การศึกษาเปรียบเทียบอัลกอริธึมสำหรับทำลายนิ้วมือแบบใช้กลุ่มคำและกลุ่มอักษร

นายเหมวัสล์ปกร พลับอินทร์

การค้นคว้าอิสระนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตร
วิทยาศาสตรมหาบัณฑิต
สาขาวิชาการบริหารเครือข่ายและความมั่นคงปลอดภัยสารสนเทศ
บัณฑิตวิทยาลัย มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ
ปีการศึกษา 2567

การศึกษาเปรียบเทียบอัลกอริธึมสำหรับทำลายนิ้วมือแบบใช้กลุ่มคำและกลุ่มอักษร

นายเหมวัสล์ปกร พลับอินทร์

การค้นคว้าอิสระนี้เป็นส่วนหนึ่งของการศึกษาหลักสูตร
วิทยาศาสตรมหาบัณฑิต
สาขาวิชาการบริหารเครือข่ายและความมั่นคงปลอดภัยสารสนเทศ
บัณฑิตวิทยาลัย มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ
ปีการศึกษา 2567

ใบรับรองการค้นคว้าอิสระ

บัณฑิตวิทยาลัย มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

เรื่อง การศึกษาเปรียบเทียบอัลกอริธึมสำหรับทำลายนิ้วมือแบบใช้กลุ่มคำและกลุ่มอักษร

โดย นายเหมวัสล์ปกร พลับอินทร์

ได้รับอนุมัติให้นับเป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรมหาบัณฑิต สาขาวิชาการบริหารเครือข่ายและความมั่นคงปลอดภัยสารสนเทศ

..... คณบดีบัณฑิตวิทยาลัย / หัวหน้าภาควิชา

(ผู้ช่วยศาสตราจารย์ ดร.สุนันฑา สดสี)

คณะกรรมการสอบการค้นคว้าอิสระ

..... ประธานกรรมการ

(ผู้ช่วยศาสตราจารย์ ดร.เทอดพงษ์ แดงสี)

..... อาจารย์ที่ปรึกษา

(ผู้ช่วยศาสตราจารย์ ดร.นวพร วิสิฐพงศ์พันธ์)

..... กรรมการ

(รองศาสตราจารย์ ว่าที่ร้อยตรี ดร.พงษ์พิสิฐ วุฒิดิษฐโชติ)

ชื่อ : นายเหมวัสล์ปกร พลับอินทร์
ชื่อการค้นคว้าอิสระ : การศึกษาเปรียบเทียบอัลกอริธึมสำหรับทำลายนิ้วมือแบบใช้กลุ่มคำและกลุ่มอักษร
สาขาวิชา : การบริหารเครือข่ายและความมั่นคงปลอดภัยสารสนเทศ มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ
อาจารย์ที่ปรึกษาการค้นคว้าอิสระหลัก : ผู้ช่วยศาสตราจารย์ ดร.นวพร วิสิฐพงศ์พันธ์
ปีการศึกษา : 2567

บทคัดย่อ

บทความนี้นำเสนออัลกอริทึมสำหรับสร้างลายนิ้วมือ (Fingerprint) แบบกลุ่มอักษร (Character-based) เพื่อใช้ในการตรวจสอบและระบุความคล้ายคลึงกันของเอกสาร โดยมีเป้าหมายเพื่อตรวจจับและป้องกันปัญหาการลอกเลียนหรือการนำข้อมูลที่เป็นความลับจากเอกสารต้นฉบับไปใช้งานโดยไม่ได้รับอนุญาต การศึกษานี้ประยุกต์วิธีการทำลายนิ้วมือแบบคำ (Word-based) ด้วยการสกัดคุณลักษณะของข้อความ เพื่อสร้างเป็นลายนิ้วมือที่เป็นเอกลักษณ์เฉพาะของเอกสาร และทำการประเมินประสิทธิภาพของเทคนิคที่นำเสนอด้วยการวัดความคล้ายคลึงด้วยวิธี Jaccard Similarity ซึ่งผลการทดสอบประสิทธิภาพของเทคนิคที่นำเสนอแสดงให้เห็นว่าการสร้างลายนิ้วมือแบบกลุ่มอักษร ที่ขนาด 4-grams และจับกลุ่มด้วย window = 4 ให้ผลการตรวจจับความคล้ายคลึงที่ใกล้เคียงกับค่าเป้าหมายที่ทดลองมากที่สุด โดยจะคลาดเคลื่อนไม่เกิน 2-3% จากค่าเป้าหมาย ทั้งในกรณีที่ซ่อนข้อมูล ณ จุดเดียว (Undivided) และหลายจุด (Divided) ซึ่งเมื่อนำมาเปรียบเทียบกับเทคนิคการสร้างลายนิ้วมือแบบกลุ่มคำ (Word-based) ที่ใช้เป็นฐานสำหรับเปรียบเทียบแล้ว พบว่าแม้ว่าลายนิ้วมือแบบกลุ่มคำจะประมวลผลได้เร็วกว่าเท่าตัว แต่ให้ผลการตรวจจับความคล้ายคลึงต่ำกว่าความเป็นจริงเมื่อข้อมูลที่ต้องการตรวจจับถูกนำมาซ่อนในไฟล์อื่นโดยคลาดเคลื่อนตั้งแต่ 5-12%

(มีจำนวนทั้งสิ้น 39 หน้า)

คำสำคัญ : อัลกอริธึมสำหรับสร้างลายนิ้วมือ ดัชนีความคล้ายคลึงของจักการ์ด การป้องกันข้อมูลรั่วไหล

_____ อาจารย์ที่ปรึกษาการค้นคว้าอิสระหลัก

Name : Mr. Hemawatpakon Plabin
Independent Study Title : A Comparative Study of Algorithms for Fingerprint Generation Using Word-based and Character-based Groupings
Major Field : Network and Information Security Management
King Mongkut's University of Technology North Bangkok
Independent Study Advisor : Assistant Professor Nawaporn Wisitpongphan, Ph.D.
Academic Year : 2024

ABSTRACT

This article presents a character-based Fingerprint algorithm designed for examining and identifying document similarities. The objective is to detect and prevent issues related to plagiarism or unauthorized extraction and use of confidential information from original documents. This study applies techniques derived from word-based Fingerprinting by extracting text features, generating unique document Fingerprints, and evaluating their performance using the Jaccard Similarity method. Our results showed that the character-based fingerprint technique with 4-grams with window size = 4 provides similarity detection results that are closest to the target results, with deviation of no more than 2-3% from the target. This holds true for both cases where confidential information is hidden at a single point (Undivided) in multiple locations within the file (Divided). When compared to the benchmark word-based fingerprinting technique, it was found that the word-based approach can detect similarity twice as fast, but tends to underestimate similarity when target data is embedded in other files. The detection accuracy deviates from the target by 5-12%.

(Total 39 Pages)

Keywords: Fingerprinting Algorithms, Jaccard Similarity Index

_____ Advisor

กิตติกรรมประกาศ

การศึกษาค้นคว้าอิสระฉบับนี้สำเร็จลุล่วงไปได้ด้วยดี ด้วยการสนับสนุนและความช่วยเหลือจากหลายฝ่าย ผู้วิจัยขอแสดงความขอบคุณ ผู้ช่วยศาสตราจารย์ ดร.นวพร วิสิฐพงศ์พันธ์ อาจารย์ที่ปรึกษา ที่ได้ให้คำแนะนำอันเป็นประโยชน์ ตรวจสอบรายละเอียด และเสนอแนะแนวทางแก้ไขต่างๆ อย่างใส่ใจตลอดกระบวนการวิจัย นอกจากนี้ ผู้วิจัยขอขอบคุณ คณาจารย์ประจำภาควิชาการบริหารเครือข่ายดิจิทัลและความมั่นคงปลอดภัยสารสนเทศ ทุกท่าน ที่ได้ถ่ายทอดองค์ความรู้ในแขนงวิชาต่าง ๆ ซึ่งมีส่วนสำคัญต่อการดำเนินงานและการสังเคราะห์ข้อมูลในงานค้นคว้าอิสระนี้

สุดท้ายนี้ ผู้วิจัยขอขอบคุณตนเองที่มีความมุ่งมั่นและพากเพียรตลอดการดำเนินโครงการตลอดจนขอขอบคุณครอบครัว เพื่อนร่วมงาน และพี่น้องในภาควิชา ที่คอยให้การสนับสนุนทั้งในด้านกำลังใจและความช่วยเหลือ จนสามารถทำให้การศึกษาค้นคว้าอิสระฉบับนี้เสร็จสมบูรณ์ได้สำเร็จ

เหมวัสล์ปกร พลับอินทร์

สารบัญ

หน้า

สารบัญตาราง

หน้า

สารบัญรูปภาพ

หน้า

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปัญหา

ในปัจจุบันข้อมูลต่าง ๆ ส่วนใหญ่ถูกเก็บไว้ในรูปแบบของเอกสารอิเล็กทรอนิกส์ในคอมพิวเตอร์หรือในอุปกรณ์สำหรับจัดเก็บ เช่น เซิร์ฟเวอร์, ฮาร์ดไดรฟ์ภายนอก ซึ่งช่วยให้ผู้ใช้งานสามารถเข้าถึงข้อมูลได้สะดวกและรวดเร็ว โดยเฉพาะเมื่อมีการเชื่อมต่อกับอินเทอร์เน็ต แต่ในความสะดวกสบายมักมีความเสี่ยงที่ติดตามมาด้วย ในเครือข่ายอินเทอร์เน็ตนั้นมีช่องทางสำหรับผู้ไม่หวังดีเข้ามาละเมิดข้อมูลเอกสารสำคัญที่ไม่ได้รับอนุญาต เพื่อแสวงหาผลประโยชน์หรือมุ่งสร้างความเสียหาย ถือเป็นภัยคุกคามทางไซเบอร์ [1] รูปแบบหนึ่ง จากสถิติการละเมิดข้อมูลจาก Identity Theft Resource Center (ITRC) [2] ได้รวบรวมสถิติการละเมิดข้อมูลในสหรัฐอเมริกา (United States) เป็นหลักตั้งแต่ปี 2005 จนถึงปี 2023 พบว่ามีการเพิ่มขึ้นของการละเมิดข้อมูลจาก 156 เหตุการณ์เป็น 3,205 เหตุการณ์ ในปี 2023 มีผู้ได้รับผลกระทบถึง 353,027,892 ราย

Fingerprinting [3, 4] เป็นวิธีหนึ่งที่ถูกนำมาใช้เพื่อเพิ่มความมั่นคงปลอดภัยของข้อมูลหรือเอกสาร โดยใช้กระบวนการสร้างเอกลักษณ์เฉพาะสำหรับข้อมูลหรือเอกสาร เพื่อให้สามารถตรวจสอบและยืนยันได้ว่าเอกสารหรือข้อมูลดังกล่าว มีความคล้ายคลึงกับเอกสารที่เป็นเอกสารอ่อนไหวมากน้อยเพียงใด เราสามารถนำมาใช้ในการตรวจสอบเอกสารที่ถูกส่งผ่านเครือข่ายอินเทอร์เน็ต โดยเทคนิคที่นิยมนำมาใช้ในการพัฒนา Fingerprint เช่น Full Fingerprint [4], Selective Fingerprint [4] Extended New Fingerprint Method [5] ขั้นตอนการทำ Fingerprint นั้นเริ่มจาก (1) การสกัดคุณลักษณะ (Feature Extraction) เป็นการเตรียมข้อมูลโดยการแยกเอกสารออกเป็นหน่วยย่อย (อักขระ, คำ, ประโยค) และใช้เทคนิค Stop Words [6] และการทำ Stemming [7] เพื่อสร้างคุณลักษณะที่สามารถใช้ในการวิเคราะห์ข้อมูลต่อไป (2) การแยกองค์ประกอบของคุณลักษณะ (Feature Separation) ซึ่งมีทั้งการทำ n-gram แบบกลุ่มคำ (word-based) และ n-gram แบบกลุ่มอักษร (character-based) เทคนิคการแบ่งข้อมูลที่อยู่ติดกันเป็นจำนวน n ตัว เรียกว่า n-gram มีทั้งแบบเป็นกลุ่มคำ (word-based n-gram) และกลุ่มตัวอักษร (character-based n-gram) ซึ่ง word-based n-gram นิยมใช้เพื่อจับบริบทของวลี และ character-based n-gram มักนิยมใช้สำหรับการการระบุภาษา การตรวจจับผู้เขียน และ การจัดหมวดหมู่ข้อความตามหัวข้อ และในบางเทคนิคเช่น Extended New Fingerprint Method [5] มีการเพิ่มขั้นตอน Sorted k-skip เข้าไปในส่วนนี้ (3) การแฮชคุณลักษณะ (Feature Hashing) เป็นการแฮชข้อมูล n-gram ที่แยกองค์ประกอบได้ใน

ขั้นตอนที่ 2 โดยใช้เทคนิคต่าง ๆ เช่น MD5 [8], SHA, Rolling Hash [9] เป็นต้น และ (4) การเลือกค่าแฮช (Selected Fingerprints) เป็นขั้นตอน

ในการเลือกค่าแฮชบางส่วนจากค่าแฮชที่สร้างขึ้นจากเอกสารเพื่อใช้เป็นลายนิ้วมือของเอกสารนั้น ๆ โดยจะเลือกค่าแฮชที่มีค่าน้อยที่สุดในแต่ละหน้าต่างของแฮช Window of Hashes เพื่อเป็นตัวแทน Fingerprint ส่วนใน Full Fingerprint ซึ่งเป็นการใช้เนื้อหาทั้งหมดของเอกสารนั้นเพื่อเป็น Fingerprint

