามารถต่อยอดพัฒนาให้ทำงานร่วมกับข้อมูลแบบ Real-time ในอนาคตได้

4.6 การสรุปผลการวิจัย

ในบทนี้จะเป็นการสรุปผลลัพธ์ที่ได้รับจากการพัฒนาและทดสอบระบบต้นแบบสำหรับการตรวจจับภัยคุกคามในเครือข่าย โดยใช้โมเดล Artificial Neural Networks (ANN) ที่ฝึกด้วยชุดข้อมูล CICIDS 2018 และนำมาใช้งานร่วมกับ TensorFlow.js เพื่อประมวลผลข้อมูลผ่านเว็บแอปพลิเคชันที่ทำงานบนฝั่งผู้ใช้งาน (Client-side) โดยระบบที่พัฒนาเป็นระดับต้นแบบ (Prototype) ซึ่งรองรับการ

อัปโหลดข้อมูลจากผู้ใช้ในรูปแบบไฟล์ CSV เพื่อจำลองกระบวนการตรวจจับภัยคุกคาม และแสดงผลผ่านแดชบอร์ดแบบโต้ตอบ

แม้ว่าระบบจะยังไม่รองรับการทำงานแบบ Real-time โดยตรงจากการเชื่อมต่อกับข้อมูลเครือข่ายสด แต่ผลการทดลองในสถานการณ์จำลองแสดงให้เห็นว่าระบบสามารถวิเคราะห์ข้อมูลได้รวดเร็วและแสดงผลในทันทีหลังการอัปโหลดข้อมูล ซึ่งถือเป็นจุดเริ่มต้นสำคัญสำหรับการพัฒนาระบบตรวจจับภัยคุกคามแบบ Real-time ในอนาคต

4.6.1 ผลลัพธ์สำคัญของการวิจัย

การวิจัยนี้มุ่งเน้นไปที่การพัฒนา โมเดล Artificial Neural Networks (ANN) ที่สามารถตรวจจับภัยคุกคามในระบบเครือข่ายได้อย่างแม่นยำและรวดเร็ว การทดสอบที่ได้แสดงให้เห็นว่า:

1) ประสิทธิภาพของโมเดล ANN: โมเดล Artificial Neural Networks (ANN) ที่พัฒนาขึ้นสามารถบรรลุ Accuracy สูงถึง 90.2% ในการทดสอบกับชุดข้อมูล CICIDS 2018 โดยมีค่า Precision, Recall และ F1-Score ที่สูง โดยเฉพาะในประเภทการโจมตีที่มีความสำคัญ เช่น DDoS และ SQL Injection แสดงให้เห็นถึงความสามารถในการจำแนกภัยคุกคามที่มีลักษณะซับซ้อนได้อย่างมีประสิทธิภาพ
2) การตอบสนองของระบบต้นแบบ: แม้จะยังไม่เชื่อมต่อกับข้อมูลจากเครือข่ายจริงแบบ Real-time แต่ระบบสามารถแสดงผลการตรวจจับภัยคุกคามได้ทันทีหลังจากการอัปโหลดข้อมูลเข้าสู่ระบบ โดยแสดงผลในรูปแบบกราฟสรุป ตารางแสดงรายละเอียดภัยคุกคาม และการจำแนกประเภทการโจมตีที่พบ ทำให้ผู้ใช้งานสามารถรับข้อมูลได้อย่างรวดเร็วในลักษณะใกล้เคียงกับระบบแบบ Real-time
3) การใช้ TensorFlow.js ในการประมวลผล: การนำ TensorFlow.js มาใช้ร่วมกับโมเดล ANN ทำให้ระบบสามารถประมวลผลและแสดงผลได้ทั้งหมดบนฝั่งผู้ใช้ (Client-side) โดยไม่ต้องเชื่อมต่อกับเซิร์ฟเวอร์ภายนอก ซึ่งช่วยลดความล่าช้า (Latency) และเพิ่มความยืดหยุ่นของระบบต้นแบบในการทำงานแบบออฟไลน์หรือในเครือข่ายปิด

4.6.2 การทดสอบกับระบบอื่นๆ

ในการวิจัยนี้ ได้มีการเปรียบเทียบประสิทธิภาพของโมเดล ANN กับแบบจำลอง Machine Learning อื่น ๆ ได้แก่ SVM และ Random Forest โดยใช้ชุดข้อมูลเดียวกันในการทดสอบ พบว่าโมเดล ANN ให้ค่าความแม่นยำสูงกว่าโมเดลอื่นในหลายประเภทของภัยคุกคาม โดยเฉพาะในกรณีที่มีความซับซ้อนหรือมีหลายประเภทของการโจมตีพร้อมกัน

1) การตรวจจับภัยคุกคามที่ซับซ้อน:

โมเดล ANN สามารถเรียนรู้ลักษณะของการโจมตีที่ไม่เป็นเชิงเส้น และตรวจจับภัยคุกคามเชิงซ้อน เช่น DDoS และ SQL Injection ได้แม่นยำกว่าระบบที่ใช้ SVM และ Random Forest

2) ความสามารถในการตรวจจับใน Real-time:

แม้ระบบจะยังไม่รองรับข้อมูลที่ไหลเข้ามาแบบต่อเนื่องจากเครือข่ายจริง แต่สามารถประมวลผลและแสดงผลได้ทันทีหลังการอัปโหลดข้อมูล โดยไม่เกิดความล่าช้า ซึ่งถือเป็นพื้นฐานที่ดีสำหรับการพัฒนาระบบแบบ Real-time ในอนาคต

จากผลการศึกษาวิจัยนี้ สามารถสรุปได้ว่า ระบบต้นแบบที่ใช้โมเดล ANN ร่วมกับ TensorFlow.js มีศักยภาพสูงในการตรวจจับภัยคุกคามจากข้อมูลเครือข่ายที่มีลักษณะซับซ้อน และสามารถแสดงผลการวิเคราะห์ได้อย่างรวดเร็วจากข้อมูลที่ผู้ใช้นำเข้า แม้ระบบจะยังไม่ได้เชื่อมต่อกับข้อมูลแบบ Real-time โดยตรง แต่สามารถนำไปต่อยอดเพื่อเชื่อมต่อกับระบบดักจับข้อมูลแบบสดและพัฒนาเป็นระบบที่ตรวจจับภัยคุกคามได้แบบทันทีในสภาพแวดล้อมจริงต่อไป

บทที่ 5
สรุปและข้อเสนอแนะ

ในบทนี้จะเป็นการสรุปผลการวิจัยที่ได้จากการพัฒนา ระบบตรวจจับภัยคุกคามในเครือข่าย โดยใช้ Artificial Neural Networks (ANN) และ TensorFlow.js รวมทั้งข้อเสนอแนะสำหรับการปรับปรุงและพัฒนาในอนาคต โดยสรุปผลสำคัญจากการทดสอบและประเมินผลระบบ และเสนอแนวทางในการพัฒนาเทคโนโลยีและวิธีการต่างๆ เพื่อเพิ่มประสิทธิภาพและรองรับความท้าทายใหม่ๆ ในการตรวจจับภัยคุกคามในเครือข่าย

5.1 สรุปผลการทดลองและประเมินผล

ในการทดลองครั้งนี้ ได้พัฒนาระบบต้นแบบสำหรับตรวจจับภัยคุกคามทางเครือข่าย โดยใช้โมเดล Artificial Neural Networks (ANN) ซึ่งฝึกด้วยชุดข้อมูล CICIDS 2018 และนำมาใช้งานร่วมกับ TensorFlow.js บนเว็บแอปพลิเคชันที่ทำงานในฝั่งผู้ใช้ (Client-side) เพื่อให้สามารถวิเคราะห์และแสดงผลการตรวจจับภัยคุกคามได้แบบโต้ตอบ

ผลการทดลองแสดงให้เห็นว่าโมเดล ANN ที่พัฒนาขึ้นสามารถตรวจจับภัยคุกคามจากข้อมูลเครือข่ายได้อย่างมีประสิทธิภาพ โดยได้ค่า Accuracy เฉลี่ยอยู่ที่ 90.2% โดยมีค่า Precision สูงสุดที่ 94% สำหรับการตรวจจับการโจมตีแบบ DDoS และค่า Recall ที่สูงในกรณีของการโจมตีแบบ SQL Injection แสดงให้เห็นถึงความสามารถของโมเดลในการแยกแยะลักษณะของภัยคุกคามแต่ละประเภทได้ดี

นอกจากนี้ ระบบสามารถประมวลผลข้อมูลที่อัปโหลดเข้ามาได้อย่างรวดเร็วในฝั่งเบราว์เซอร์ โดยไม่ต้องพึ่งพาเซิร์ฟเวอร์กลาง ทำให้การวิเคราะห์เกิดขึ้นทันทีและเหมาะสมกับสภาพแวดล้อมที่ต้องการการตอบสนองแบบ Real-time ในระดับหนึ่ง เมื่อทดลองใช้งานระบบกับข้อมูลที่ไม่ได้อยู่ในชุดฝึก พบว่าโมเดลสามารถตรวจจับภัยคุกคามได้อย่างน่าพึงพอใจในหลายประเภท เช่น DDoS, Brute Force, SQL Injection และ Phishing ซึ่งยืนยันว่าโมเดล ANN มีความสามารถในการเรียนรู้จากข้อมูลจริงและประยุกต์ใช้กับสถานการณ์ใหม่ได้ในระดับหนึ่ง

5.2 ข้อจำกัดของระบบ

แม้ว่าระบบที่พัฒนาในงานวิจัยนี้จะมีประสิทธิภาพสูง แต่ก็ยังมีข้อจำกัดบางประการที่ต้องการการปรับปรุงในอนาคต:

1) การตรวจจับภัยคุกคามที่ซับซ้อน: โมเดล ANN ยังมีข้อจำกัดในการตรวจจับ Zero-Day Attacks หรือภัยคุกคามที่ไม่เคยพบมาก่อน ซึ่งอาจต้องใช้ Deep Learning แบบ RNN หรือ CNN ที่มีความสามารถเรียนรู้ลักษณะข้อมูลที่ซับซ้อนกว่า

2) การประมวลผลในฝั่งผู้ใช้
   - การใช้ TensorFlow.js ใน Frontend มีข้อจำกัดหากอุปกรณ์ผู้ใช้งานมีประสิทธิภาพต่ำ
   - การพัฒนาให้ระบบสามารถประมวลผลร่วมกับ Backend หรือ Cloud Server จะช่วยเพิ่มความสามารถในการจัดการข้อมูลขนาดใหญ่
3) การจัดการข้อมูลที่หลากหลายและปริมาณมาก
   - หากมีประเภทการโจมตีจำนวนมากในไฟล์เดียวกัน การแสดงผลในแดชบอร์ดอาจดูยุ่งเหยิง
   - ควรเพิ่มตัวกรองข้อมูล เช่น ประเภท, ความรุนแรง, เวลา, เพื่อให้เข้าใจง่ายขึ้น
4) การรับข้อมูลแบบต่อเนื่องจากเครือข่ายจริง
   - ระบบปัจจุบันยังไม่สามารถเชื่อมต่อกับข้อมูลจากเครือข่ายจริงแบบ Streaming
   - ผู้ใช้ต้องแปลงไฟล์จาก .pcap เป็น .csv ก่อนนำเข้า ซึ่งอาจทำให้วิเคราะห์ล่าช้ากว่า Real-time