เทคนิค Fingerprint ที่กล่าวมาส่วนใหญ่นั้น นิยมสร้างด้วยวิธีการใช้ n-gram แบบกลุ่มคำ (word-based) เป็นหลัก โดยมีผลการศึกษาที่พบว่าการทำ fingerprint โดยใช้เนื้อหาทั้งหมดของเอกสารตามเทคนิค Full Fingerprint [4] นั้นอาจทำให้เกิดการตรวจจับผิดพลาดเมื่อมีการปรับแต่งคำในเอกสารเพียงเล็กน้อย และด้วยเทคนิคเดิมนั้นมุ่งเน้นการนำไปใช้เพื่อตรวจจับการลอกเลียนแบบเอกสารจึงทำให้ Fingerprint ที่สร้างขึ้นนั้นมีข้อมูลเนื้อหาทั้งหมดของเอกสาร ต่อมาเมื่อ Fingerprint ถูกนำมาประยุกต์ใช้เพื่อป้องกันข้อมูลลับรั่วไหล [5] จึงทำให้มีการพัฒนาต่อยอดจากเทคนิคเดิม โดยการเลือกนำเสนอวิธีการที่เรียกว่า Extended New Fingerprint Method ที่สร้าง Fingerprint จากข้อมูลที่ลับเท่านั้นด้วยเทคนิค k-skip-n-grams ซึ่งเทคนิคนี้มีความแม่นยำกว่า Full Fingerprint เมื่อกำหนดให้ใช้พื้นที่ในการจัดเก็บข้อมูลเท่ากัน ข้อเสียของ Extended New Fingerprint Method คือความซับซ้อนของการปรับแต่ง

อย่างไรก็ตาม เนื่องจากเทคนิคการสร้าง n-gram นั้นเป็นเทคนิคพื้นฐานในการวิจัยด้านภาษา ดังนั้นจึงมีการนำเทคนิคดังกล่าวไปประยุกต์ใช้ในหลากหลายแอปพลิเคชัน หนึ่งในนั้นคือการนำไปประยุกต์ใช้ในการตรวจจับอีเมลสแปม [10] ซึ่งมีการใช้ทั้ง n-gram แบบกลุ่มคำและกลุ่มอักษรในการวิจัย โดยผลลัพธ์ของการพัฒนาเทคนิคตรวจจับอีเมลสแปมนี้ระบุว่า n-gram แบบกลุ่มอักษรนั้นให้ผลลัพธ์ที่ดีกว่า n-gram แบบกลุ่มคำ

ในการค้นคว้าอิสระนี้ ผู้วิจัยจึงเลือกต่อยอดการพัฒนา Fingerprint เพื่อตรวจจับข้อมูลรั่วไหลจากเทคนิค Extended New Fingerprint Method เนื่องจากมีความยืดหยุ่นสูงในการตรวจสอบเอกสารที่มีการเปลี่ยนแปลง แต่ในการค้นคว้าอิสระนี้จะเสนอวิธีการใช้ n-gram โดยการแบ่งเป็น กลุ่มของ “ตัวอักษร” (Character-based) ร่วมกับเทคนิคการเรียงลำดับของคำ (Sorting) และเลือกทดลองใช้ขนาด 3-gram, 4-gram, และ 5-gram [11] สำหรับการทำ Fingerprint แล้วนำผล ที่ได้มาเปรียบเทียบกับการทำ Fingerprint แบบกลุ่มคำเพื่อให้เห็นถึงความแตกต่างระหว่างการทำ Fingerprint แบบ Word-based และ Character-based

1.2 วัตถุประสงค์

1.2.1 เพื่อเปรียบเทียบผลลัพธ์ค่าที่ได้จากการทำ Fingerprinting แบบ n-gram ที่ใช้กลุ่มคำ (Word-based) และกลุ่มตัวอักษร (Character-based)

1.2.2 เพื่อนำผลการเปรียบเทียบมาใช้เป็นแนวทางในการเลือกใช้ Fingerprinting ที่เหมาะกับวัตถุประสงค์ของผู้ใช้

1.3 ขอบเขตของการวิจัย

1.3.1 เทคนิคในการทำ Fingerprinting ต้นแบบที่นำมาใช้เป็นฐานของงานค้นคว้าอิสระคือ New Fingerprinting Method

1.3.2 ใช้โปรแกรมภาษา Python สำหรับพัฒนาเทคนิค Fingerprint

1.3.3 การทดสอบนี้ใช้เฉพาะกับข้อมูลประเภทข้อความภาษาอังกฤษ (textual data)

1.4 ประโยชน์ที่คาดว่าจะได้รับ

1.4.1 เพื่อให้เพื่อให้ทราบประสิทธิภาพและข้อจำกัดของการสร้าง Fingerprint แบบกลุ่มคำและกลุ่มอักษร

1.4.2 สามารถเลือกใช้เทคนิคการทำ Fingerprint ที่เหมาะสมกับวัตถุประสงค์การใช้งาน

บทที่ 2
ทฤษฎีอ้างอิงและงานวิจัยที่เกี่ยวข้อง

การเพิ่มความมั่นคงปลอดภัยของข้อมูลและเอกสาร โดยใช้กระบวนการสร้างเอกลักษณ์เฉพาะสำหรับแต่ละข้อมูลหรือเอกสาร เพื่อให้สามารถตรวจสอบและยืนยันระดับความคล้ายคลึงกับเอกสารที่มีความอ่อนไหว มีการประยุกต์ใช้ทฤษฎีและงานวิจัยที่เกี่ยวข้องเป็นแนวทางในการดำเนินงานดังต่อไปนี้

2.1 การตรวจจับและป้องกันการรั่วไหลของข้อมูลที่มีความสำคัญ DLP (Data Leak Prevention)

2.2 กระบวนการสร้างลายนิ้วมือของข้อมูล (Fingerprinting)

2.3 ดัชนีความคล้ายคลึงของจักการ์ด (Jaccard Similarity)

2.4 งานวิจัยที่เกี่ยวข้อง

2.1 การตรวจจับและป้องกันการรั่วไหลของข้อมูลที่มีความสำคัญ DLP (Data Leak Prevention)

DLP (Data Leak Prevention) [12, 13] เป็นแนวทางและเทคโนโลยีที่ใช้สำหรับ ป้องกันและตรวจจับการรั่วไหลของข้อมูลที่มีความสำคัญ ออกจากองค์กรโดยไม่ได้รับอนุญาต การรั่วไหลของข้อมูลอาจเกิดขึ้นได้ทั้งจาก ความตั้งใจ หรือ ความประมาทเลินเล่อของผู้ใช้ ซึ่งอาจส่งผลกระทบต่อความมั่นคงปลอดภัยและความลับของข้อมูลในองค์กร ดังนั้น เพื่อลดความเสี่ยงในการรั่วไหลของข้อมูล จึงมีการคิดค้นการใช้เทคนิคต่าง ๆ ในการควบคุมและป้องกันการเข้าถึงหรือส่งต่อข้อมูล ที่ไม่เหมาะสม เช่น การฝังลายน้ำลงในฐานข้อมูลเชิงสัมพันธ์ การจำกัดนามสกุลของไฟล์ที่สามารถส่งออกจากระบบได้ เป็นต้น

2.1.1 การฝังลายน้ำในความสัมพันธ์ของฐานข้อมูล (Watermarking Relational Databases) [14]

การฝังลายน้ำดิจิทัล เป็นกระบวนการที่ซอฟต์แวร์ แทรกข้อผิดพลาดขนาดเล็ก ลงในข้อมูลหรือวัตถุที่ต้องการปกป้อง ซึ่งข้อผิดพลาดเหล่านี้เรียกว่า "เครื่องหมาย" (Marks) และเมื่อเครื่องหมายทั้งหมดถูกรวมเข้าด้วยกัน จะถือเป็น "ลายน้ำ" (Watermark) ที่ใช้ในการติดตามและพิสูจน์ความเป็นเจ้าของข้อมูล

ลายน้ำที่ถูกฝังลงไปต้องถูกออกแบบให้ ไม่มีผลกระทบต่อการใช้งานของข้อมูลในเชิงนัยสำคัญ และต้องฝังอยู่ในลักษณะที่ ไม่สามารถถูกลบออกได้โดยง่าย โดยผู้ไม่หวังดี หากมีการลบลายน้ำ ข้อมูลต้องเสียหายหรือไม่สามารถใช้งานได้ตามปกติ

ดังนั้น แม้ว่าการฝังลายน้ำ ไม่สามารถป้องกันการคัดลอกข้อมูลได้โดยตรง แต่ช่วยให้สามารถระบุแหล่งที่มาของข้อมูลและป้องกันการละเมิดลิขสิทธิ์ได้ โดยใช้เป็นหลักฐานพิสูจน์ความเป็นเจ้าของข้อมูลที่ถูกแจกจ่ายหรือถูกนำไปใช้โดยไม่ได้รับอนุญาต

2.1.2 การจำกัดนามสกุลไฟล์ (Extension Restriction)

การจำกัดนามสกุลไฟล์เป็นอีกหนึ่งกลยุทธ์ที่ใช้ในระบบ DLP เพื่อ ควบคุมประเภทของไฟล์ที่สามารถส่งออกจากระบบองค์กร โดยกำหนดให้ อนุญาตเฉพาะไฟล์ที่สามารถตรวจสอบและจัดการได้ เช่น

ไฟล์ที่อนุญาตให้ส่งออก :

Excel (.xls)

Word (.doc)

PDF (.pdf)

Text File (.txt)

ไฟล์ที่ถูกจำกัด (ไม่สามารถส่งออกได้) :

รูปภาพ (.jpg, .png)

วิดีโอ (.mp4)

ไฟล์เสียง (.mp3)

ไฟล์ที่เข้ารหัสหรือบีบอัด (.zip, .rar)

2.1.3 การสร้างลายนิ้วมือของข้อมูล (Fingerprinting)

การสร้างลายนิ้วมือ เป็นการกำหนดเอกลักษณ์เฉพาะให้กับข้อมูลเป็นหนึ่งในเทคนิคที่ใช้ใน DLP (Data Leak Prevention) เพื่อควบคุมและป้องกันการเข้าถึงหรือการส่งต่อข้อมูลที่ไม่เหมาะสม

ทั้งยังช่วยตรวจสอบความถูกต้องและป้องกันการรั่วไหลของข้อมูลภายในองค์กรได้อย่างมีประสิทธิภาพ เทคนิคนี้ช่วยให้สามารถตรวจสอบและจำแนกความแตกต่างของข้อมูลจากชุดข้อมูลอื่นที่มีลักษณะใกล้เคียงกัน โดยสามารถนำไปใช้เพื่อเพิ่มระดับความมั่นคงปลอดภัยของข้อมูลและระบุแหล่งที่มาของข้อมูลที่อาจถูกนำไปใช้โดยไม่ได้รับอนุญาต ซึ่งช่วยเสริมความสามารถของระบบ DLP ในการป้องกันและติดตามการรั่วไหลของข้อมูลในองค์กร

2.2 กระบวนการสร้างลายนิ้วมือของข้อมูล (Fingerprinting)

กระบวนการสร้างลายนิ้วมือของข้อมูลสามารถทำได้โดยอาศัยหลักการสำคัญต่อไปนี้

2.2.1 การตัดคำฟุ่มเฟือย (Stop Words Removal)

Stop Words คือ คำที่ไม่มีความสำคัญในการวิเคราะห์ข้อความ เช่น "ที่", "ซึ่ง", "และ", "คือ" ในภาษาไทย หรือ "the", "is", "and", "or" ในภาษาอังกฤษ ซึ่งมักถูกตัดออกจากข้อความก่อนทำ Fingerprinting เพื่อลดขนาดของข้อมูลที่ต้องการประมวลผล เพื่อป้องกันไม่ให้คำที่ไม่มีความหมายสำคัญมีผลต่อ Fingerprint เพื่อเพิ่มความแม่นยำในการวัดผลความคล้ายคลึงของเอกสาร การ Stop Word Removal ถูกนำมาใช้จาก NLTK (Natural Language Toolkit) ซึ่งเป็นไลบรารีที่ใช้สำหรับการประมวลผลภาษาธรรมชาติ (NLP) ก่อนดำเนินการตัดคำฟุ่มเฟือยข้อความจะถูกแปลงตัวอักษรทั้งหมดให้เป็น ตัวพิมพ์เล็ก เพื่อให้มีรูปแบบที่สม่ำเสมอ ลดความแตกต่างที่เกิดจากตัวพิมพ์ใหญ่และพิมพ์เล็ก ซึ่งช่วยให้การวิเคราะห์ข้อความและกระบวนการตัดคำฟุ่มเฟือยมีความแม่นยำมากขึ้น

2.2.2 การลดรูปคำ (Stemming)

Stemming เป็นกระบวนการลดคำให้อยู่ในรูปพื้นฐาน (Root Form) โดยใช้กฎการตัดคำต่อท้ายหรือปรับเปลี่ยนรูปคำ

ตารางที่ 2-1 ตัวอย่างของ Stemming

คำต้นฉบับ	หลังทำ Stemming
Running	**Run**
Studies	**Studi**
Happily	**Happi**
connected	**Connect**

เพื่อลดจำนวนคำที่แตกต่างกันในข้อมูล ปรับปรุงการตรวจจับเอกสารที่คล้ายกัน ช่วยให้ Fingerprint มีความแม่นยำมากขึ้น

2.2.3 การแบ่งคำออกเป็นหน่วยย่อย (n-gram)

n-gram เป็นเทคนิคที่ใช้ในการแบ่งข้อความออกเป็นกลุ่มของ n คำ หรือ n ตัวอักษร ต่อเนื่องกัน ซึ่งเป็นพื้นฐานสำคัญของการทำ Fingerprinting การแบ่งข้อความอาจแบ่งเป็น กลุ่มคำ (Word-base) หรือแบบใช้ อักษร (Character-base) แล้วแต่ความเหมาะสม เพื่อช่วยให้ Fingerprint ทนทานต่อการเปลี่ยนแปลงบางส่วนของข้อความ ปรับปรุงการตรวจจับเอกสาร ที่คล้ายกัน ใช้ในระบบตรวจจับการคัดลอกและการเปรียบเทียบข้อความ ซึ่งตัวอย่างการแบ่งคำมีดังนี้