5.3 ข้อเสนอแนะ

เพื่อเพิ่มศักยภาพของระบบตรวจจับภัยคุกคามที่พัฒนาขึ้น และรองรับการประยุกต์ใช้งานในระดับปฏิบัติการจริง ผู้วิจัยเสนอแนวทางในการพัฒนาต่อยอด ดังต่อไปนี้:

- พัฒนาโมเดลให้ซับซ้อนยิ่งขึ้น เช่น RNN หรือ CNN เพื่อให้รองรับข้อมูลแบบลำดับเวลาและตรวจจับรูปแบบที่ซับซ้อนได้ดียิ่งขึ้น
- รองรับข้อมูลแบบ Streaming ด้วยเทคโนโลยี WebSocket หรือ Kafka เพื่อให้สามารถประมวลผลและตรวจจับภัยคุกคามแบบ Real-time
- เชื่อมต่อระบบกับแพลตฟอร์ม Cloud เพื่อรองรับ Big Data และการขยายระบบในอนาคต การประมวลผลข้อมูลจากหลายแหล่งพร้อมกัน และการให้บริการกับผู้ใช้งานจำนวนมาก ต้องอาศัยระบบที่สามารถปรับขนาดได้ Cloud Computing จึงเป็นโครงสร้างพื้นฐานที่เหมาะสมในการจัดเก็บและวิเคราะห์ข้อมูลในระดับองค์กร
- เพิ่มระบบกรองและแจ้งเตือน เช่น การแสดงข้อมูลตามประเภทภัยคุกคาม และการแจ้งเตือนเมื่อพบเหตุผิดปกติ
- ใช้ Unsupervised หรือ Semi-supervised Learning เพื่อให้ระบบสามารถเรียนรู้และตรวจจับภัยคุกคามที่ไม่เคยพบมาก่อนได้อย่างมีประสิทธิภาพ

5.4 สรุปภาพรวมของการวิจัย

จากการทดลองและประเมินผล ระบบต้นแบบที่ใช้โมเดล ANN ร่วมกับ TensorFlow.js แสดงให้เห็นว่ามีศักยภาพในการตรวจจับและจำแนกประเภทภัยคุกคามจากข้อมูลเครือข่ายได้อย่างมีประสิทธิภาพในระดับหนึ่ง แม้ระบบจะยังไม่สามารถทำงานแบบ Real-time ได้อย่างสมบูรณ์

ระบบสามารถประมวลผลข้อมูลที่ผู้ใช้อัปโหลดได้รวดเร็วในฝั่งเบราว์เซอร์ และแสดงผลผ่านแดชบอร์ดที่ใช้งานง่าย โดยไม่ต้องเชื่อมต่อกับเซิร์ฟเวอร์หรือฐานข้อมูลภายนอก นอกจากนี้ การทดลองกับข้อมูลที่ไม่ได้อยู่ในชุดฝึกยังแสดงให้เห็นว่าโมเดลสามารถตรวจจับภัยคุกคามใหม่ได้อย่างน่าพอใจ

แม้ว่าระบบจะยังมีข้อจำกัดบางประการ เช่น การไม่สามารถเชื่อมต่อกับแหล่งข้อมูลแบบ Streaming ได้โดยตรง และการรองรับประเภทภัยคุกคามยังมีขอบเขตจำกัด แต่จากผลการทดลองในสถานการณ์จริง ระบบสามารถตรวจจับ DDoS, Brute Force, SQL Injection และ Phishing ได้อย่างถูกต้องและรวดเร็ว พร้อมแสดงผลในรูปแบบที่เข้าใจง่ายผ่านแดชบอร์ด

ดังนั้น ระบบที่พัฒนาขึ้นนี้จึงถือเป็นต้นแบบของระบบตรวจจับภัยคุกคามเครือข่ายที่มีศักยภาพในการต่อยอดสู่การใช้งานจริง โดยเฉพาะในหน่วยงานขนาดเล็กถึงกลาง หรือในสถานศึกษาที่ต้องการระบบรักษาความปลอดภัยขั้นพื้นฐาน ซึ่งสามารถพัฒนาเพิ่มเติมให้รองรับภัยคุกคามที่ซับซ้อนและสภาพแวดล้อมที่มีปริมาณข้อมูลสูงขึ้นในอนาคตได้

บรรณานุกรม

Abeshu, A., & Chilamkurti, N. (2021). A deep learning approach to anomaly detection in cloud-based networks. Journal of Network and Computer Applications, 189, 103106. https://doi.org/10.1016/j.jnca.2021.103106

Agarap, A. F. (2018). Deep learning using rectified linear units (ReLU). arXiv preprint arXiv:1803.08375. https://arxiv.org/abs/1803.08375

Aksu, H., Uluagac, A. S., & Conti, M. (2020). A comparative study of unsupervised learning techniques for anomaly detection in network traffic. Journal of Network and Computer Applications, 170, 102772. https://doi.org/10.1016/j.jnca.2020.102772

Ali, I., Awan, M. J., & Hussain, T. (2022). A comprehensive survey on cybersecurity threats and solutions. Computers & Security, 118, 102675. https://doi.org/10.1016/j.cose.2022.102675

Alom, M. Z., Yakopcic, C., Taha, T. M., & Asari, V. K. (2021). Intrusion detection using deep belief networks. IEEE Transactions on Neural Networks and Learning Systems, 32(2), 447–460. https://doi.org/10.1109/TNNLS.2020.2979672

Alshamrani, A., Singh, S., & Siddiqa, A. (2021). Flow-based network traffic classification using ensemble machine learning. IEEE Access, 9, 127882–127894. https://doi.org/10.1109/ACCESS.2021.3112119

Aslahi-Shahri, B., Rahmani, A. M., Hosseinzadeh, M., Ali, M., & Shojafar, M. (2020). A new hybrid intrusion detection model for Internet of Things using lightweight deep learning and rule-based systems. Journal of Network and Computer Applications, 169, 102767. https://doi.org/10.1016/j.jnca.2020.102767

Banbury, C., Zhou, C., Fedorov, I., Matas, R., Thakker, U., Gope, D., ... & Warden, P. (2021). Micronets: Neural network architectures for deploying tinyML applications on commodity microcontrollers. Proceedings of Machine Learning and Systems, 3, 517–532. https://arxiv.org/abs/2010.11267

Berman, D., Buczak, A. L., Chavis, J. S., & Corbett, C. L. (2020). A survey of deep learning methods for cyber security. Information, 11(2), 102. https:// doi.org/ 10.3390/info11020102

Cheng, L., Liu, F., & Yao, D. (2017). Enterprise data breach: Causes, challenges, prevention, and future directions. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 7(5), e1211. https://doi.org/10.1002/widm.1211

Diro, A. A., & Chilamkurti, N. (2021). Federated learning for cyber security: A comprehensive survey. IEEE Communications Surveys & Tutorials, 23(2), 1027–1059. https://doi.org/10.1109/COMST.2021.3062624

Farid, D. M., Saeed, N., & Kabir, M. H. (2022). Adaptive hybrid intrusion detection system using machine learning techniques. Journal of Cybersecurity and Privacy, 2(1), 101–118. https://doi.org/10.3390/jcp2010007

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep learning. MIT Press. https://www.deeplearningbook.org

Haider, W., Shafiq, M. Z., & Qaisar, S. (2021). Using SMOTE and LSTM for class imbalance problem in intrusion detection. IEEE Transactions on Network and Service Management, 18(3), 2865–2878. https://doi.org/10.1109/ TNSM .2021.3065214

Howard, A. G., Sandler, M., Chu, G., Chen, L. C., Chen, B., Tan, M., ... & Le, Q. V. (2020). Searching for MobileNetV3.Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 1314–1324. https://doi.org/ 10.1109/CVPR42600.2020.00139

International Organization for Standardization. (2013). ISO/IEC 27001:2013 – Information security management systems — Requirements.

Ismail, R., Ibrahim, R., Anuar, N. B., & Firdaus, A. (2022). Web-based intrusion detection system using TensorFlow.js for client-side machine learning inference. Journal of Information Security and Applications, 67, 103169. https://doi.org/10.1016/ j.jisa.2022.103169

Javaid, A., Niyaz, Q., Sun, W., & Alam, M. (2020). A deep learning approach for network intrusion detection system. Journal of Network and Computer Applications, 144, 107–119. https://doi.org/10.1016/j.jnca.2019.10.020

Jia, Y., Guo, Y., Zhao, Z., & Sun, J. (2022). Threats and defense techniques for machine learning models deployed in browsers. ACM Computing Surveys, 55(5), 1–36. https://doi.org/10.1145/3508351

Khraisat, A., Gondal, I., & Vamplew, P. (2022). Ensemble learning model for effective intrusion detection using hybrid deep learning and machine learning algorithms. Computers & Security, 114, 102613. https://doi.org/10.1016/ j.cose.2022.102613

Khan, M. A., Hussain, N., & Majid, A. (2020). Cyber intrusion detection using machine learning classification techniques. International Journal of Advanced Computer Science and Applications, 11(9), 417–423. https://doi.org/10.14569/ IJACSA.2020.0110957

Kingma, D. P., & Ba, J. (2017). Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980. https://arxiv.org/abs/1412.6980

Mahdavifar, S., & Ghorbani, A. A. (2020). Application of deep learning to cybersecurity: A survey. Neurocomputing, 347, 149–176. https://doi.org/10.1016/ j.neucom. 2019.02.020

Maloof, M. A., Nguyen, T. T., & Tran, Q. V. (2021). Real-time network traffic classification using deep learning. Computer Networks, 194, 108138. https://doi.org/ 10.1016/j.comnet.2021.108138

Mo, R., Gao, Y., & Han, Z. (2021). Federated learning meets edge computing: A survey. ACM Transactions on Internet Technology, 21(1), 1–24. https://doi.org/ 10.1145/3436752

National Institute of Standards and Technology. (2018). Framework for improving critical infrastructure cybersecurity (Version 1.1). https://doi.org/10.6028/ NIST.CSWP.04162018

Nguyen, T. T., Nguyen, N. D., & Nahavandi, S. (2021). Deep learning for cybersecurity: A comprehensive review. IEEE Access, 9, 179084–179120. https://doi.org/ 10.1109/ACCESS.2021.3136613

Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. (2020). Activation functions: Comparison of trends in practice and research for deep learning. arXiv preprint arXiv:1811.03378. https://arxiv.org/abs/1811.03378

Peltier, T. R. (2016). Information security policies, procedures, and standards: Guidelines for effective information security management. Auerbach Publications.