ข้อความต้นฉบับ

“This research paper offers a thorough examination”

การแบ่งข้อมูลแบบกลุ่มคำ (Word-base) เป็นวิธีการแยกข้อความออกเป็นหน่วยย่อยโดยพิจารณาตามขอบเขตของคำ (Word) ตามหลักภาษาศาสตร์ ซึ่งโดยทั่วไปในภาษาอังกฤษจะใช้ช่องว่าง (Space) เป็นตัวแบ่งคำ

ตัวอย่างการแบ่งกลุ่มคำแบบ Word 3-grams

‘research paper offer’, ‘paper offer thorough’ ‘offer thorough examin’

การแบ่งข้อมูลแบบกลุ่มอักขระ (Character-base) เป็นการแยกข้อความออกเป็นหน่วยย่อยที่เล็กที่สุดคือ อักขระ (Character) ซึ่งไม่มีการพิจารณาโครงสร้างของคำ ทำให้ได้ผลลัพธ์เป็นลำดับของตัวอักษรที่แยกออกจากกันทีละตัว

ตัวอย่างการแบ่งกลุ่มคำแบบ Character 3-grams

‘res’, ‘ese’, ‘sea’,...’ami’, ‘min’

2.2.4 การแบ่งข้อความหรือตัวอักษรออก ออกเป็นกลุ่มย่อย (Window Formation) [6]

Window Formation จะใช้ Window Size เป็นตัวกำหนดขนาดของหน้าต่างกลุ่มข้อมูลที่ได้หลังจากการทำ n-gram หรือ hash ตามต้องการ เช่น

ข้อความต้นฉบับ

“This research paper offers a thorough examination of the incorporation of state-of-the-art technologies, such as”

หากใช้วิธีแบ่งคำแบบ Word-based 3-grams ใช้ Window Size=4 จะได้เป็น

[‘offer paper research’, ‘offer paper thorough’, ‘examin offer thorough’, ‘examin incorpor thorough’] [‘offer paper thorough’, ‘examin offer thorough’, ‘examin incorpor thorough’, ‘examin incorpor stateoftheart’]...

และหากใช้วิธีแบ่งคำแบบ Character-based 3-grams ใช้ Window Size=4 จะได้เป็น

[‘res’, ‘ese’, ‘sea’, ‘ear’] [‘ese’, ‘sea’, ‘ear’, ‘arc’] [‘sea’, ‘ear’, ‘arc’, ‘rch’]...

2.2.5 การจัดเรียงลำดับของคำ (Sorting)

Sorting n-gram เป็นการจัดเรียงลำดับของคำภายในแต่ละ n-gram เพื่อทำให้ Fingerprint มีความทนทานต่อการเปลี่ยนแปลงลำดับของคำ เพื่อป้องกันผลกระทบจากการสลับลำดับคำ ปรับปรุงความแม่นยำของการเปรียบเทียบเอกสาร ลดปัญหาการเปลี่ยนแปลงข้อความที่เกิดจากการเรียงลำดับใหม่ยกตัวอย่างเช่น

ข้อความต้นฉบับ

“This research paper offers a thorough examination of the incorporation of state-of-the-art technologies, such as”

Character-based 3-grams (Window Size=4) ปกติ :

[‘res’, ‘ese’, ‘sea’, ‘ear’]

Sorting Character-based 3-grams (Window Size=4) :

[‘ear’, ‘ese’, ‘res’, ‘sea’]

Word-based 3-grams ปกติ:

‘research paper offer’, ‘paper offer thorough’, ‘offer thorough examin’, ‘thorough examin incorpor’

Sorting แล้วได้ผลดังนี้

Sorting Word-based 3-grams (1):

‘offer paper research’, ‘offer paper thorough’, ‘examin offer thorough’, ‘examin incorpor thorough’

ในกรณีที่ข้อความต้นฉบับมีความแตกต่างกันเล็กน้อย เช่น

“This paper research offers a examination thorough of the incorporation of technologies state-of-the-art , such as”

โดยสลับคำดังนี้

สลับ paper มาไว้ด้านหน้า	**research**
สลับ examination มาไว้ด้านหน้า	**thorough**
สลับ technologies มาไว้ด้านหน้า	**state-of-the-art**

Word-based 3-grams ปกติ:

‘paper research offer’, ‘research offer examin’, ‘offer examin thorough’, ‘examin thorough incorpor’

Sorting แล้วได้ผลดังนี้

Sorting Word-based 3-grams (2):

‘offer paper research’, ‘examin offer research’, ‘examin offer thorough’, ‘examin incorpor thorough’

นำผลจากการ Sorting Word-based 3-grams (1):และ Sorting Word-based 3-grams (2): มาเปรียบเทียบกัน

Sorting Word-based 3-grams (1):

‘offer paper research’, ‘offer paper thorough’, ‘examin offer thorough’, ‘examin incorpor thorough’

Sorting Word-based 3-grams (2):

‘offer paper research’, ‘examin offer research’, ‘examin offer thorough’, ‘examin incorpor thorough’

จะเห็นได้ว่าจาก (1) และ (2) การเปลี่ยนแปลงข้อความที่เกิดจากการเรียงลำดับใหม่ยังสามารถทำให้ข้อความมีความใกล้เคียงกันสามกลุ่มคำในสี่กลุ่มคำ

2.2.6 การแฮชข้อมูล (Hashing)

กระบวนการแปลงข้อมูลจากรูปแบบเดิมไปเป็นค่าตัวเลขมีขนาดคงที่ (Hash Value) โดยใช้ฟังก์ชันแฮช (Hash Function) เพื่อสร้างลายนิ้วมือดิจิทัล (Fingerprint) ของข้อมูลนั้น ๆ ซึ่งช่วยให้สามารถเปรียบเทียบ ตรวจสอบความถูกต้อง และระบุความคล้ายคลึงของข้อมูลได้อย่างรวดเร็วและมีประสิทธิภาพ มีรูปแบบดังสมการ

$$H(c_1...c_k)=c_1*b^{(k-1)}+c_2*b^{(k-2)}+...+c_{(k-1)} * b + c_k \quad (2\text{-}1)$$

โดยที่

$c_i \longrightarrow$ ค่า ASCII ของตัวอักษรตัวที่ i

b $\longrightarrow$ ค่าฐาน (Base Prime Number) ที่ใช้สำหรับแฮช (เช่น 37)

k $\longrightarrow$ ขนาดของ k-grams (จำนวนอักษรที่ใช้ทำแฮช)

ยกตัวอย่างเช่นคำว่า **“paper”**

b (base) = 37

m (modulus) = 1,000,003

n = 5 (จำนวนตัวอักษรใน "paper")

i เริ่มจาก 0 ไปจนถึง k-1

ord (char) คือค่า ASCII ของตัวอักษรแต่ละตัว

ตารางที่ 2-2 ตัวอย่างการคำนวณค่า Hash

Character	ASCII Value	Base Power ($37^{(n-i-1)}$)	Contribution
p	112	37^4=1,874,161	209,906,032
a	97	37^3=50,653	4,913,341
p	112	37^2=1,369	153,328
e	101	37^1=37	3,737
r	114	37^0=1	114

ตัวอย่างของการคำนวณค่า Hash

คำนวณผลรวม : 209,906,032+4,913,341+153,328+3,737+114=214,976,552

นำไป Modulus (mod 1,000,003) : 214,976,552mod1,000,003=**975,910**

```
Window 1 : ['paper', 'aperp', 'perpa', 'erpap']
Window 1 : [975910, 608791, 181647, 221147]
```

ภาพที่ 2-1 ตัวอย่างค่า Hash

2.2.7 การคัดเลือกค่า Fingerprint (Selected Fingerprints)

ในการเลือกค่าที่ใช้เป็น Fingerprint จะทำโดยเลือกค่าที่มีค่าน้อยที่สุดจากแต่ละ Window ที่กำหนด หากค่าใน Window ถัดไปยังคงมีค่าที่น้อยที่สุดจาก Window ก่อนหน้าอยู่ด้วย จะต้องทำการเปรียบเทียบว่ามีค่าที่น้อยกว่าหรือไม่

หากไม่มีค่าที่น้อยกว่า จะไม่มีการบันทึกค่าใหม่

หากพบค่าที่น้อยกว่า จะเลือกค่าที่น้อยกว่านั้นและบันทึกเป็น Fingerprint

ค่าที่ได้จากกระบวนการนี้จะถือเป็นผลลัพธ์ของ Fingerprint

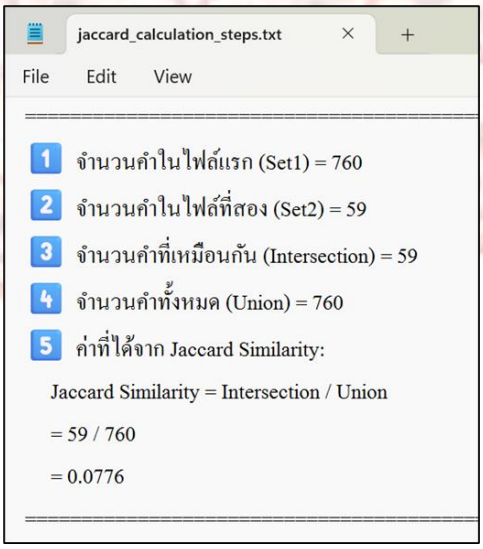
Window of hashed	Fingerprint result
[157029, 137038, 157179, 142601]	137038
[137038, 157179, 142601, 160385]	
[157179, 142601, 160385, 159918]	142601
[142601, 160385, 159918, 142625]	
[160385, 159918, 142625, 161269]	142625
[159918, 142625, 161269, 141972]	141972
[142625, 161269, 141972, 137110]	137110

ภาพที่ 2-2 ผลของการคัดเลือก Fingerprint

2.3 ดัชนีความคล้ายคลึงของจัคการ์ด (Jaccard Similarity)

ดัชนีความคล้ายคลึงของจัคการ์ด [15] เป็นวิธีการวัดความคล้ายคลึงของข้อมูลซึ่งสามารถทำได้โดยใช้ สัมประสิทธิ์ Jaccard (Jaccard Index) ซึ่งเป็นเมทริกซ์ที่นิยมใช้ในการวัดความคล้ายคลึงของชุดข้อมูลสองชุด มีค่าตั้งแต่ 0 ถึง 1 โดยที่ค่าเท่ากับ 0 หมายถึงไม่มีความคล้ายคลึงเลย ค่าเท่ากับ 1 หมายถึงชุดข้อมูลทั้ง 2 ชุดเป็นชุดข้อมูลที่เหมือนกัน ซึ่งสูตรในการหาสัมประสิทธิ์ Jaccard คือ

$$J(A, B) = |A \cap B| / |A \cup B| \quad (2\text{-}2)$$



ภาพที่ 2-3 ภาพแสดงขั้นตอนการคำนวณและผลลัพธ์ของ Jaccard Similarity

2.4 งานวิจัยที่เกี่ยวข้อง

Fingerprint นั้นเป็นกระบวนการเพิ่มลักษณะเฉพาะเข้าไปในสิ่งต่าง ๆ ที่ต้องการ เพื่อใช้แยกแยะความคล้ายคลึงกันของสิ่งที่สนใจ การทำ Fingerprint สามารถนำมาประยุกต์ใช้ในหลายด้าน เช่น การระบุตัวตนโดยการใช้ลายนิ้วมือ [16] ลายนิ้วมือเกิดจากลวดลายที่ก่อตัวขึ้นจากสันปาปิลลารีที่ยกขึ้นบนปลายนิ้ว ซึ่งมีรูพรุนเชื่อมต่อกับต่อมเหงื่อ จากการวิจัยและการศึกษาเป็นระยะเวลานาน พบว่าสันปาปิลลารีบนปลายนิ้ว ฝ่ามือ และฝ่าเท้า ยังคงรูปแบบดั้งเดิมตลอดชีวิต การเข้าใจถึงคุณค่าของลายนิ้วมือทำให้เกิดการวิจัยเทคนิคการตรวจจับและการใช้งานเชิงปฏิบัติ ทำให้นักบุกเบิกด้านลายนิ้วมือสามารถคิดค้นระบบการจำแนกลายนิ้วมือได้ในปัจจุบัน

ปัจจุบันมีการนำแนวคิดเรื่อง Fingerprint มาใช้ในการระบุตำแหน่งในอาคารของอุปกรณ์บนระบบเครือข่าย โดยการสร้าง Fingerprint จากรูปแบบความแรงสัญญาณ (RSSI) ของแต่ละพื้นที่ [17, 18] นอกจากนี้ยังมีการประยุกต์ใช้ Fingerprint ในการตรวจจับการรั่วไหลข้อมูล (Data Leak Detection) [19] และระบบป้องกันการรั่วไหลของข้อมูล (Data Leak Prevention) ซึ่งเป็นการป้องกันข้อมูลที่มีความสำคัญถูกส่งออกจากองค์กรโดยไม่ได้รับอนุญาต ซึ่งอาจเกิดขึ้นจากบุคคลภายใน ที่ตั้งใจทำให้ข้อมูลรั่วไหล หรือจากความประมาทเลินเล่อของผู้ใช้งานข้อมูล

ระบบการป้องกันการรั่วไหล DLP สามารถแบ่งออกเป็นหลายเทคนิค เช่น การฝังลายน้ำในความสัมพันธ์ของฐานข้อมูล (Watermarking Relational Databases) ซึ่งช่วยติดตามแหล่งที่มาของข้อมูลที่รั่วไหล การจำกัดนามสกุลไฟล์ (Extension Restriction) เพื่อลดความเสี่ยงในการรั่วไหลของข้อมูลผ่านไฟล์ที่ไม่สามารถตรวจสอบได้ง่าย และเทคนิคการทำลายนิ้วมือ (Fingerprinting)

ระบบป้องกันการรั่วไหลของข้อมูลที่ใช้หลักการของการทำลายนิ้วมือนั้นมีข้อดีคือสามารถนำมาใช้ป้องกันการรั่วไหลของข้อมูลที่ละเอียดอ่อน เนื่องจากมีความสามารถในการตรวจจับและป้องกันการรั่วไหลของเอกสารทั้งหมดหรือส่วนหนึ่งของเอกสารได้ อย่างไรก็ตาม การใช้ลายนิ้วมือแบบดั้งเดิมอาจไม่สามารถนำมาตรวจจับข้อมูลละเอียดอ่อนที่ถูกเปลี่ยนแปลงหรือแก้ไขได้ เนื่องจากการแฮชแบบดั้งเดิมที่ใช้ในการสร้างลายนิ้วมือข้อมูล เช่น MD5 และ SHA1 มีความไวต่อการเปลี่ยนแปลง การเปลี่ยนแปลงเพียงเล็กน้อยในข้อมูลอาจทำให้เกิดลายนิ้วมือที่แตกต่างออกไปทุกครั้งที่ทำการแฮช ซึ่งอาจส่งผลให้ระบบ DLPS ไม่สามารถตรวจจับข้อมูลที่รั่วไหลได้ ปัญหานี้สามารถแก้ไขได้บางส่วนโดยใช้การแฮชข้อมูลหลายครั้ง ซึ่งข้อมูลต้นฉบับจะถูกแบ่งออกเป็นส่วนเล็ก ๆ เช่น ย่อหน้าและประโยค โดยแต่ละส่วนจะถูกแฮชแยกกัน ทำให้สามารถตรวจจับข้อมูลที่ละเอียดอ่อนได้ดียิ่งขึ้น แต่ก็ยังไม่สามารถแก้ปัญหาเรื่องข้อมูลถูกเปลี่ยนแปลงได้ร้อยเปอร์เซ็นต์ ซึ่งทำให้เกิดวิธี

การตรวจจับข้อมูลรั่วไหลแบบอื่น เช่น similarity digest [20] การใช้ลายนิ้วมือแบบ Rabin Fingerprinting [20] การใช้ piecewise hashes [20] Extended New Fingerprint Method เทคนิคการทำลายนิ้วมือแบบ Winnowing [21] เป็นต้น ซึ่งแต่ละเทคนิคยังคงมีข้อจำกัดแตกต่างกัน

ผลการศึกษา พบว่า Extended New Fingerprint Method เป็นเทคนิคที่ใช้หลักการแบ่งเนื้อหาของเอกสารออกเป็นส่วน ๆ ก่อนทำการแฮชวิธีหนึ่งซึ่งสามารถนำมาพัฒนาต่อยอดได้ โดยหลักการของ Extended New Fingerprint Method เริ่มจากการสกัดคุณลักษณะสำคัญที่ประกอบด้วยโทเค็น ซึ่งจะได้รับการประมวลผลด้วยเทคนิค การลบคำหยุด การตัดคำ เพื่อให้เหลือเนื้อหาที่เป็นส่วนประกอบหลัก เมื่อสกัดเนื้อหาแล้ว นำมาแยกคุณลักษณะ (Feature separation) โดยใช้เทคนิค sorting k-skip-n-gram ซึ่งเป็นการจัดกลุ่มคำแบบ n-gram โดยจัดการรวมตัวอักษรเป็นกลุ่มละ n ตัวอักษรไปจนครบข้อมูลทั้งหมด ซึ่งจากการศึกษาพบว่าขนาดของ n = 3 จะส่งผลให้ Fingerprint มีประสิทธิภาพในการตรวจจับแม่นยำสูงสุด เมื่อเทียบกับ n ขนาดอื่น ผลของการจัดกลุ่มจะถูกนำมาแฮชด้วยอัลกอริทึม เช่น MD5, SHA หรือฟังก์ชันแฮชอื่น ๆ จะได้ค่าขนาดคงที่แทนข้อมูลเดิม ข้อมูลที่ได้จากการแฮชนี้ จะนำมาใช้แทนลายนิ้วมือของเอกสาร และสุดท้ายจะนำมาเข้ากระบวนการ Hash Selection เพื่อลดขนาดของข้อมูลเพื่อใช้เป็นลายนิ้วมือของเอกสาร

ในการตรวจจับการรั่วไหลของข้อมูลนั้น มักใช้วิธีการวัดความคล้ายคลึงของข้อมูล fingerprint ซึ่งสามารถทำได้โดยใช้ สัมประสิทธิ์ Jaccard (Jaccard Index) ซึ่งเป็นเมทริกซ์ที่นิยมใช้ในการวัดความคล้ายคลึงของชุดข้อมูลสองชุด ซึ่งมีค่าตั้งแต่ 0 ถึง 1 โดยที่ค่าเท่ากับ 0 หมายถึงไม่มีความคล้ายคลึงเลย ค่าเท่ากับ 1 หมายถึงชุดข้อมูลทั้ง 2 ชุดเป็นชุดข้อมูลที่เหมือนกัน

จากงานวิจัยที่เกี่ยวข้อง พบว่าการทำ Fingerprint แบบกลุ่มคำ (Word-Based) และ แบบกลุ่มอักษร (Character-Based) นั้นมีจุดเด่นต่างกัน โดยการทำ Fingerprint แบบกลุ่มคำนั้น นิยมใช้สำหรับการตรวจจับความคล้ายคลึงของเอกสารทั้งหมดเพื่อป้องกันการลอกเลียนแบบ แต่การใช้ Fingerprint แบบกลุ่มอักษรนั้นมักใช้สำหรับตรวจสอบข้อมูลบางส่วนของเอกสารที่ต้องการปกปิดเป็นความลับหรือต้องการตรวจจับ ซึ่งงานวิจัยส่วนใหญ่มักใช้วิธีการเปรียบเทียบเอกสารโดยตรงเพื่อหาความคล้ายคลึงหรือใช้สำหรับจำแนกประเภทเอกสาร ดังนั้น ผู้วิจัยจึงต้องการศึกษาเพิ่มเติมด้านความยืดหยุ่นของ Fingerprint แต่ละเทคนิคในการตรวจสอบเมื่อข้อมูลถูกดัดแปลงเพื่อหลบซ่อนการตรวจจับ เช่น การทยอยนำข้อมูลลับออกจากองค์กรโดยการแบ่งข้อมูลลับบางส่วนมาซ่อนในไฟล์ข้อมูลอื่นในสัดส่วนต่าง ๆ หรือการซ่อนข้อมูลลับตามจุดต่าง ๆ ของไฟล์ เป็นต้น ซึ่งในงานค้นคว้าอิสระนี้ผู้วิจัยจึงได้เสนอแนวทางการทำ Fingerprint แบบกลุ่มอักษร ร่วมกับการแบ่ง window size และจัดเรียงข้อมูล (sorting) ภายใน window โดยจะเปรียบเทียบผลกับการทำ Fingerprint แบบ กลุ่มคำ แบบ 3-gram ตามข้อเสนอแนะของ [5]

บทที่ 3
วิธีการดำเนินการวิจัย

การนำเทคนิค Fingerprinting แบบต่าง ๆ มาทดสอบและเปรียบเทียบประสิทธิภาพเพื่อใช้เป็นแนวทางในการเลือกใช้ fingerprinting ที่เหมาะกับวัตถุประสงค์ของผู้ใช้งาน เพื่อตรวจสอบความคล้ายคลึงของเอกสารและป้องกันการรั่วไหลของข้อมูล ซึ่งในการวิจัยครั้งนี้ มีการออกแบบกระบวนการและทดสอบโดยใช้ข้อมูลตัวอย่างเอกสาร ที่แตกต่างกัน และการนำผลลัพธ์ที่ได้มาเปรียบเทียบเพื่อประเมินความแม่นยำของระบบ โดยขั้นตอนการดำเนินการวิจัย มีดังนี้

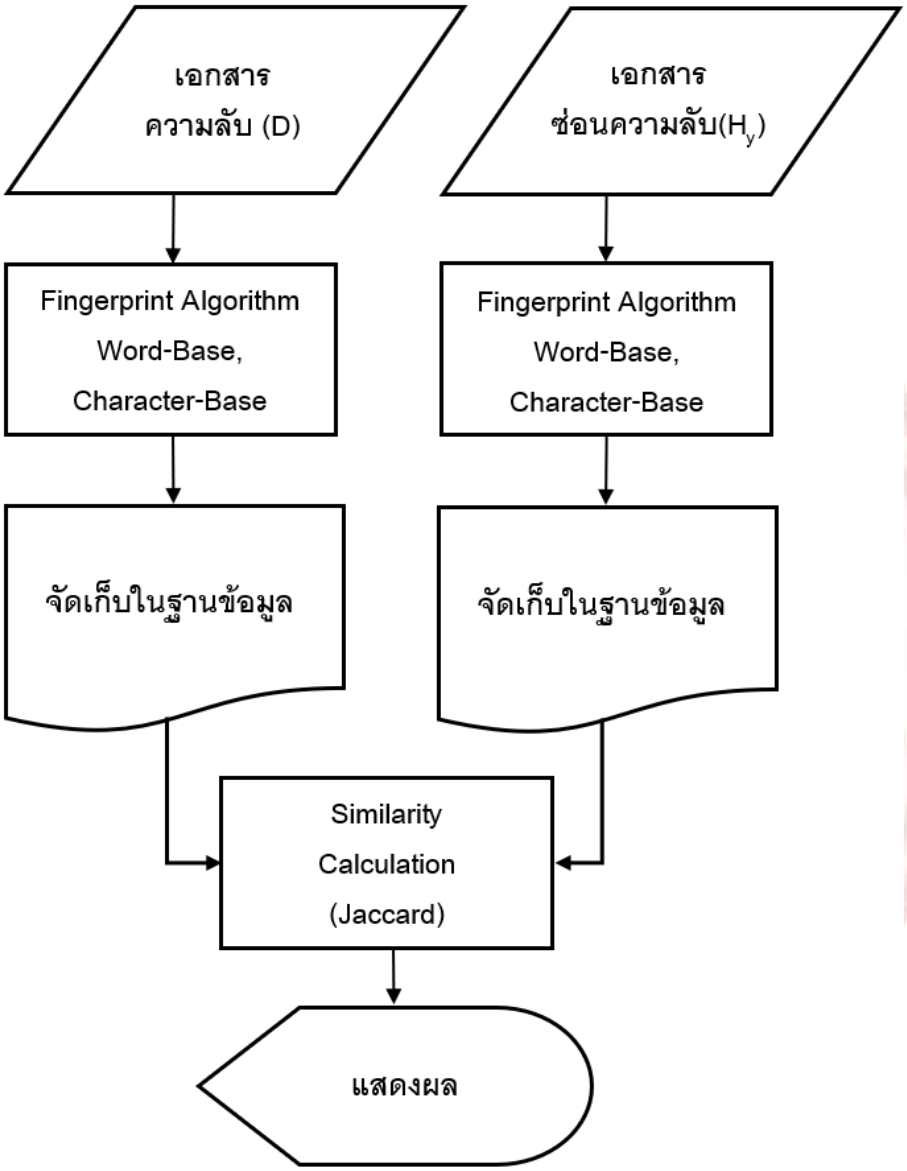
3.1 กรอบแนวคิดการดำเนินการวิจัย

3.2 การจัดเตรียมข้อมูลเอกสารสำหรับการทดสอบ

3.3 การทำลายนิ้วมือ (Fingerprinting)

3.4 การวิเคราะห์ประสิทธิภาพของลายนิ้วมือ

3.1 กรอบแนวคิดการดำเนินการวิจัย



ภาพที่ 3-1 ภาพรวมการดำเนินการวิจัย

ภาพที่ 3-1 แสดงกรอบแนวคิดในการออกแบบการทดสอบเทคนิค fingerprint แบบกลุ่มอักษรที่นำเสนอ โดยการนำข้อความจากเอกสารลับบางส่วนมาซ่อนในไฟล์อื่นเพื่อจำลองการลักลอบนำข้อมูลออกจากองค์กร โดยการตรวจจับการรั่วไหลของข้อมูลลับนี้สามารถทำได้โดยการเปรียบเทียบ Fingerprint ของไฟล์ดังกล่าวกับไฟล์ลับต้นฉบับ โดยกระบวนการหลักมีดังต่อไปนี้

3.2 การจัดเตรียมข้อมูลเอกสารสำหรับการทดสอบ

สร้างชุดข้อมูลสำหรับการทดสอบ โดยใช้ข้อมูลจากสองแหล่งที่มีเนื้อหาบางส่วนถูกตัดออกเพื่อรักษาความเป็นส่วนตัวและความมั่นคงปลอดภัยของข้อมูล ดังนี้

3.2.1 ข้อมูลสำหรับใช้ซ่อนข้อความที่ต้องการรักษาความเป็นส่วนตัว (H):

ใช้บทความที่นำมาจากข่าว "Why blocking China's DeepSeek from using US AI may be difficult" บนเว็บไซต์ Reuters [22] โดยข้อมูลนี้จะถูกจัดเก็บในรูปแบบไฟล์ .txt

3.2.2 ข้อมูลที่เป็นความลับที่ต้องการรักษาความเป็นส่วนตัว (D):