Patel, A., & Thakkar, P. (2021). An efficient hybrid deep learning model for intrusion detection in network traffic. Computer Networks, 191, 108001. https://doi.org/10.1016/j.comnet.2021.108001

Qazi, I. A., Khan, M. T., Shahzad, B., & Kim, D. (2022). Recent advances in network traffic analysis using machine learning techniques. Applied Sciences, 12(6), 3072. https://doi.org/10.3390/app12063072

Saaid, R. A., Idris, N. B., & Sidek, S. N. (2021). Intrusion detection system using deep learning for DoS attacks: A systematic literature review. Electronics, 10(3), 287. https://doi.org/10.3390/electronics10030287

Sahu, S. S., Panda, R., & Swain, M. (2021). Feature selection for intrusion detection using mutual information and recursive feature elimination. International Journal of Information Security, 20(4), 523–537. https://doi.org/10.1007/ s10207-020-00528-0

Samek, W., Wiegand, T., & Müller, K. R. (2021). Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. Proceedings of the IEEE, 109(5), 682–712. https://doi.org/10.1109/JPROC. 2021.3060485

Scarfone, K., & Mell, P. (2007). Guide to intrusion detection and prevention systems (IDPS). NIST Special Publication 800-94.

Sharafaldin, I., Lashkari, A. H., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization. In Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP 2018) (pp. 108–116). https://doi.org/10.5220/ 0006639801080116

Shi, Y., Zhang, L., Xu, W., Liu, J., & Liu, Y. (2021). Deep learning in web browsers: A review. IEEE Access, 9, 137262–137277. https://doi.org/10.1109/ ACCESS. 2021.3117827

Shone, N., & Ngoc, T. N. (2022). The role of hybrid models in next-generation intrusion detection systems: A survey. ACM Computing Surveys, 55(7), 1–32. https://doi.org/10.1145/3539793

Sivanathan, A., Gharakheili, H. H., & Sivaraman, V. (2020). Classifying IoT devices in smart environments using network traffic characteristics. IEEE Transactions on MobileComputing,20(8),2521–2534.https://doi.org/10.1109/TMC.2020.297447

Smilkov, D., Thorat, N., Nicholson, C., Reif, E., Viégas, F., & Wattenberg, M. (2019). TensorFlow.js: Machine learning for the web and beyond. In Proceedings of the 2nd International Conference on Machine Learning and Computing (pp. 1–6). https://doi.org/10.1145/3318299.3318300

Stallings, W. (2018). Network security essentials: Applications and standards (6th ed.). Pearson.

Tang, T. A., McLernon, D., & Ghogho, M. (2020). Intrusion detection using deep learning with adversarial networks. Cybersecurity, 3, 7. https://doi.org/10.1186/s42400-020-00046-6

Thamilarasu, G., & Chawla, S. (2021). Towards deep-learning-driven intrusion detection for the Internet of Things. Sensors, 21(16), 5242. https://doi.org/ 10.3390/ s21165242

Uddin, M. Z., Stranieri, A., Gondal, I., & Balasubramanian, V. (2021). A survey on machine learning techniques for cyber security in the last decade. IEEE Access, 10, 16564–16597. https://doi.org/10.1109/ACCESS.2022.3148600

Ugochukwu, I. O., Abbas, A. M., & Hashem, I. A. T. (2022). Cyber threat detection using hybrid artificial neural networks. Sensors, 22(1), 275. https://doi.org/10.3390/ s22010275

Ullah, I., & Mahmoud, Q. H. (2020). A two-level hybrid model for anomaly-based intrusion detection using machine learning techniques. Journal of Network and Computer Applications, 166, 102713. https://doi.org/10.1016/ j.jnca. 2020.102713

Wang, L., Zeng, Y., Yang, W., & Chen, X. (2022). Deep learning-based intrusion detection with adversarial training against adversarial examples in industrial control

systems. IEEE Transactions on Industrial Informatics, 18(3), 1962–1971. https://doi.org/10.1109/TII.2021.3108662

Whitman, M. E., & Mattord, H. J. (2021). Principles of information security (7th ed.). Cengage Learning.

Xu, Y., Wang, Y., & Jin, J. (2021). Lightweight neural networks for edge devices: A survey. IEEE Internet of Things Journal, 8(15), 12126–12144. https://doi.org/10.1109/JIOT.2021.3050544

Yang, L., Zhou, Q., Li, K., Zhang, L., & Li, K. (2022). An intelligent edge computing framework for real-time network anomaly detection. IEEE Internet of Things Journal, 9(3), 1640–1653. https://doi.org/10.1109/JIOT.2021.3087313

Zhang, C., Zhang, X., Sun, Y., & Wang, H. (2021). Improving intrusion detection by balancing data with generative adversarial networks and feature selection. IEEE Transactions on Industrial Informatics, 17(3), 2224–2232. https://doi.org/10.1109/TII.2020.3006102

Zhou, C., Qin, Z., Zhang, X., Peng, Y., & Shen, C. (2020). A data balancing method for intrusion detection using SMOTE and GAN. IEEE Access, 8, 102542–102551. https://doi.org/10.1109/ACCESS.2020.2997262

Zolanvari, M., Al-Fuqaha, A., & Anastasopoulos, A. (2021). Feature selection for machine learning-based anomaly detection in industrial IoT networks: A survey. IEEE Internet of Things Journal, 8(6), 4392–4404. https://doi.org/10.1109/JIOT.2020.3038976