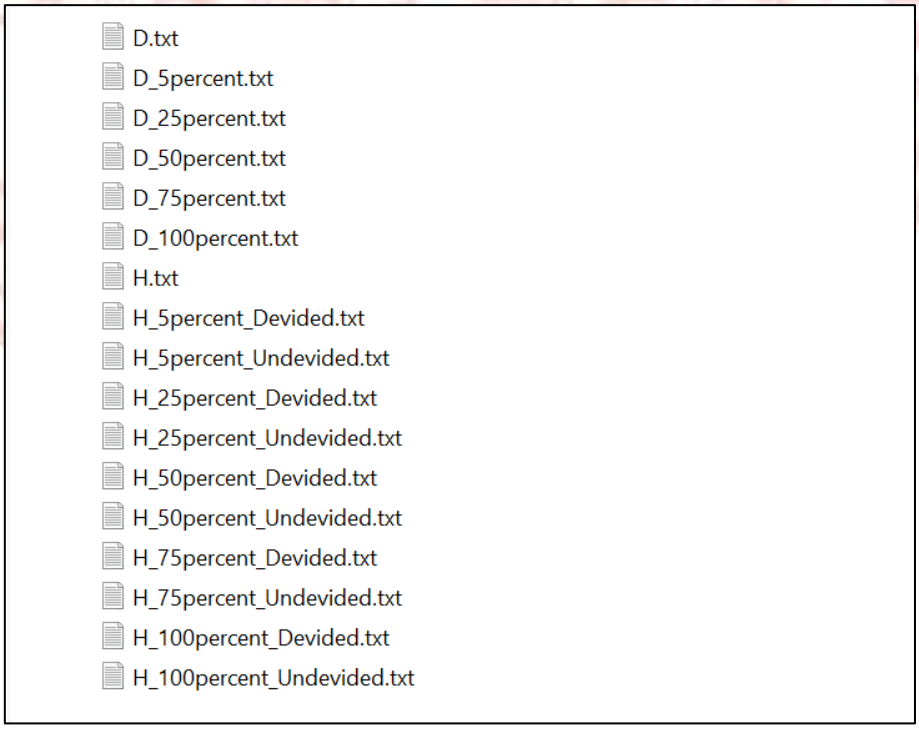
ใช้บทความวิจัย "AI Techniques in Product Design and Development" จากเว็บไซต์ SSRN [23] ข้อมูลนี้จะถูกจัดเก็บในรูปแบบไฟล์ .txt เช่นเดียวกัน

หลังจากจัดเก็บข้อมูลทั้งสองในรูปแบบไฟล์ .txt แล้ว ดำเนินการสร้างชุดข้อมูลทดสอบจากข้อมูล D โดยการสุ่มเลือกเนื้อหาบางส่วนในสัดส่วนต่างๆ ได้แก่ 5%, 25%, 50%, และ 75% ของเนื้อหาต้นฉบับ นำข้อมูลที่ได้ไปซ่อนใน H โดยแบ่งเป็นสองวิธี

1 นำข้อมูลทั้งหมด (Undivided) แทรกเข้าไปในจุดเดียวในเอกสาร H

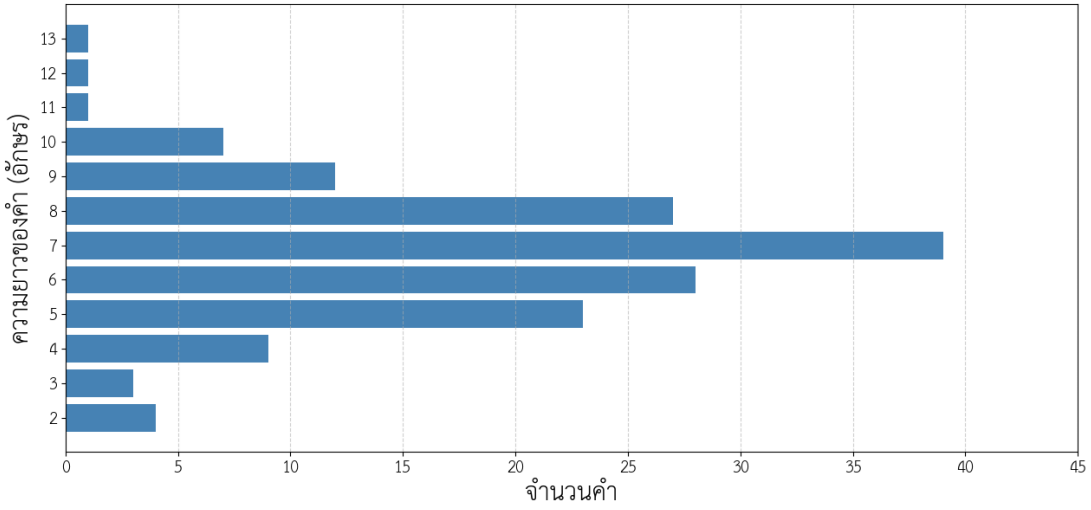
2 แบ่งข้อมูลออกเป็นส่วนเล็กๆ (Divided) แทรกเข้าไปในแต่ละจุดในเอกสาร H

จัดเก็บในรูปแบบไฟล์ .txt เพื่อใช้ในการทดสอบและวิเคราะห์ต่อไป



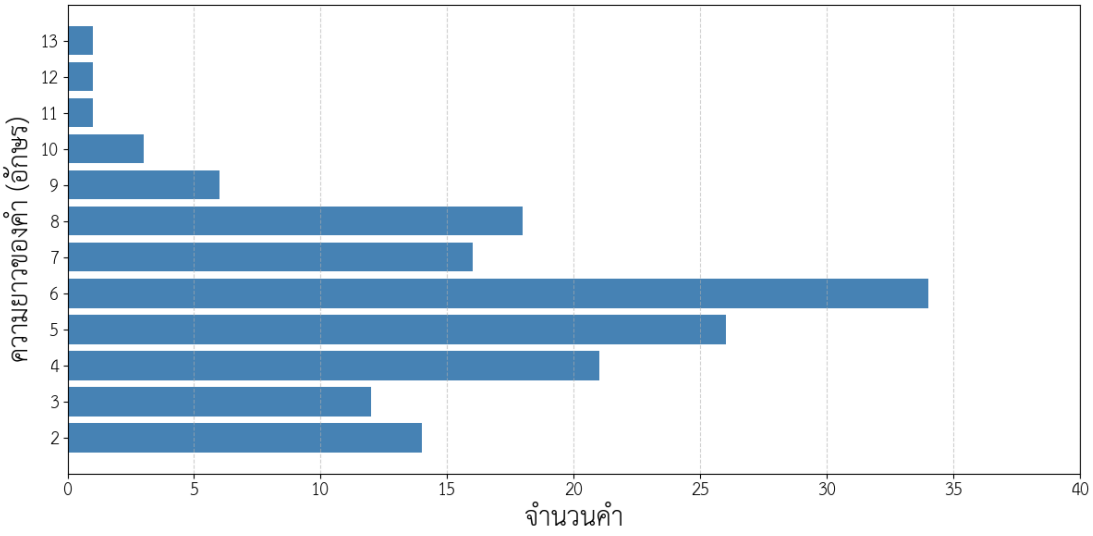
ภาพที่ 3-2 ไฟล์ข้อมูลที่ได้ทำการเตรียมไว้เพื่อทดสอบ

ไฟล์ข้อมูลเอกสารลับ มีจำนวนคำทั้งหมด 153 คำ โดยขนาดความยาวของคำในชุดข้อมูล ส่วนใหญ่มีความยาวอยู่ระหว่าง 5–8 ตัวอักษร และมีความยาวเฉลี่ยใกล้เคียงกับ 7 ตัวอักษร ดังแสดงในภาพที่ 3-3



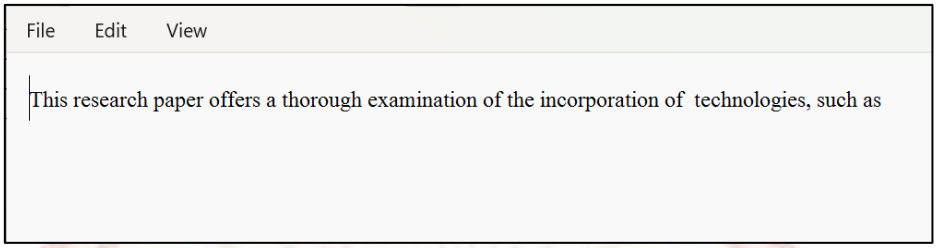
ภาพที่ 3-3 การกระจายตัวของจำนวนตัวอักษรที่อยู่ในคำของเอกสารลับ (D)

ส่วนไฟล์ข้อมูลที่จะใช้ซ่อนนั้น มีจำนวนคำทั้งหมด 155 คำ โดยคำส่วนใหญ่มีความยาวอยู่ระหว่าง 2–8 ตัวอักษร และมีความยาวเฉลี่ยใกล้เคียงกับ 6 ตัวอักษร ดังแสดงในภาพที่ 3-4



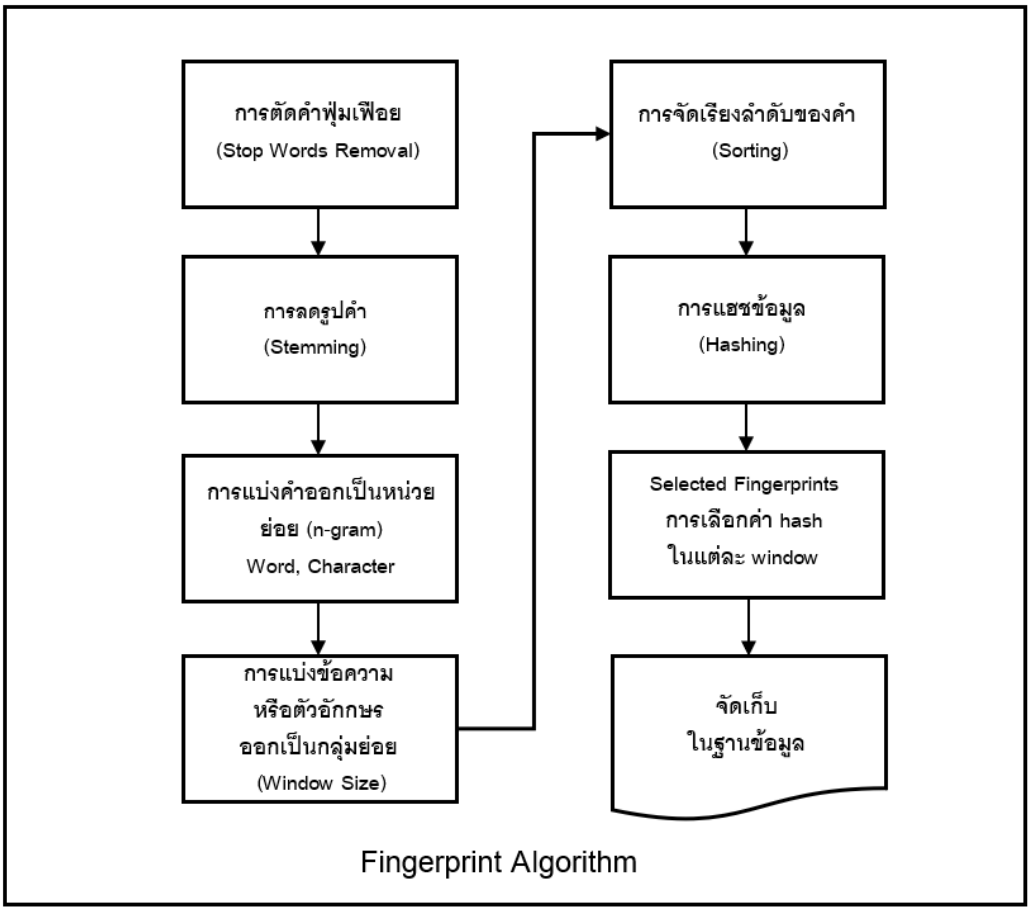
ภาพที่ 3-4 การกระจายตัวของจำนวนตัวอักษรที่อยู่ในคำของเอกสารสำหรับซ่อนข้อมูล (H)

ตัวอย่างของเนื้อหาที่ถูกแบ่งออกมาเพื่อนำมาซ่อน แสดงดังภาพที่ 3-5



ภาพที่ 3-5 ตัวอย่างเนื้อหาในไฟล์ข้อมูลที่ได้ทำการเตรียมไว้เพื่อทดสอบ

3.3 การทำลายนิ้วมือ (Fingerprinting)



ภาพที่ 3-6 ขั้นตอนกระบวนการทำลายนิ้วมือ

การจัดทำลายนิ้วมือ เริ่มจากการเตรียมข้อมูลโดยตัดคำฟุ่มเฟือย (Stop Words Removal) ที่ไม่มีความสำคัญในการวิเคราะห์ข้อความ ก่อนการนำมาลดรูปคำ (Stemming) ให้อยู่ในรูปพื้นฐาน (Root Form) โดยใช้กฎการตัดคำต่อท้ายหรือปรับเปลี่ยนรูปคำ จากนั้นแบ่งคำออกเป็นหน่วยย่อย (n-gram) หรือเป็นกลุ่มของ n คำ หรือ n ตัวอักษรต่อเนื่องกัน ซึ่งเป็นพื้นฐานสำคัญของการสร้างลายนิ้วมือ

เมื่อได้ n-gram แล้ว จะทำการแบ่งข้อความหรือตัวอักษรออกเป็นกลุ่มย่อย (Window Formation) โดยใช้ Window Size เป็นตัวกำหนดขนาดของกลุ่มข้อมูลที่ได้หลังจากการทำ n-gram หรือ hash จากนั้นจัดเรียงลำดับของคำ (Sorting) ภายในแต่ละ n-gram เพื่อทำให้ ลายนิ้วมือมีความทนทานต่อการเปลี่ยนแปลงลำดับของคำ และสุดท้ายทำการแฮชข้อมูล (Hashing) และเลือกค่าตัวแทนลายนิ้วมือก่อนนำมาจัดเก็บในฐานข้อมูล