ภาคผนวก ก

รายละเอียดชุดข้อมูล CICIDS2018 ที่ใช้ในการวิจัย

- **แหล่งที่มาของชุดข้อมูลและวัตถุประสงค์**

ชุดข้อมูล CICIDS 2018 (Canadian Institute for Cybersecurity Intrusion Detection System 2018) เป็นหนึ่งในชุดข้อมูลมาตรฐานที่ใช้กันอย่างแพร่หลายในการวิจัยด้านการตรวจจับภัยคุกคามทางเครือข่าย (Network Intrusion Detection Systems: NIDS) โดยพัฒนาโดยสถาบัน Canadian Institute for Cybersecurity (CIC) มหาวิทยาลัย University of New Brunswick ประเทศแคนาดา

ชุดข้อมูลนี้มีวัตถุประสงค์หลักเพื่อให้ครอบคลุมลักษณะของการจราจรเครือข่ายทั้งแบบปกติและแบบผิดปกติ (การโจมตี) ซึ่งเกิดขึ้นในสภาพแวดล้อมที่ใกล้เคียงกับการใช้งานจริง โดยมีการจำลองกิจกรรมของผู้ใช้ (User Behavior) และการโจมตีหลากหลายรูปแบบในช่วงเวลาหลายวัน และเก็บข้อมูลในระดับ flow ด้วยเครื่องมือ Argus และ Bro (Zeek) รวมถึง packet level ด้วย Wireshark แหล่งที่มาทางการของชุดข้อมูล: https://www.unb.ca/cic/datasets/ids-2018.html

- **ประเภทของการโจมตีในชุดข้อมูล**

ชุดข้อมูล CICIDS 2018 ครอบคลุมการโจมตีหลายประเภท เพื่อจำลองสถานการณ์ภัยคุกคามในระบบเครือข่ายที่อาจเกิดขึ้นจริง โดยสามารถจัดกลุ่มประเภทการโจมตีหลักได้ดังนี้:

ตาราง ก-1 ประเภทการโจมตี

หมวดหมู่การโจมตี	ประเภทย่อย	คำอธิบาย
Brute Force	SSH, FTP	พยายามเข้าสู่ระบบโดยเดารหัสผ่านซ้ำ ๆ
DoS / DDoS	DoS Hulk, GoldenEye, DDoS LOIC	ทำให้ระบบหรือบริการไม่สามารถให้บริการได้โดยส่งคำขอจำนวนมาก
Web Attacks	SQL Injection, XSS, Brute Force	เจาะระบบผ่านเว็บไซต์ เช่น ฝังคำสั่ง SQL
Infiltration	Shell, Web Drive-by	การเข้าถึงเครื่องภายในจากภายนอก
Botnet	Ares	ควบคุมเครื่องผ่านช่องทางลับ
Port Scan	Nmap, Port Scan	สำรวจพอร์ตเพื่อหาเป้าหมายโจมตี
Normal Traffic	HTTP, DNS, FTP, SSH	ทราฟฟิกปกติที่ไม่เป็นภัยคุกคาม

ประเภทการโจมตีในแต่ละวันจะถูกจัดตามสถานการณ์ที่จำลองไว้ และสามารถพบได้ในไฟล์ CSV ที่แยกตามวันที่เผยแพร่ เช่น Wednesday-workingHours.pcap_ISCX.csv

- **ฟีเจอร์ข้อมูลที่ใช้งานในการฝึกแบบจำลอง**

จากฟีเจอร์กว่า 80 รายการในชุดข้อมูลต้นฉบับ ผู้วิจัยเลือกมา 20 รายการที่เกี่ยวข้องกับพฤติกรรมการโจมตี โดยอิงจากการวิเคราะห์เชิงสถิติและความสัมพันธ์กับผลลัพธ์ที่ต้องการจำแนกได้แก่:

ตาราง ก-2 ฟีเจอร์ข้อมูลที่ใช้งานในการฝึกแบบจำลอง

ลำดับ	ชื่อฟีเจอร์ (Feature Name)	คำอธิบาย
1	Flow Duration	ระยะเวลาของการสื่อสาร (milliseconds)
2	Total Fwd Packets	จำนวนแพ็กเกจฝั่งขาออก (Forward)
3	Total Backward Packets	จำนวนแพ็กเกจฝั่งย้อนกลับ
4	Total Length of Fwd Packets	ขนาดข้อมูลขาออกทั้งหมด
5	Total Length of Bwd Packets	ขนาดข้อมูลย้อนกลับทั้งหมด
6	Flow Bytes/s	ความเร็วในการส่งข้อมูล (Bytes per second)
7	Flow Packets/s	จำนวนแพ็กเกจต่อวินาที
8	Fwd Packet Length Mean	ขนาดเฉลี่ยของแพ็กเกจขาออก
9	Bwd Packet Length Mean	ขนาดเฉลี่ยของแพ็กเกจย้อนกลับ
10	Flow IAT Mean	ค่าเฉลี่ยของเวลาห่างระหว่างแพ็กเกจ
11	Fwd IAT Mean	เวลาห่างระหว่างแพ็กเกจขาออก
12	Bwd IAT Mean	เวลาห่างระหว่างแพ็กเกจย้อนกลับ
13	Fwd PSH Flags	จำนวน PSH flag ฝั่งขาออก
14	Fwd URG Flags	จำนวน URG flag ฝั่งขาออก
15	Fwd Header Length	ขนาด header ของแพ็กเกจขาออก
16	Bwd Header Length	ขนาด header ของแพ็กเกจย้อนกลับ
17	Subflow Fwd Packets	จำนวนแพ็กเกจย่อยฝั่งขาออก
18	Subflow Bwd Packets	จำนวนแพ็กเกจย่อยฝั่งย้อนกลับ
19	Init_Win_bytes_forward	ขนาด window แรกของการเชื่อมต่อ
20	Label	หมวดหมู่: ‘BENIGN’ หรือประเภทการโจมตี

ก่อนนำเข้าโมเดล ANN จะมีการทำ Normalization และแปลง Label ให้อยู่ในรูปแบบที่ระบบประมวลผลได้

- **ตัวอย่างข้อมูลที่ใช้ในการฝึกโมเดล**

ด้านล่างคือตัวอย่างข้อมูลที่นำมาใช้ฝึกโมเดล (ข้อมูลสมมุติจาก CICIDS 2018 ที่ผ่านการเตรียมไว้แล้ว):

ตาราง ก-3 ตัวอย่างข้อมูลที่ใช้ในการฝึกโมเดล

Flow Duration	Total Fwd Packets	Fwd Packet Length Mean	Flow Bytes/s	Label
12345	10	75.6	1200.5	BENIGN
54321	1	100.1	5000.0	DoS Hulk
10200	8	45.8	900.2	BENIGN
80000	30	95.0	15000.3	DDoS LOIC

หมายเหตุ: ข้อมูลในตารางผ่านการคัดกรอง, ทำความสะอาด (data cleaning), และปรับสเกลเรียบร้อยแล้ว เพื่อใช้ในการฝึกแบบจำลองด้วย TensorFlow.js บนระบบต้นแบบ

ภาคผนวก ข

ผลการประเมินประสิทธิภาพของแบบจำลองการตรวจจับภัยคุกคาม

- **ตารางประสิทธิภาพของแบบจำลอง ANN, SVM และ Random Forest**

แบบจำลองทั้งสามถูกนำมาฝึกและทดสอบกับชุดข้อมูลเดียวกัน โดยใช้ชุดข้อมูล CICIDS 2018 ที่ผ่านการเตรียมข้อมูลแล้ว ผลการประเมินใช้ค่าชี้วัดมาตรฐาน ได้แก่ Accuracy, Precision, Recall และ F1-Score

ตาราง ข-1 ตารางประสิทธิภาพของแบบจำลอง ANN, SVM และ Random Forest

แบบจำลอง	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
ANN	90.20	91.15	89.75	90.44
SVM	86.70	87.90	85.10	86.48
RF	88.10	89.40	87.00	88.18

หมายเหตุ: ค่าทั้งหมดประเมินจากชุดทดสอบที่แยกไว้ 20% ของข้อมูลทั้งหมด และใช้ค่าเฉลี่ยจากการรันซ้ำ 3 ครั้ง

- **Confusion Matrix ของแต่ละแบบจำลอง**

ตาราง Confusion Matrix แสดงจำนวนรายการที่แบบจำลองทำนายถูกหรือผิดในแต่ละกลุ่ม (ค่าจริงกับค่าที่ทำนาย)

ตาราง ข-2 Confusion Matrix – แบบจำลอง ANN

	ทำนายเป็น "โจมตี"	ทำนายเป็น "ปกติ"
ข้อมูลจริง: โจมตี	1052	98
ข้อมูลจริง: ปกติ	86	1310

ตาราง ข-3 Confusion Matrix – แบบจำลอง SVM

	ทำนายเป็น "โจมตี"	ทำนายเป็น "ปกติ"
ข้อมูลจริง: โจมตี	1001	149
ข้อมูลจริง: ปกติ	120	1276

ตาราง ข-4 Confusion Matrix – แบบจำลอง Random Forest

	ทำนายเป็น "โจมตี"	ทำนายเป็น "ปกติ"
ข้อมูลจริง: โจมตี	1030	120
ข้อมูลจริง: ปกติ	102	1294

- **ตารางเปรียบเทียบผลรวมระหว่างแบบจำลอง**

เพื่อนำเสนอผลการเปรียบเทียบอย่างชัดเจนระหว่างแบบจำลอง ANN, SVM และ RF สามารถสรุปผลรวมของตัวชี้วัดที่สำคัญได้ดังนี้:

ตาราง ข-5 ตารางเปรียบเทียบผลรวมระหว่างแบบจำลอง

ตัวชี้วัด / แบบจำลอง	ANN (%)	SVM (%)	RF (%)
Accuracy	90.20	86.70	88.10
Precision	91.15	87.90	89.40
Recall	89.75	85.10	87.00
F1-Score	90.44	86.48	88.18
จำนวนทำนายถูก (TP + TN)	2362	2277	2324

ANN แสดงผลแม่นยำสูงสุดในทุกตัวชี้วัด โดยเฉพาะ F1-Score ซึ่งสะท้อนถึงความสมดุลของ Precision และ Recall ได้ดีที่สุดในบริบทของการตรวจจับภัยคุกคาม

ภาคผนวก ค

โค้ดหลักของระบบต้นแบบสำหรับการตรวจจับภัยคุกคาม

- **โค้ดหลักของระบบต้นแบบสำหรับการตรวจจับภัยคุกคาม**

ระบบต้นแบบได้รับการพัฒนาในรูปแบบ Web-based application โดยใช้ภาษา JavaScript ร่วมกับไลบรารี TensorFlow.js สำหรับโหลดและใช้งานแบบจำลองปัญญาประดิษฐ์ (Artificial Neural Network) ที่ถูกฝึกไว้ล่วงหน้าในรูปแบบ JSON และ BIN