3.4 การวิเคราะห์ประสิทธิภาพของลายนิ้วมือ

ในการตรวจจับการรั่วไหลของข้อมูล ผู้วิจัยใช้การวัดความคล้ายคลึงของลายนิ้วมือข้อมูล (fingerprint) โดยใช้ Jaccard (Jaccard Index) สำหรับวัดประสิทธิภาพของลายนิ้วมือแบบกลุ่มอักษร (Character-Based) ที่นำเสนอ เปรียบเทียบกับการใช้ลายนิ้วมือแบบกลุ่มคำ (Word-Based) ที่ใช้การตั้งค่าตัวแปรที่เสนอโดย [5] คือใช้ 3-gram และ window size = 4
การดำเนินการเปรียบเทียบ Fingerprint ของไฟล์ลับ (D) เทียบกับไฟล์ที่ถูกนำข้อมูลลับบางส่วนไปซ่อน (H_y) มีขั้นตอนดังนี้

3.4.1 ทดสอบความคล้ายคลึงระหว่างไฟล์ลับ (D) และไฟล์ที่ใช้ซ่อนข้อมูล (H) เพื่อแสดงให้เห็นว่า ไฟล์ทั้งสองไฟล์ที่นำมาใช้ในการทดสอบนั้นมีความแตกต่างกันมากเพียงไร

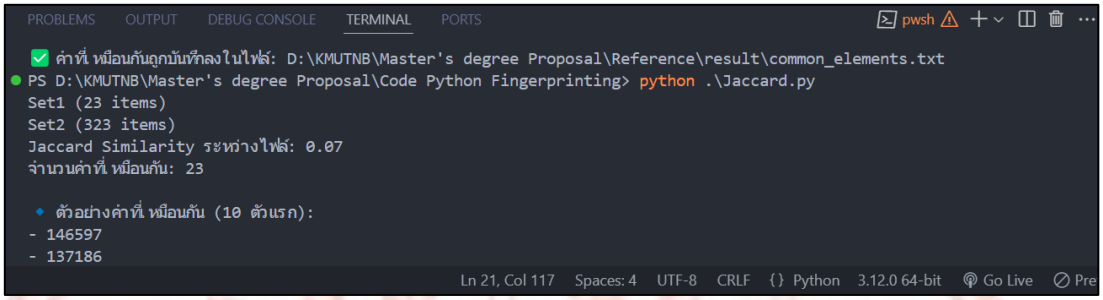
3.4.2 ทดสอบความคล้ายคลึงระหว่างไฟล์ลับ (D) และไฟล์ลับบางส่วนที่จะนำไปซ่อน (D_y) โดย y = สัดส่วนการแบ่งข้อมูล ซึ่งในการทดสอบนี้จะกำหนดให้ y = 5%, 25%, 50%, และ 75% การทดสอบนี้จะแสดงให้เห็นว่าสัดส่วนของข้อมูลที่แบ่งออกมานั้นเป็นไปตามที่วางแผนไว้ และเพื่อใช้วิเคราะห์หาค่า n ที่เหมาะสมในการทำ n-gram

3.4.3 ทดสอบความคล้ายคลึงของไฟล์ลับ (D) และไฟล์ที่ถูกใช้ซ่อนข้อมูลลับในสัดส่วนต่าง ๆ (H_y) โดย H_y คือไฟล์ H ที่มีข้อมูล y% จาก D ซ่อนอยู่ ซึ่งผู้วิจัยจะทำการซ่อนข้อมูล 2 แบบคือ การซ่อนข้อมูลในตำแหน่งเดียว (Undivided) และการซ่อนข้อมูลหลายตำแหน่ง (Divided)

3.4.4 ทดสอบคลวามคล้ายคลึงของไฟล์ที่ใช้ซ่อน (H) เทียบกับไฟล์ที่นำมีการนำข้อมูลมาซ่อนตามสัดส่วนที่กำหนด (H_y)

3.4.5 ทดสอบระยะเวลาที่แต่ละเทคนิคใช้ในการตรวจสอบความคล้ายคลึงของข้อมูล

ผลการทดสอบจะถูกส่งออก (export) ในรูปแบบไฟล์ .txt จากนั้นจึงนำค่าที่ได้ไปใส่ใน Excel เพื่อเปรียบเทียบผลลัพธ์ระหว่างการทดสอบในรูปแบบต่างๆ



ภาพที่ 3-7 ผลการปรียบเทียบ Jaccard Similarity

บทที่ 4
ผลการวิจัย

ในการเปรียบเทียบและวิเคราะห์ประสิทธิภาพของ Fingerprint ทางผู้วิจัยได้ทำการทดสอบเปรียบเทียบลายนิ้วมือ 2 รูปแบบ ได้แก่ แบบกลุ่มอักษร (Character-based) และ แบบกลุ่มคำ (Word-based) โดยใช้เกณฑ์การวัดค่าความคล้ายคลึงกัน เช่น Jaccard (Jaccard Index) ซึ่งเป็นมาตรวัดสำคัญในการประเมินผล ซึ่งในการทดสอบประสิทธิภาพนี้ จะดำเนินการในสภาพแวดล้อมที่ควบคุม เพื่อให้สามารถวิเคราะห์ผลลัพธ์ที่เกิดขึ้นได้อย่างถูกต้องและแม่นยำ และเพื่อให้ง่ายต่อการอธิบายผลการทดสอบ กำหนดให้

D	คือ	ข้อมูลที่ต้องการความเป็นส่วนตัว (ต้นฉบับ)
D_y	คือ	ข้อมูล y% ของข้อมูลต้นฉบับ (D)
H	คือ	ไฟล์ข้อมูลที่ใช้ซ่อนข้อมูลจาก D
H_y	คือ	ไฟล์ข้อมูลจำนวน y% ที่ถูกอำพรางในไฟล์ H
M’	คือ	Fingerprint ของไฟล์ข้อมูล m
M_{sw}	คือ	การจัดเรียงข้อมูล m ในกลุ่ม โดยใช้ window size = w
J(m, n)	คือ	ผล Jaccard ของ ไฟล์หรือข้อมูล m, n
T(m, n)	คือ	เวลาที่ใช้ในการเปรียบเทียบข้อมูล m และ n

ผู้วิจัยได้ดำเนินการสร้างชุดข้อมูลทดสอบ D และ H ซึ่งมีความคล้ายคลึงกัน 1.65% เพื่อนำมาใช้เป็นชุดข้อมูลตั้งต้นในการทดสอบประสิทธิภาพของเทคนิคการทำ Fingerprint แบบกลุ่มอักษร

จากการวิเคราะห์ความคล้ายคลึงกันของ Fingerprint ที่ได้ ซึ่งสามารถแบ่งผลการวิจัยออกเป็นหัวข้อต่าง ๆ ดังต่อไปนี้

4.1 ผลการเปรียบเทียบประสิทธิภาพในการตรวจจับความคล้ายคลึงระหว่าง D กับ H

4.2 ผลการเปรียบเทียบประสิทธิภาพในการตรวจจับความคล้ายคลึงระหว่าง D กับ D_y

4.3 ผลการเปรียบเทียบประสิทธิภาพในการตรวจจับความคล้ายคลึงระหว่าง D กับ H_y

4.4 ประสิทธิภาพด้านระยะเวลาที่ใช้ในการตรวจสอบความคล้ายคลึง

4.1 ประสิทธิภาพการตรวจจับความคล้ายคลึงระหว่าง D กับ H

เมื่อนำไฟล์ข้อมูล D กับ H มาเปรียบเทียบกันก่อนและหลังการทำ Fingerprint ด้วยค่าความคล้ายคลึงของ Jaccard ค่าความคล้ายคลึงภายหลังจากการทำ n-gram หรือ J(D,H) นั้น

ให้ผลค่าความคล้ายคลึงใกล้เคียงกับความเป็นจริง (1.65%) มากที่สุด คือ 0.02 หรือ 2% เมื่อ n = 5 แต่เมื่อดำเนินการจัดกลุ่ม n-gram ด้วย w = 4 และ จัดเรียงลำดับ n-gram ในแต่ละกลุ่มตามตัวอักษร ค่าความคล้ายคลึง $J(Ds_4,H_{s4})$ ของ 4-gram และ 5-gram จะเท่ากับ 0 ซึ่งบ่งบอกว่าไฟล์ข้อมูล D และ H มีข้อมูลต่างกัน 100% และเมื่อดำเนินการทำ Fingerprint ของทั้งสองไฟล์ข้อมูลจะพบว่าค่าความคล้ายคลึง J(D',H') นั้นใกล้เคียงความจริงมากที่สุดเมื่อ n = 5 เช่นกัน ดังตารางที่ 4-1

ตารางที่ 4-1 ค่าความคล้ายคลึงของ Jaccard ของชุดข้อมูลต้นฉบับ แบบกลุ่มอักษร (Character Based)

n-gram	***J(D,H)***	***$J(D_{s4},H_{s4})$***	***J(D',H')***
ค่าเป้าหมาย	0.01	0.01	0.01
3-gram	0.21	0.01	0.19
4-gram	0.06	0	0.06
5-gram	0.02	0	0.02

เมื่อใช้เทคนิคการจัดทำ Fingerprint แบบกลุ่มคำ (word-based) โดยใช้ 3-gram ตามผลการวิจัยใน [5] จะพบว่าค่าความคล้ายคลึงของทั้งสองไฟล์นั้นได้เท่ากับ 0 ทั้งก่อนและหลังการทำ Fingerprint ซึ่งบ่งบอกว่าทั้งสองไฟล์นั้นต่างกันโดยสิ้นเชิง

หากเปรียบเทียบระยะเวลาในการตรวจสอบความคล้ายคลึงแล้วพบว่า ระยะเวลาเฉลี่ยในการหาค่าความคล้ายคลึงก่อนทำ Fingerprint แบบกลุ่มอักษร อยู่ที่ 0.8099 ms และหลังทำ Fingerprint อยู่ที่ 0.5286 ms ซึ่งลดลงถึง 34.7% ส่วนระยะเวลาเฉลี่ยในการตรวจจับความคล้ายคลึงกรณีใช้ n-gram แบบกลุ่มคำ พบว่า ระยะเวลาที่ใช้นั้นน้อยกว่าการใช้ n-gram แบบกลุ่มอักษรเกือบครึ่งหนึ่ง เนื่องจากจำนวน gram ที่ได้จากการแบ่งข้อมูลตามกลุ่มคำนั้นน้อยกว่าการแบ่งข้อมูลตามกลุ่มตัวอักษรเท่าตัวนึง รายละเอียด แสดงดังตารางที่ 4-2

ตารางที่ 4-2 ระยะเวลาในการตรวจสอบความคล้ายคลึงของชุดข้อมูลต้นฉบับ แบบกลุ่มอักษร (Character-Based)

วิธีการ	***T(D,H)***	***T(D',H')***
กลุ่มอักษร	0.8099	0.5286
กลุ่มคำ	0.3846	0.2720

4.2 ประสิทธิภาพการตรวจจับความคล้ายคลึงระหว่าง D กับ D_y

เพื่อทดสอบประสิทธิภาพการตรวจจับความคล้ายคลึงของข้อมูลในแต่ละวิธีเมื่อเราทราบค่าสัดส่วนข้อมูลที่ถูกแบ่ง ผู้วิจัยได้ทำการเปรียบเทียบค่าความคล้ายคลึงโดยใช้วิธีการ Fingerprint แบบกลุ่มคำและกลุ่มอักษร ได้ผลดังแสดงในตารางที่ 4-3

ตารางที่ 4-3 ค่าความคล้ายคลึงของ Jaccard ระหว่างไฟล์ข้อมูลลับต้นฉบับ (D) และข้อมูลลับที่ถูกแบ่งออกมา (D_y) โดยใช้เทคนิคแบบกลุ่มอักษร

n-gram	$J(D',D'_5)$	$J(D',D'_{25})$	$J(D',D'_{50})$	$J(D',D'_{75})$
ค่าเป้าหมาย	0.05	0.25	0.50	0.75
3-gram	0.07	0.34	0.61	0.81
4-gram	0.06	0.3	0.56	0.79
5-gram	0.06	0.53	0.53	0.79

จากตารางที่ 4-3 สามารถสรุปได้ว่า หากปริมาณข้อมูลที่ถูกแบ่งออกไปเพิ่มขึ้น การจัดกลุ่มอักษรแบบ 4-gram จะได้ค่าความคล้ายคลึงใกล้เคียงกับสัดส่วนข้อมูลที่แบ่งออกมามากที่สุดในเกือบทุกกรณี

เมื่อใช้เทคนิคการจัดทำ Fingerprint แบบกลุ่มคำ และเปรียบเทียบความคล้ายคลึงกับข้อมูลที่แบ่งออกมาในสัดส่วน 0.05, 0.25, 0.50, และ 0.75 จะพบกว่าค่าเฉลี่ยความคล้ายคลึงที่ได้คือ 0.04, 0.23, 0.49, และ 0.77 ซึ่งใกล้เคียงกับค่าความคล้ายคลึงเป้าหมายมากกว่าการเทียบกับ Fingerprint แบบกลุ่มอักษร

4.3 ประสิทธิภาพการตรวจจับความคล้ายคลึงระหว่าง D กับ H_y

ประสิทธิภาพการตรวจจับความคล้ายคลึงระหว่างเอกสารหากนำไปใช้จริงในเครื่องมือ Data Leak Prevention ขององค์กรนั้น จะต้องจำลองสถานการณ์ 2 กรณี คือ การตรวจจับเมื่อข้อมูลลับชุดถูกซ่อนในตำแหน่งเดียวของไฟล์ (Undivided) และข้อมูลลับถูกซ่อนในตำแหน่งต่าง ๆ ของไฟล์ (Divided) ในงานวิจัยนี้ จำนวนคำในไฟล์ลับ (D) มีทั้งสิ้น 155 คำ และไฟล์ที่ถูกนำมาซ่อน (H) มีจำนวนคำทั้งสิ้น 153 คำ ซึ่งเมื่อนำข้อมูลลับในสัดส่วน 5%, 25%, 50%, และ 75% มาซ่อนในไฟล์ H ซึ่งมีคำเหมือนกับต้นฉบับอยู่ 5 คำคิดเป็น 1.65% นั้น จะทำให้สามารถคำนวณสัดส่วนความคล้ายคลึงเป้าหมายได้เท่ากับ 0.07, 0.22, 0.35, และ 0.44 ตามลำดับ