- **โค้ดโหลดโมเดล TensorFlow.js และเรียกใช้งาน**

```
let model;
async function loadModel() {
  try {
    model = await tf.loadLayersModel('model/model.json');
    console.log("Model loaded successfully.");
  } catch (error) {
    console.error("Error loading model:", error);
  }
}
```

<u>คำอธิบาย</u>:

- tf.loadLayersModel(...) ใช้โหลดโมเดล ANN ที่ฝึกแล้วและแปลงมาเป็น TensorFlow.js
- ไฟล์ model.json และ group1-shard1of1.bin ต้องอยู่ในโฟลเดอร์ /model
- คำสั่ง await รอให้โหลดเสร็จก่อนใช้งาน
- ตัวแปร model จะถูกเรียกใช้ในฟังก์ชันอื่นต่อไป

- **โค้ดอ่านไฟล์ CSV และประมวลผลข้อมูลนำเข้า**

ผู้ใช้สามารถอัปโหลดไฟล์ .csv ที่มีโครงสร้างตามฟีเจอร์ที่เลือกไว้ เพื่อวิเคราะห์ข้อมูล โดยใช้ PapaParse ในการแปลงข้อมูล

```
function handleFileUpload(event) {
  const file = event.target.files[0];
  Papa.parse(file, {
    header: true,
    dynamicTyping: true,
    complete: function(results) {
      const rawData = results.data;
```

```
      const inputData = preprocessData(rawData);
      predict(inputData);
    }
  });
}
```

คำอธิบาย:

- header: true ช่วยให้ PapaParse อ่านชื่อคอลัมน์จากแถวแรก
- dynamicTyping: true แปลงค่าตัวเลขอัตโนมัติ
- preprocessData(...) คือฟังก์ชันที่แปลงข้อมูลให้พร้อมสำหรับโมเดล
- เมื่อประมวลผลเสร็จ จะเรียก predict(...) เพื่อทำการพยากรณ์

ตัวอย่างฟังก์ชัน แปลงข้อมูล (Normalization และจัดรูปแบบ):

```
function preprocessData(dataArray) {
  const featureKeys = ["Flow Duration", "Total Fwd Packets", "Fwd Packet Length
Mean", "Flow Bytes/s"];
  const inputs = dataArray.map(row => {
    return featureKeys.map(key => normalize(row[key]));
  });
  return tf.tensor2d(inputs);
}
```

```
function normalize(value) {
  // ปรับค่านี้ให้เหมาะกับ dataset จริง
  const min = 0;
  const max = 100000;
  return (value - min) / (max - min);
}
```

- **โค้ดแสดงผลการตรวจจับภัยคุกคามบน Dashboard**

ผลลัพธ์จากโมเดลจะถูกแสดงออกมาเป็นข้อความ และอาจวาดกราฟสรุปประเภทของภัยคุกคามที่ตรวจพบ

ฟังก์ชันพยากรณ์ (Predict)

```
function predict(inputTensor) {
  if (!model) {
    alert("Model not loaded!");
    return;
  }
  const prediction = model.predict(inputTensor);
  prediction.array().then(scores => {
    displayResults(scores);
  });
}
```

แสดงผลลัพธ์แบบข้อความ

```
function displayResults(scores) {
  const outputDiv = document.getElementById("resultArea");
  outputDiv.innerHTML = "";
  scores.forEach((score, index) => {
    const label = score[0] > 0.5 ? "Attack" : "Normal";
    const confidence = (score[0] * 100).toFixed(2);
    const entry = `<p>Record ${index + 1}: ${label} (${confidence}%)</p>`;
    outputDiv.innerHTML += entry;
  });
}
```

แสดงผลลัพธ์แบบกราฟ

```
function drawSummaryChart(countNormal, countAttack) {
  const ctx = document.getElementById("summaryChart").getContext("2d");
  new Chart(ctx, {
    type: 'bar',
    data: {
      labels: ['Normal', 'Attack'],
      datasets: [{
        label: 'จำนวนการตรวจพบ',
        data: [countNormal, countAttack],
        backgroundColor: ['green', 'red']
      }]
    },
    options: {
      responsive: true,
      plugins: {
        legend: { display: false }
      }
    }
  });
}
```

ข้อกำหนดเบื้องต้นในการใช้งานโค้ด

- ไฟล์โมเดล .json และ .bin ต้องอยู่ในโฟลเดอร์ย่อย /model
- ต้องแนบไลบรารี: tensorflow.js, papaparse.js, chart.js

ระบบทำงานทั้งหมดแบบ client-side โดยไม่ต้องมีเซิร์ฟเวอร์

ประวัติผู้เขียน

ชื่อ	นายสีสุก สายเวหา
ชื่อวิทยานิพนธ์	การพัฒนาระบบตรวจจับภัยคุกคามในระบบเครือข่ายโดยใช้เทคโนโลยีปัญญาประดิษฐ์
สาขาวิชา	วิทยาศาสตร์ข้อมูลเพื่อนวัตกรรม
ประวัติ	พ.ศ 2565 สำเร็จการศึกษาระดับปริญญาตรี สาขาวิชาวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยแห่งชาติประเทศลาว พ.ศ 2555 ถึงปัจจุบันตำแหน่งวิชาการหน่วยงานไอทีวิทยาลัยครูสาละวัน