ผลการตรวจสอบค่าความคล้ายคลึงดังแสดงใน ตารางที่ 4-4

ตารางที่ 4-4 ค่าความคล้ายคลึงระหว่างไฟล์ลับและไฟล์ที่ถูกซ่อนข้อมูลในตำแหน่งเดียว (Undivided) ด้วยเทคนิค Fingerprint แบบกลุ่มอักษร (Character-Based)

n-gram	$J(D',H'_5)$	$J(D',H'_{25})$	$J(D',H'_{50})$	$J(D',H'_{75})$
Target	0.07	0.22	0.35	0.44
3-gram	0.21	0.30	0.44	0.52
4-gram	0.08	0.20	0.33	0.44
5-gram	0.04	0.16	0.28	0.4

พบว่าการจัดกลุ่มอักษรแบบ 4-gram ส่งผลให้สามารถตรวจจับค่าความคล้ายคลึงได้ใกล้เคียงค่าความคล้ายคลึงเป้าหมายมากที่สุด โดยคลาดเคลื่อนจากค่าเป้าหมายไม่เกิน 2% และหากเปรียบเทียบกับการจัดกลุ่มคำแบบ 3-gram แล้วพบว่าค่าความคล้ายคลึงอยู่ที่ 0.02, 0.12, 0.24, และ 0.37 ตามลำดับ ซึ่งผลที่ได้นี้แสดงให้เห็นว่าการตรวจจับความคล้ายคลึงโดยใช้เทคนิค Fingerprint แบบกลุ่มคำนั้นตรวจจับความคล้ายได้น้อยกว่าความเป็นจริง โดยคลาดเคลื่อนจากค่าเป้าหมาย 5% - 10%

ในกรณีที่ข้อมูลถูกนำมาซ่อนแบบกระจายจุดซ่อนในไฟล์ (Divided) จะได้ค่าความคล้ายคลึง ดังแสดงในตารางที่ 4-5

ตารางที่ 4-5 ค่าความคล้ายคลึงระหว่างไฟล์ลับและไฟล์ที่ถูกซ่อนข้อมูลหลายตำแหน่ง (Divided) ด้วยเทคนิค Fingerprint แบบกลุ่มอักษร (Character-Based)

n-gram	$J(D',H'_5)$	$J(D',H'_{25})$	$J(D',H'_{50})$	$J(D',H'_{75})$
ค่าเป้าหมาย	0.07	0.22	0.35	0.44
3-gram	0.21	0.3	0.43	0.52
4-gram	0.08	0.19	0.32	0.43
5-gram	0.04	0.15	0.27	0.39

ตารางที่ 4-5 แสดงให้เห็นว่า แม้ว่าข้อมูลจะถูกซ่อนแบบกระจายอยู่ในไฟล์ แต่เทคนิค Fingerprint แบบกลุ่มอักษรที่นำเสนอก็ยังสามารถตรวจจับความคล้ายคลึงได้ใกล้เคียงกับค่าเป้าหมายมากที่สุด หากใช้การจัดกลุ่มอักษรแบบ 4-gram โดยคลาดเคลื่อนไม่เกิน 3% จากค่าเป้าหมายนอกจากนี้เมื่อเปรียบเทียบประสิทธิภาพการตรวจจับความคล้ายคลึงของข้อมูลเทียบกับเทคนิคการทำ fingerprint แบบกลุ่มคำแบบ 3-gram แล้ว พบว่าผลความคล้ายคลึงอยู่ที่ 0.02, 0.10, 0.23, และ 0.32 ตามลำดับ ซึ่งผลดังกล่าวบ่งชี้ว่าเทคนิค fingerprint แบบกลุ่มคำนั้น ตรวจจับ

ความคล้ายคลึงได้ลดลงจากเดิมเมื่อข้อมูลถูกซ่อนแบบกระจายอยู่หลายจุด โดยคลาดเคลื่อนมากถึง 12%

4.4 ประสิทธิภาพด้านระยะเวลาที่ใช้ในการตรวจสอบความคล้ายคลึง

เมื่อพิจารณาประสิทธิภาพด้านระยะเวลาที่ใช้ในการตรวจสอบความคล้ายคลึงของข้อมูลแล้ว พบว่าระยะเวลาแปรผันตามปริมาณข้อมูลที่ตรวจสอบ โดยปริมาณข้อมูลมากจะใช้เวลามาก และวิธีการซ่อนข้อมูลนั้นไม่มีผลกับระยะเวลาในการตรวจจับ นอกจากนี้ยังพบว่าเทคนิคการทำ Fingerprint แบบกลุ่มอักษรนั้นใช้เวลาในการตรวจสอบมากกว่าเทคนิค Fingerprint แบบกลุ่มคำกว่าเท่าตัว ดังแสดงในตารางที่ 4-6

ตารางที่ 4-6 ผลการเปรียบเทียบระยะเวลาเวลาในการตรวจสอบความคล้ายคลึง

กรณี	ระยะเวลาในการตรวจสอบความคล้ายคลึง (ms)			
	$T(D',H'_5)$	$T(D',H'_{25})$	$T(D',H'_{50})$	$T(D',H'_{75})$
Undivided (กลุ่มอักษร)	0.5211	0.5623	0.5900	0.6289
Divided (กลุ่มอักษร)	0.5178	0.5473	0.5559	0.5930
Undivided (กลุ่มคำ)	0.2643	0.2909	0.2818	0.2975
Divided (กลุ่มคำ)	0.2789	0.2795	0.2894	0.2973

บทที่ 5
สรุป อภิปรายผล และข้อเสนอแนะ

5.1 อภิปรายผลการวิจัย

ผลการวิจัยแสดงให้เห็นว่าการใช้ Fingerprint มาเปรียบเทียบข้อมูลนั้นช่วยลดภาระการคำนวณเปรียบเทียบข้อมูลของเอกสารได้อย่างชัดเจน ซึ่งเป็นประโยชน์สำหรับระบบที่ต้องการวิเคราะห์ความคล้ายคลึงของข้อมูลจำนวนมากหรือมีความถี่ในการตรวจสอบสูง ซึ่งถือเป็นคุณสมบัติที่ตอบโจทย์ของระบบที่ต้องตรวจจับการรั่วไหลของข้อมูลแบบ near real-time หรือในสภาพแวดล้อมที่มีความเปลี่ยนแปลงของข้อมูลตลอดเวลา ในงานวิจัยนี้ได้นำเสนอประสิทธิภาพของ fingerprint แบบใช้กลุ่มคำและกลุ่มอักษร ซึ่งผลการทดสอบประสิทธิภาพโดยใช้ Jaccard Similarity แสดงให้เห็นว่าการใช้เทคนิคการสร้าง Fingerprint แบบกลุ่มอักษรนั้นสามารถตรวจจับความคล้ายคลึงระหว่างเอกสารได้แม่นยำกว่าเทคนิค Fingerprint แบบกลุ่มคำ แม้เอกสารจะถูกปรับเปลี่ยนโครงสร้างประโยคหรือเปลี่ยนลำดับของข้อมูลเล็กน้อยก็ตาม ส่งผลให้เทคนิคนี้มีความแม่นยำสูงกว่าในการประเมินความคล้ายคลึงของเอกสาร โดยเฉพาะในกรณีที่มีการพยายามดัดแปลงหรือพรางข้อมูล อย่างไรก็ตาม หากจะนำเทคนิคดังกล่าวไปประยุกต์ใช้จริงเพื่อตรวจจับการรั่วไหลของข้อมูลแล้ว ควรต้องคำนึงถึงปัจจัยเพิ่มเติม การออกแบบวิธีการเพื่อให้สามารถตรวจหาข้อมูลที่รั่วไหลได้อย่างแม่นยำ เช่น ควรศึกษาวิธีการการกำหนด threshold ที่เหมาะสมในการแจ้งเตือนเพื่อหลีกเลี่ยง false positive และ false negative ซึ่งอาจมีผลต่อความน่าเชื่อถือของระบบ นอกจากนี้ ในกรณีที่มีการรั่วไหลข้อมูลแบบค่อยเป็นค่อยไป (gradual leakage) เช่น การกระจายข้อมูลลับออกไปในหลายไฟล์และทยอยส่งออกทีละน้อย ระบบ DLP ที่ใช้ Fingerprint แบบดั้งเดิมอาจยังไม่สามารถตรวจจับได้อย่างครบถ้วน จึงควรมีการพัฒนาโมดูลเสริม เช่น การ เชื่อมโยง fingerprint ระหว่างหลายเอกสาร, การตรวจจับรูปแบบการรั่วไหลในระดับ session หรือ timeline เพื่อให้ครอบคลุมภัยคุกคามรูปแบบใหม่ได้อย่างมีประสิทธิภาพ

ดังนั้น แม้เทคนิค Fingerprint จะมีประโยชน์และประสิทธิภาพดีในการตรวจจับความคล้ายคลึงของข้อมูล แต่การประยุกต์ใช้งานจริงในบริบท DLP จำเป็นต้องมีการวางโครงสร้างที่รอบด้าน ทั้งด้าน algorithm, strategy และ policy เพื่อให้ระบบสามารถทำงานได้อย่างแม่นยำและมีประสิทธิภาพ

5.2 สรุปผลการวิจัย

งานวิจัยนี้นำเสนอเทคนิคการสร้างลายนิ้วมือแบบกลุ่มอักษร (Character-Based) ซึ่งพัฒนามาจากการสร้างลายนิ้วมือแบบกลุ่มคำ (Word-Based) ซึ่งผลการทดสอบประสิทธิภาพของเทคนิคที่นำเสนอแสดงให้เห็นว่าการสร้างลายนิ้วมือแบบกลุ่มอักษร ที่ขนาด 4-grams และจับกลุ่มด้วย window = 4 ให้ผลการตรวจจับความคล้ายคลึงที่ใกล้เคียงกับค่าเป้าหมายที่ทดลองมากที่สุด ซึ่งมีความสอดคล้องกับ [10] โดยผลความคลาดเคลื่อนในการทดลองนี้ไม่เกิน 2-3% จากค่าเป้าหมายทั้งในกรณีที่ซ่อนข้อมูล ณ จุดเดียว (Undivided) และหลายจุด (Divided) เมื่อนำมาเปรียบเทียบกับเทคนิคการสร้างลายนิ้วมือแบบกลุ่มคำ (Word-based) ที่ใช้เป็นฐานสำหรับเปรียบเทียบแล้ว พบว่าแม้ว่าลายนิ้วมือแบบกลุ่มคำจะประมวลผลได้เร็วกว่าเท่าตัว แต่ให้ผลการตรวจจับความคล้ายคลึงต่ำกว่าความเป็นจริงเมื่อข้อมูลที่ต้องการตรวจจับถูกนำมาซ่อนในไฟล์อื่น โดยคลาดเคลื่อนตั้งแต่ 5-12%

5.3 ข้อเสนอแนะ

ในการพัฒนาเทคนิคการทำ fingerprint แบบใช้กลุ่มอักษร (Character-based Fingerprinting) สามารถพิจารณาแนวทางปรับปรุงดังต่อไปนี้

5.3.1 การใช้เทคนิค k-skip-n-gram ร่วมกับ Window-size ขนาดต่างๆ

การผสมผสานเทคนิค k-skip-n-gram กับการปรับขนาด window-size สามารถช่วยเพิ่มความแม่นยำในการตรวจจับข้อความที่มีการเรียงลำดับคำแตกต่างกัน หรือมีคำแทรกเข้ามา โดยควรทดลองหลายค่าเพื่อหา configuration ที่เหมาะสมที่สุด ซึ่งจะส่งผลให้การจับคู่ข้อความมีความยืดหยุ่นและแม่นยำยิ่งขึ้น

5.3.2 การตรวจจับรูปแบบซ้ำที่เกิดบ่อยในข้อมูล

นำเทคนิคที่สามารถตรวจจับ pattern หรือกลุ่มคำที่มักปรากฏซ้ำในข้อความมาใช้ร่วมด้วย เช่น การวิเคราะห์ phrase-based หรือ chunk-based pattern ซึ่งสามารถช่วยให้การเปรียบเทียบข้อความมีประสิทธิภาพมากขึ้น

5.3.3 การทดสอบกับภาษาที่ไม่มีการเว้นวรรคตายตัว

ควรทดลองใช้ fingerprint กับภาษาเช่น ภาษาไทย หรือ ภาษาจีน ซึ่งไม่มีการเว้นวรรคคำที่แน่นอน โดยอาศัยเทคนิคการตัดคำ ที่หลากหลาย แล้วเปรียบเทียบผลลัพธ์กับภาษาที่มีโครงสร้างชัดเจน เช่น ภาษาอังกฤษ เพื่อวิเคราะห์ผลกระทบของลักษณะภาษาและโครงสร้างต่อค่าความคล้ายคลึงที่ได้

5.3.4 การออกแบบฟังก์ชันแฮชที่เหมาะสม

การเลือกค่าพารามิเตอร์สำหรับฟังก์ชันแฮชมีผลอย่างมากต่อคุณภาพของ fingerprint ที่ได้ ค่าฐาน (Base Prime Number) ควรเลือกเป็นจำนวนเฉพาะที่ไม่ใหญ่เกินไป เพื่อรักษาความเร็วในการคำนวณ และลดโอกาสเกิด pattern ซ้ำในค่าแฮช ค่า Modulus ควรเป็นจำนวนเฉพาะขนาดใหญ่ เพื่อป้องกันการชน (hash collision) และช่วยควบคุมขนาดของค่าแฮชให้อยู่ในช่วงที่เหมาะสม

บรรณานุกรม

[1] "สถิติภัยคุกคามทางไซเบอร์," NCSA, [Online]. Available: https://www.ncsa.or.th/service-statistics.html. [Accessed: Sept. 10, 2024].

[2] "ITRC 2023 Annual Data Breach Report," ITRC, Jan. 2024. [Online]. Available: https://www.idtheftcenter.org/wp-content/uploads/2024/01/ITRC_2023-Annual-Data-Breach-Report.pdf. [Accessed: Sept. 10, 2024

[3] N. R. Wagner, "Fingerprinting," 1983 IEEE Symposium on Security and Privacy, Oakland, CA, USA, 1983, pp. 18-18

[4] N. Heintze, "Scalable document fingerprinting (Extended Abstract)," in Proceedings of the 1996 USENIX Workshop on Electronic Commerce, 1996. [Online]. Available: http://www.cs.cmu.edu/afs/cs/user/nch/www/koala.html

[5] Y. Shapira, B. Shapira, and A. Shabtai, "Content-based data leakage detection using extended fingerprinting," arXiv preprint, arXiv:1302.2028 [cs.CR], 2013. [Online]. Available: https://doi.org/10.48550/arXiv.1302.2028

[6] E. Stamatatos, "Plagiarism detection using stopword n-grams," Journal of the American Society for Information Science and Technology, vol. 62, no. 12, pp.2512-2527, 2011. [Online]. Available: https://doi.org/10.1002/asi.21630

[7] M. F. Porter, "An algorithm for suffix stripping," Program: Electronic Library and Information Systems, vol. 14, no. 3, pp. 130-137, 1980. [Online]. Available: https://doi.org/10.1108/eb046814

[8] R. Rivest, "The MD5 message-digest algorithm," RFC 1321, 1992. [Online]. Available: https://doi.org/10.17487/RFC1321

[9] J. Kornblum, "Identifying almost identical files using context triggered piecewise hashing," Digital Investigation, vol. 3, special issue 1, pp. 91-97, 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1742287606000764. [Accessed: Sept. 10, 2024].

[10] I. Kanaris, K. Kanaris, I. Houvardas, and E. Stamatatos, "Words vs. character n-grams for anti-spam filtering," Int. J. Artif. Intell. Tools, vol. 16, no. 6, pp. 1047–

1070, Dec. 2007. [Online]. Available: ttps://doi.org/10.1142/S0218213007003692

[11] S. Shrestha, S. Gautam, K. Sharma, and A. Bhandari, "Winnowing algorithm: A powerful tool for identifying plagiarism in assignments," Journal of Trends in Computer Science and Smart Technology, 2023. [Online]. Available: https://doi.org/10.1109/ICIS.2018.8466429

[12] A. Jadhav and P. M. Chawan, "A System for Detection and Prevention of Data Leak" International Research Journal of Engineering and Technology (IRJET), vol. 9, no. 9, pp. 1019-1026, Sep. 2022. [Online]. Available: https://www.irjet.net/archives/V9/i9/IRJET-V9I9178.pdf

[13] A. Kozachok and S Kopylov, “Robust text watermarking based on line shifting,” Journal of Science and Technology on Information Security, December 24, 2018.[Online].Available: https://tailieu.antoanthongtin.vn/Files/files/site2/files/1_%20Robust%20text%20watermarking%20based%20on%20line%20shifting.pdf. [Accessed: March. 6, 2025].

[14] R. Agrawal and J. Kiernan, "Watermarking Relational Databases," VLDB ’02: Proceedings of the 28th International Conference on Very Large Data Bases, 2002,pp. 155–166. [Online]. Available: https://www.vldb.org/conf/2002/S05P03.pdf

[15] "Jaccard index," Wikipedia, Mar. 1, 2011. [Online]. Available: https://en.wikipedia.org/wiki/Jaccard_index. [Accessed: Sept. 10, 2024].

[16] N. Kaushal and P. Kaushal, "Human identification and fingerprints: A review," Journal of Biometrics & Biostatistics, vol. 2, no. 4, 2011. [Online]. Available: https://doi.org/10.4172/2155-6180.1000123

[17] N. Wisitpongphan, "Wireless sensor network planning for fingerprint based indoor localization using ZigBee: Empirical study," in The 9th International Conference on Computing and Information Technology (IC2IT 2013), P. Meesad, H. Unger, and S. Boonkrong, Eds., Advances in Intelligent Systems and Computing, vol. 209, Springer, Berlin, Heidelberg, 2013. [Online]. Available: https://doi.org/10.1007/978-3-642-37371-8_12

[18] Q. D. Vo and P. De, "A survey of fingerprint-based outdoor localization," IEEE Communications Surveys & Tutorials, vol. 18, no. 1, pp. 491-506, First Quarter 2016

[19] X. Shu and D. Yao, "Data leak detection as a service," in Security and Privacy in Communication Networks, A. Keromytis and R. Di Pietro, Eds., vol. 106, Springer, Berlin, Heidelberg, 2013, pp. 222-240.

[20] V. Roussev, "Data fingerprinting with similarity digests," in Advances in Digital Forensics VI, Springer, 2010, pp. 207-226.

[21] S. Schleimer, D. S. Wilkerson, and A. Aiken, "Winnowing: Local Algorithms for Document Fingerprinting," in Proceedings of the ACM SIGMOD International Conference on Management of Data, ACM, 2003, pp. 76–85. [Online]. Available: https://dl.acm.org/doi/pdf/10.1145/872757.872770

[22] S. Nellis, K. Hu, J. Dastin, A. Tong, and K. Paul, "Why blocking China's DeepSeek from using US AI may be difficult," Reuters, Jan. 29, 2025. [Online]. Available: https://www.reuters.com/technology/artificial-intelligence/why-blocking-chinas-deepseek-using-us-ai-may-be-difficult-2025-01-29/. [Accessed: Sept. 10, 2024].

[23] N. Rane, S. Choudhary, and J. Rane, "Enhanced Product Design and Development Using Artificial Intelligence (AI), Virtual Reality (VR), Augmented Reality (AR), 4D/5D/6D Printing, Internet of Things (IoT), and Blockchain: A Review," SSRN Electronic Journal, Nov. 2023. [Online]. Available: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4644059. [Accessed: Sept. 10, 2024].

This is Mendeley biography

ภาคผนวก

อัลกอริธึม สำหรับสร้าง n-gram

```
import nltk
from nltk.tokenize import word_tokenize
from nltk.corpus import stopwords
from nltk.stem import PorterStemmer
import os
import re
```

 # เพิ่มพาธข้อมูล NLTK

```
nltk.data.path.append( )
```

 # ตรวจสอบการดาวน์โหลด stopwords **และ** punkt (**สำหรับ** tokenization)

```
nltk.download( )
nltk.download( )
```

 # ระบุพาธและชื่อไฟล์ .txt **ที่ต้องการนำเข้ามาใช้งาน**

```
file_path =
with open( ) as file:
    text = file.read()
```

 # แปลงข้อความเป็นตัวพิมพ์เล็กและลบสัญลักษณ์พิเศษทั้งหมด ยกเว้นตัวอักษรและช่องว่าง

```
text = text.lower()
text = re.sub( )
```

 # ลบ stop words **และการทำ** stemming

```
ps = PorterStemmer()
stop_words = set(stopwords.words("english"))
tokens = word_tokenize(text)
processed_text = ''.join( )
```

Export **ผลลัพธ์การเตรียมข้อมูล** (Preprocessed Text)

```
base_name = os.path.splitext(os.path.basename(file_path))[0]
output_preprocessed_path = os.path.join( )
with open( ) as output_file:
    output_file.write( )
    output_file.write(processed_text)
```

 # ฟังก์ชันสำหรับสร้าง n-gram

```
def generate_n-gram(text):
    return [ ]
```

 # ฟังก์ชันการทำ Hash

```
def rolling_hash(  ):
    hash_value = 0
    for i, char in enumerate( ):
        hash_value += ord(char) * ( )
    return hash_value % modulus
```

 # ฟังก์ชันการแบ่ง window size=4

```
def winnowing_algorithm(hashes, window_size=4):
    fingerprints = []
    prev_selected = None
    if len(hashes) < window_size:
        return fingerprints
    for i in range( ):
        window = hashes[ ]
        min_value = min(window)
        if prev_selected is None:
            fingerprints.append(min_value)
            prev_selected = min_value
            continue
        if prev_selected in window:
            if min_value < prev_selected:
                fingerprints.append(min_value)
                prev_selected = min_value
        else:
            fingerprints.append(min_value)
            prev_selected = min_value
    return fingerprints
```

 # 1) สร้าง n-gram

```
normal_n-gram = generate_n-gram(processed_text)
```

 # 2) แบ่ง n-gram **ออกเป็น "หน้าต่างย่อย" (แต่ละชุดมี 4 ตัว)** + hash

```
window_size = 4
hashed_windows = []
windows_sorted = []
for i in range( ):
    window = normal_n-gram[ ]
    windows_sorted.append( )
    hashed_window = [ ]
    hashed_windows.extend( )
```

 # 3) เรียก Winnowing

```
fingerprints = winnowing_algorithm( )
```

 # 4) Export **ผลลัพธ์ลงไฟล์**

```
base_name = os.path.splitext( )
output_n-gram_path = os.path.join(os.path.dirname(file_path), f"{base_name}-ngrams.txt")
output_hashed_path = os.path.join(os.path.dirname(file_path), f"{base_name}-hash.txt")
output_fingerprint_path = os.path.join(os.path.dirname(file_path), f"{base_name}-fingerprints.txt")
output_window_sorted_path = os.path.join(os.path.dirname(file_path), f"{base_name}-window-sorted.txt")
```

 # 4.1) บันทึกผลลัพธ์ n-gram

```
with open(output_n-gram_path, "w", encoding="utf-8") as output_file:
    output_file.write( )
    output_file.write()
    for gram in normal_n-gram:
        output_file.write( )
```

 # 4.2) บันทึกผลลัพธ์ Hashed Values

```
with open( ) as output_file:
    output_file.write( )
    output_file.write(f )
    for hashed in hashed_windows:
        output_file.write( )
```

 # 4.3) บันทึกผลลัพธ์ Fingerprints

```
with open( ) as output_file:
    output_file.write( )
     output_file.write( )
     for fingerprint in fingerprints:
        output_file.write( )
```

 # 4.4) บันทึกผลลัพธ์ n-gram **ที่ถูกแบ่งเป็น** windows **และ** sorted **ภายในแต่ละ** window

```
output_n-gram_window_sorted_path = os.path.join(os.path.dirname(file_path))
window_size_4 = 4
window_sorted_n-gram = []

for i in range( ) - window_size_4 + 1):
    window = normal_n-gram[ ]
    sorted_window = sorted(window)
    window_sorted_n-gram.append(sorted_window)

with open( ) as output_file:
    for idx, window in enumerate( ):
        output_file.write( )
print( )
```

 # 4.5) บันทึกผลลัพธ์ n-gram **ที่ทำการ** Windows Sorted **และ** Hashed **แล้ว**

```
with open( ) as output_file:
    output_file.write( )
    output_file.write( )
    for idx, window in enumerate( ):
        hashed_window = [ ]
        output_file.write( )
        output_file.write( )
```

 # 5) ใช้ค่า Hash **จากไฟล์** xxx **และเก็บเฉพาะค่าที่มีตัวเลข**

```
input_hashed_sorted_file = output_window_sorted_path
output_hashed_windows_path = os.path.join(os.path.dirname(file_path))
hashed_windows_from_file = []
# อ่านค่า Hash จากไฟล์ xxx
with open( ) as file:
   lines = file.readlines()
   for line in lines:
        # ใช้ Regular Expression ค้นหาเฉพาะค่าที่อยู่ใน []
      match = re.search( )
      if match:
         numbers = match.group(1).strip()  # ดึงค่าภายใน []
         number_list = [ ]

           # หากมีค่าข้างใน ให้เพิ่มเข้าไปในลิสต์ในรูปแบบ string พร้อม []
         if number_list:
            hashed_windows_from_file.append( )
# บันทึกผลลัพธ์ลงไฟล์ใหม่
with open( ) as output_file:
   for window in hashed_windows_from_file:
      output_file.write( )  # บันทึกค่าเป็น string พร้อม []
```

```
# 6) ใช้ค่า Hash จากไฟล์ที่เก็บค่า hash มาทำ Fingerprinting
#  โดยการ reconstruct ลำดับ hash ต่อเนื่องจาก window ที่ซ้อนทับกัน
hashed_values_for_fingerprint = []
first_window = True
with open( ) as file:
   for line in file:
      line = line.strip()
      if line.startswith( ):
         inner = line[ ]
         window_numbers = [ ]
         if first_window:
            # เพิ่มค่า hash ทั้งหมดจาก window แรก
            hashed_values_for_fingerprint.extend(window_numbers)
            first_window = False
         else:
            if window_numbers:
               # สำหรับ window ถัดไป เพิ่มเฉพาะค่าใหม่ (ค่าตัวสุดท้าย) เท่านั้น
               hashed_values_for_fingerprint.append( )

# ทำ fingerprint ด้วยลำดับ hash ที่ต่อเนื่องที่สร้างได้
fingerprints_from_hashed = winnowing_algorithm( )

# บันทึก Fingerprints ที่ได้จาก hashed_values_for_fingerprint ลงไฟล์

output_fingerprint_from_hashed_path = os.path.join( )

with open( ) as output_file:

   output_file.write( )

   output_file.write( )

   for fingerprint in fingerprints_from_hashed:

      output_file.write( )

print( )
```

ประวัติผู้เขียน

ชื่อ	นายเหมวัสล์ปกร พลับอินทร์
ชื่อการค้นคว้าอิสระ	การศึกษาเปรียบเทียบอัลกอริธึมสำหรับทำลายนิ้วมือแบบใช้กลุ่มคำและกลุ่มอักษร
สาขาวิชา	การบริหารเครือข่ายและความมั่นคงปลอดภัยสารสนเทศ
ประวัติ	ระดับปริญญาตรี ภาควิชาวิศวกรรมโทรคมนาคม คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง