

การปรับปรุงระบบสกัด รวบรวมข้อมูล และสรุปเหตุการณ์น้ำท่วมจากข่าวออนไลน์

ด้วยโมเดลภาษาขนาดใหญ่

IMPROVING THE SYSTEM FOR EXTRACTING, GATHERING, AND SUMMARIZING FLOOD EVENTS FROM ONLINE NEWS WITH LARGE LANGUAGE MODELS

ชลิต โควีระวงศ์¹, รัตถชล อ่างมณี², ไพรินทร์ มีศรี³ และ ดารธดา วีระพันธ์^{1,*}

Chavalit Koweerawong¹, Rattachon Angmanee², Phairin Meesri³

and Daorathar Weerapan^{1,*}

¹สาขาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยราชภัฏวไลยอลงกรณ์ ในพระบรมราชูปถัมภ์

²สาขาการจัดการสารสนเทศ คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยราชภัฏวไลยอลงกรณ์ ในพระบรมราชูปถัมภ์

³สาขาเทคโนโลยีสารสนเทศ คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยราชภัฏวไลยอลงกรณ์ ในพระบรมราชูปถัมภ์

¹ Computer Science, Faculty of Science and Technology,

Valaya Alongkorn Rajabhat University under the Royal Patronage University.

² Disaster Management, Faculty of Science and Technology,

Valaya Alongkorn Rajabhat University under the Royal Patronage University.

³ Information technology, Faculty of Science and Technology,

Valaya Alongkorn Rajabhat University under the Royal Patronage University.

Received: 21 August 2026

Revised: 30 August 2026

Accepted: 31 August 2026

บทคัดย่อ

งานวิจัยนี้มีวัตถุประสงค์เพื่อปรับปรุงและพัฒนาประสิทธิภาพของระบบกึ่งอัตโนมัติสำหรับการสกัด รวบรวม และวิเคราะห์เชิงพื้นที่ของข้อมูลเหตุการณ์น้ำท่วมจากข่าวออนไลน์ในประเทศไทย โดยปรับปรุงข้อจำกัดของระบบเดิมที่พัฒนาด้วยภาษา Python และเทคนิคการสกัดคำสำคัญเชิงระบุตำแหน่ง (Named Entity Recognition: NER) อย่างโมเดล WangchanBERTa ซึ่งมีความซับซ้อนในการจัดการข้อมูลที่ไม่มีโครงสร้างและมีขั้นตอนประมวลผลหลายขั้นตอน การวิจัยนี้นำเสนอระบบที่ปรับปรุงใหม่โดยการใช้สถาปัตยกรรมภาษา JavaScript ร่วมกับ Node.js ในการควบคุมกระบวนการดึงข้อความจากหน้าเว็บ และ

* Corresponding author: ดารธดา วีระพันธ์

E-mail: daorathar@vru.ac.th

ใช้งานร่วมกับอินเทอร์เฟซจำลองกระบวนการทำงานอัตโนมัติ (n8n Workflow Automation) เป็นส่วนติดต่อผู้ใช้งาน โดยประยุกต์ใช้โมเดลภาษาขนาดใหญ่ (Large Language Model: LLM) ผ่าน Gemini Flash 2.5 API ในการทำหน้าที่เป็นตัวประมวลผลภาษาธรรมชาติ สกัดข้อมูลตัวตนเชิงเวลาและพื้นที่ (วันที่ สถานที่ และจังหวัด) สรุปใจความข่าว ตลอดจนประเมินระดับความรุนแรงของภัยพิบัติ การทดสอบระบบประมวลผลเปรียบเทียบกับชุดข้อมูลข่าวเหตุการณ์น้ำท่วมปี พ.ศ. 2567 จำนวน 555 ข่าว พบว่า ระบบใหม่ที่น่าเสนอสามารถลดระยะเวลารวมในกระบวนการประมวลผลข้อมูลตั้งแต่เริ่มต้นจนถึงการสร้างแผนภูมิภูมิสารสนเทศลงจาก 1,125 วินาที (19 นาที) เหลือเพียง 788 วินาที (13 นาที) หรือคิดเป็นการลดเวลาลงร้อยละ 29.95 ในด้านคุณภาพของข้อมูลที่สกัดได้ โมเดลภาษาขนาดใหญ่มีอัตราความแม่นยำเฉลี่ยสูงกว่าระบบเดิม โดยสามารถลดปัญหาข้อผิดพลาดจากการสกัดภาษาธรรมชาติที่มีโครงสร้างประโยคซับซ้อนหรือคำที่มีความกำกวม และสามารถสกัดมิติของเหตุการณ์เพิ่มเติม เช่น ข้อมูลระดับความรุนแรง ซึ่งระบบเดิมไม่สามารถจัดหมวดหมู่ได้ ผลการวิจัยชี้ให้เห็นว่าระบบที่ปรับปรุงใหม่มีความยืดหยุ่นในการเขียนโปรแกรมและการรวมระบบข้ามแพลตฟอร์มที่สูงขึ้น ช่วยให้เกิดแนวทางการสกัดและจัดเตรียมข้อมูลภัยพิบัติที่มีคุณภาพสูงสำหรับการแสดงผลผ่านแผนที่ความร้อนและการวิเคราะห์เชิงนโยบายเพื่อสนับสนุนการแจ้งเตือนภัยแบบกึ่งเรียลไทม์ได้อย่างมีนัยสำคัญ อย่างไรก็ตาม การนำระบบไปปรับใช้ในวงกว้างจำเป็นต้องคำนึงถึงประเด็นด้านความเป็นส่วนตัวและความปลอดภัยของข้อมูลในการสื่อสารผ่าน API ภายนอก รวมถึงการบริหารจัดการต้นทุนค่าใช้จ่ายในการเรียกใช้งานโมเดล

คำสำคัญ: ระบบอัตโนมัติเวิร์กโฟลว์, โมเดลภาษาขนาดใหญ่, การสกัดข้อมูล, การประมวลผลภาษาธรรมชาติ, การแจ้งเตือนภัยพิบัติน้ำท่วม

Abstract

This research aims to enhance the efficiency of a semi-automated system for extracting, gathering, and spatially analyzing flood event data from online news in Thailand. It addresses the limitations of the previous system, which was built on Python and Named Entity Recognition (NER) techniques like WangchanBERTa model, which suffered from complexity in handling unstructured data and multi-step workflows. The proposed system leverages a JavaScript

and Node.js-based architecture to manage web scraping tasks, integrated with n8n workflow automation as the user interface, and utilizes Gemini Flash 2.5 API (a Large Language Model: LLM) to process natural language, extract spatiotemporal entities (date, location, and province), summarize news content, and evaluate disaster severity levels. The system was evaluated using a dataset of 555 flood-related news articles from 2024. The experimental results demonstrated that the proposed system reduced the total processing time from 1,125 seconds (19 minutes) to 788 seconds (13 minutes), indicating a 29.95% reduction in computation time. In terms of data quality, the LLM exhibited higher extraction accuracy than the traditional NER, successfully handling complex sentence structures, resolving linguistic ambiguities, and capturing additional dimensions such as severity ratings that the previous system could not classify. The findings suggest that the redesigned system offers superior development flexibility and cross-platform integration, facilitating high-quality data preparation for heatmaps and policy-level spatial analysis to support semi-real-time warning systems. Nevertheless, deploying this system at scale requires careful consideration of data privacy and security during external API calls, as well as the cost management of language model token consumption.

Keywords: Workflow Automation, Large Language Models, Information Extraction, Natural Language Processing, Flood Disaster Warning

บทนำ

ในยุคข้อมูลขนาดใหญ่สื่อออนไลน์และข่าวสารดิจิทัลได้กลายเป็นแหล่งข้อมูลทางเลือกที่มีความสำคัญอย่างยิ่งในการติดตามสถานการณ์ภัยพิบัติได้อย่างรวดเร็วและครอบคลุม (Eze et al., 2020) เนื่องจากเหตุการณ์น้ำท่วมมักเกิดขึ้นอย่างฉับพลันและส่งผลกระทบเป็นบริเวณกว้าง การรายงานข่าวเชิงพื้นที่จากผู้สื่อข่าวภาคสนามหรือการรายงานข่าวท้องถิ่นออนไลน์จึงสามารถสะท้อนข้อมูลจริงได้เร็วกว่ารายงานอย่างเป็นทางการของหน่วยงานภาครัฐ ซึ่งบ่อยครั้งมีกระบวนการรายงานตามลำดับขั้นที่ซับซ้อน ส่งผลให้เกิดความล่าช้าของข้อมูลหรือขาด

รายละเอียดเชิงลึกในบางพื้นที่ห่างไกลที่ระบบตรวจวัดทางกายภาพยังเข้าไม่ถึง การนำข้อมูลเหล่านี้มาประมวลผลและวิเคราะห์เชิงพื้นที่และเวลาจึงเป็นแนวทางเสริมประสิทธิภาพการรายงานของภาครัฐ ช่วยลดความล่าช้าในการรวบรวมข้อมูล และเพิ่มความครอบคลุมเพื่อสร้างระบบเตือนภัยล่วงหน้าที่ตอบสนองต่อเหตุการณ์ได้ทันทั่วทั้ง (Revilla-Romero et al., 2015) อย่างไรก็ตาม การพัฒนาระบบสกัดข้อมูลภัยพิบัติแบบเดิมที่อาศัยเทคนิค Named Entity Recognition (NER) บนภาษา Python ยังคงมีข้อจำกัดเชิงปฏิบัติอยู่หลายประการ (ขวลิต โควิระวงศ์ และคณะ, 2568) ประเด็นแรกคือปัญหาความหน่วงในการประมวลผลเนื่องจากระบบทำงานในลักษณะของ Web server ทำให้การดึงข้อมูลและการตอบสนองต่อคำขอมีความหน่วงสะสมเชิงเครือข่าย ประเด็นต่อมาคือความซับซ้อนในการพัฒนาระบบผู้พัฒนาจำเป็นต้องเขียนโค้ดและเงื่อนไขจำนวนมากเพื่อเชื่อมโยงกระบวนการดึงข้อมูลจากหน้าเว็บ การทำความสะอาดข้อความ ตลอดจนการคัดเลือกและปรับแต่งโมเดลสกัดคำสำคัญภาษาไทย เช่น WangchanBERTa ซึ่งภาษาไทยยังมีข้อจำกัดเรื่องการแบ่งคำและการตีความคำกำกวม (Lowphansirikul et al., 2021; Phatthiyaphaibun et al., 2023) ประเด็นสุดท้ายคือระบบเดิยังขาดระบบการควบคุมลำดับงานที่เป็นระบบอัตโนมัติ ทำให้ไม่สามารถปรับขั้นตอนการสกัดข้อมูลหรือเชื่อมต่อข้ามแพลตฟอร์มแบบกึ่งเรียลไทม์เพื่อรองรับปริมาณข่าวสารจำนวนมากได้ ช่องว่างเหล่านี้ส่งผลให้ระบบเดิยังมีข้อจำกัดในการนำไปประยุกต์ใช้งานในสถานการณ์ภัยพิบัติจริงที่ต้องการข้อมูลที่รวดเร็วและต่อเนื่อง

จากที่กล่าวมางานวิจัยนี้ได้นำเสนอแนวทางประยุกต์ใช้แพลตฟอร์ม workflow automation แบบ open-source อย่าง n8n ซึ่งทำงานบนภาษา JavaScript และ NodeJS ทำให้สามารถออกแบบและจัดการขั้นตอนการประมวลผลข้อมูลในลักษณะ visual workflow ที่ลดภาระการเขียนโค้ดเชิงเทคนิคลงได้อย่างมาก (Deelman et al., 2015) ร่วมกับการใช้โมเดลภาษาขนาดใหญ่ (LLM) ผ่าน Gemini API ของ Google ซึ่งมีความสามารถในการเข้าใจไวยากรณ์และบริบทภาษาธรรมชาติภาษาไทยได้อย่างลึกซึ้ง (Google Gemini Team, 2023) ทำให้สามารถสกัดตัวตนเชิงพื้นที่ และเวลา สรุบน้ำวนหาข่าว และตีความระดับความรุนแรงของเหตุการณ์ได้โดยตรงแบบ Zero-shot (Wei et al., 2023) การผสมผสานนวัตกรรมทั้งสองจะทำให้ระบบใหม่ที่พัฒนาขึ้นมีความยืดหยุ่นในการเขียนโปรแกรมข้ามแพลตฟอร์มและมีความคล่องตัวเชิงพัฒนาสูงกว่าระบบเดิม นอกจากนี้ NodeJS ซึ่งมีจุดเด่นด้านสถาปัตยกรรมแบบขับเคลื่อนด้วยเหตุการณ์ร่วมกับการประมวลผลแบบอะซิงโครนัสและไม่บล็อกการทำงาน (Asynchronous Non-blocking I/O) ทำให้ระบบสามารถกระจายคำขอดึงข้อมูลหน้าเว็บ

และเรียกใช้ API ภายนอกได้พร้อมกันแบบคู่ขนานโดยไม่ต้องรอให้คำขอก่อนหน้าเสร็จสิ้น เหมือนธรรมชาติของ Python (Lei et al., 2014) และ Makarov et al. (2025) ได้ศึกษาผลของโมเดลการประมวลผลคำสั่งแบบอะซิงโครนัสเทียบกับแบบมัลติเธรดต่อประสิทธิภาพของเว็บแอปพลิเคชันฝั่งเซิร์ฟเวอร์ โดยการใช้การทดสอบโหลดพบว่าโมเดลอะซิงโครนัสของ Node.js ลด latency และเพิ่ม throughput ได้ดีในงานที่ผูกกับ I/O ซึ่งเป็นหลักฐานเชิงประจักษ์ที่รองรับประสิทธิภาพของ event loop

งานวิจัยของ ขวลิท โค้วระวงศ์ และคณะ (2568) ซึ่งเป็นงานวิจัยเดิมของผู้วิจัย มุ่งเน้นพัฒนาระบบกึ่งอัตโนมัติสำหรับการรวบรวมและวิเคราะห์ข่าวสารเกี่ยวกับน้ำท่วมในประเทศไทย โดยใช้เทคนิค Web Scraping ซึ่งเป็นการดึงข้อมูลจากหน้าเว็บ (Web Scraping) เป็นขั้นตอนพื้นฐานที่มีบทบาทอย่างยิ่งในการเปลี่ยนผ่านข้อมูลข่าวสารบนเว็บเพจที่ไม่มีโครงสร้างให้กลายเป็นชุดข้อมูลเชิงโครงสร้างสำหรับการทำเหมืองข้อมูลและการวิเคราะห์ขั้นสูง (Singrodia et al., 2019) ร่วมกับ Natural Language Processing (NLP) เพื่อระบุพื้นที่เสี่ยงและแนวโน้มสถานการณ์น้ำท่วม ผ่านการสกัดข้อมูลสำคัญ เช่น ชื่อสถานที่และเวลา ด้วยห้องสมุดระบบประมวลผลภาษาธรรมชาติภาษาไทย ร่วมกับการแสดงผลข้อมูลในรูปแบบแผนที่ความร้อน ผลการทดลองพบว่าสามารถรวบรวมและวิเคราะห์ข่าวน้ำท่วมจำนวนมากในปี 2567 ได้ภายในระยะเวลาเพียง 1 ชั่วโมง 33 นาที แสดงให้เห็นถึงศักยภาพของระบบในการสรุปสถานการณ์ได้อย่างรวดเร็วและแม่นยำ แม้จะยังมีข้อจำกัดด้านความหลากหลายของข้อมูลและความซับซ้อนของการประมวลผลก็ตาม โดยมีข้อเสนอแนะให้ขยายแหล่งข้อมูลไปยังโซเชียลมีเดียและหน่วยงานภาครัฐ รวมถึงประยุกต์ใช้เทคนิค AI และระบบภูมิสารสนเทศเพื่อพัฒนาระบบแจ้งเตือนภัยที่ทันสมัยและเข้าถึงผู้ใช้งานได้อย่างมีประสิทธิภาพ ซึ่งงานวิจัยดังกล่าวเป็นการรันการทำงานบน Web Server ของภาษา Python อาจทำให้เกิดความล่าช้าจากการสื่อสารข้ามเน็ตเวิร์ค

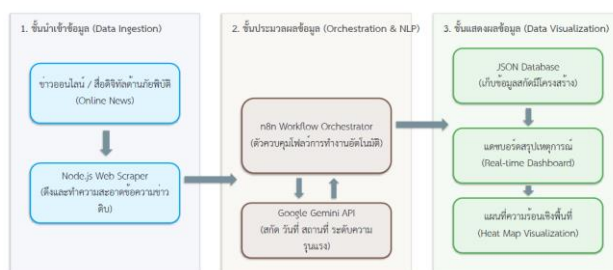
จากการทบทวนวรรณกรรมข้างต้น ผู้วิจัยได้ทำการสรุปเปรียบเทียบจุดแข็งและข้อจำกัดของเทคโนโลยีแต่ละประเภทที่นิยมนำมาใช้ในการสกัดข้อมูลภัยพิบัติเชิงเหตุการณ์ออนไลน์ ได้แก่ Traditional NER, BERT-based NER และ Large Language Models (LLMs) Traditional NER (CRF / Rule-based) เป็นแนวทางที่พึ่งพาพจนานุกรม กฎเกณฑ์ทางภาษาศาสตร์ และแบบจำลองความน่าจะเป็นเชิงลำดับ มีจุดเด่นคือประหยัดทรัพยากร ทำงานแบบออฟไลน์บนเครื่องประมวลผลทั่วไปได้โดยไม่ต้องพึ่งพา GPU และแม่นยำในระบบปิด แต่มีข้อจำกัดด้านความยืดหยุ่น ไม่สามารถตรวจจับคำศัพท์ใหม่นอกฐานข้อมูลได้ดี ต้องอาศัย

ผู้เชี่ยวชาญในการปรับแต่งกฎ และความแม่นยำจะลดลงเมื่อเจอประโยคที่ซับซ้อนหรือกำกวม BERT-based NER (เช่น WangchanBERTa) เป็นการประยุกต์ใช้โมเดลภาษาขนาดเล็กกลุ่ม Transformer ที่ผ่านการฝึกฝนข้อความภาษาไทย จึงเข้าใจโครงสร้างบริบทและความสัมพันธ์ของคำได้ดี สามารถสกัดตัวตนเฉพาะด้านในระดับประโยคได้แม่นยำ และติดตั้งแบบ On-premise ในระบบปิดเพื่อความปลอดภัยได้ อีกทั้งวิวัฒนาการของการประมวลผลภาษาธรรมชาติในปัจจุบัน ทั้งแบบจำลองกลุ่มตัวแทนคำอย่าง WangchanBERTa และโมเดลภาษาขนาดใหญ่ตระกูล Gemini ล้วนมีรากฐานมาจากสถาปัตยกรรม Transformer ที่ใช้กลไกการให้ความสนใจต่อตนเอง (Vaswani et al., 2017) ซึ่งช่วยให้โมเดลสามารถจับความสัมพันธ์ของคำและความหมายเชิงบริบทในระดับข้อความยาวได้อย่างแม่นยำกว่าโครงสร้างแบบเดิม

งานวิจัยนี้มีวัตถุประสงค์หลักเพื่อปรับปรุงและพัฒนาประสิทธิภาพของระบบกึ่งอัตโนมัติสำหรับสกัดข้อมูลเหตุการณ์น้ำท่วมออนไลน์ โดยเปลี่ยนผ่านจากโครงสร้าง Python NER แบบเดิมมาเป็นเวิร์กโฟลว์กึ่งอัตโนมัติบนพื้นฐานสถาปัตยกรรม JavaScript ร่วมกับ n8n และประยุกต์ใช้ความสามารถของ Gemini API เพื่อช่วยลดระยะเวลาประมวลผล ลดภาระการเขียนโปรแกรมจัดการข้อมูลที่ไม่จำเป็น และยกระดับความถูกต้องแม่นยำในการระบุพื้นที่เวลา ตลอดจนระดับความรุนแรงของเหตุการณ์น้ำท่วมเพื่อใช้ในการสนับสนุนข้อมูลการตัดสินใจและแจ้งเตือนภัยได้อย่างเสถียรและมีประสิทธิภาพยิ่งขึ้น

วิธีดำเนินการวิจัย

1. สถาปัตยกรรมระบบและการไหลของข้อมูล เพื่อให้เห็นภาพสถาปัตยกรรมของระบบกึ่งอัตโนมัติที่เสนอและเส้นทางการไหลของข้อมูลตั้งแต่อินพุตจนถึงการสร้างภาพสารสนเทศ แผนภูมิสถาปัตยกรรมระบบ (รูปที่ 1) ด้านล่างจำลองขั้นตอนการไหลของข้อมูลหลัก 3 ชั้น ได้แก่ ชั้นนำเข้าข้อมูล ชั้นประมวลผล และชั้นแสดงผลลัพธ์



รูปที่ 1 แผนภาพโครงสร้างและทิศทางการไหลของข้อมูลในระบบกึ่งอัตโนมัติที่เสนอ

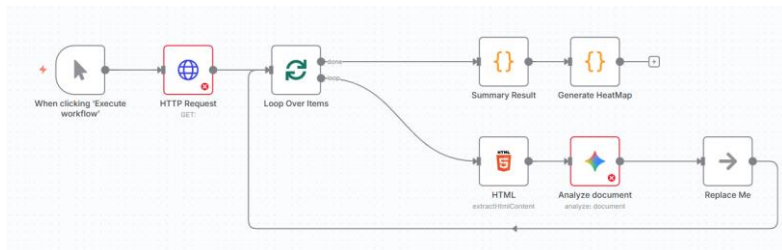
2. การวิเคราะห์เปรียบเทียบเชิงสถาปัตยกรรมระหว่างระบบเดิมและระบบใหม่ ความแตกต่างทางโครงสร้างการทำงานระหว่างระบบเดิม (ขลิต โควีระวงศ์ และคณะ, 2568) ที่พัฒนาขึ้นด้วยภาษา Python และระบบใหม่ที่เปลี่ยนมาใช้ JavaScript บน Node.js สามารถทำความเข้าใจได้ผ่านลักษณะการจัดสรรทรัพยากรคอมพิวเตอร์ เนื่องจากกระบวนการทำงานหลัก ทั้งขั้นตอนการดึงข้อมูลหน้าเว็บข่าว (Web Scraping) และการส่งข้อมูลข่าวดิบไปประมวลผลที่ Gemini API ล้วนจัดเป็นงานประเภทที่จำกัดด้วยประสิทธิภาพของสัญญาณเข้า/ออก ซึ่งความเร็วโดยรวมจะขึ้นอยู่กับการรอคอยการตอบรับทางเครือข่ายอินเทอร์เน็ตเป็นหลัก

ในระบบเดิมที่ใช้ Python Script ตัวโปรแกรมจะประมวลผลแบบซิงโครนัสและบล็อกสัญญาณเข้า/ออก ซึ่งกระบวนการทำงานจะทำได้ทีละบรรทัดคำสั่งตามลำดับ เมื่อระบบส่งคำขอดึงข้อมูลหน้าเว็บหรือส่งข่าวไปวิเคราะห์ที่ API เซรตการประมวลผลหลักจะต้องหยุดรอจนกว่าเซิร์ฟเวอร์ปลายทางจะส่งผลลัพธ์กลับมา ส่งผลให้มีระยะเวลาว่างเปล่าของซีพียูสะสมค่อนข้างสูง และนำไปสู่ความล่าช้าโดยรวมเมื่อต้องจัดการกับชุดข้อมูลข่าวปริมาณมาก

สำหรับระบบใหม่ที่พัฒนาขึ้นบน Node.js ตัวระบบใช้ประโยชน์จากกลไกของสถาปัตยกรรมแบบขับเคลื่อนด้วยเหตุการณ์ ร่วมกับการประมวลผลแบบอะซิงโครนัสและไม่บล็อกการทำงาน ผ่านระบบ Event Loop ทำให้อุปกรณ์สามารถยิงคำร้องขอ Web Scraping หรือเรียก API ออกไปได้พร้อมกันจำนวนมากในลักษณะพร้อมกัน โดยไม่ต้องรอให้คำขอแรกได้รับคำตอบกลับมาก่อน กลไกการจัดการงานแบบอะซิงโครนัสนี้ทำให้ระยะเวลาที่โปรแกรมต้องสูญเสียไปกับการรอคอยสัญญาณเครือข่ายลดลงอย่างเห็นได้ชัด ซึ่งสะท้อนออกมาในผลลัพธ์การทดลองประมวลผลข่าวทั้งหมด 555 ข่าวที่ใช้เวลารวมลดลงอย่างมีนัยสำคัญ เพื่อเป็นการปรับปรุงกระบวนการ โดยพยายามลดความซับซ้อนและเพิ่มความแม่นยำของกระบวนการวิจัย จึงได้พัฒนาระบบใหม่โดยประมวลผลโดยใช้ภาษา JavaScript และใช้งานร่วมกับ Node.js ระบบจะดึงเนื้อหาข่าวจาก URL และสกัดข้อความเหมือนกับระบบเดิม และส่งข้อมูลต่อไปยัง Gemini ผ่าน API (Google Gemini Team, 2023) ซึ่งทำหน้าที่สกัดข้อมูลและทำความสะอาดข้อมูล เพื่อระบุวันที่ สถานที่ จังหวัด และระดับความรุนแรงของเหตุการณ์ รวมถึงสรุปเนื้อหาของข่าวจากนั้นนำข้อมูลที่ได้นำไปจัดทำ Heat Map, Pivot Table หรือ Dashboard เพื่อใช้ในการวิเคราะห์และตัดสินใจเชิงพื้นที่ได้แบบกึ่งเรียลไทม์ สุดท้ายเมื่อนำไปใช้งานจริงจะใช้ n8n ซึ่งเป็นแพลตฟอร์ม workflow automation แบบ open-source เป็นผู้ติดต่อกับผู้ใช้งาน

ในการประเมินประสิทธิภาพของระบบใหม่ ได้มีการทดลองโดยใช้ชุดข้อมูลข่าวจำนวน 555 ข่าวที่รวบรวมในปี 2567 ซึ่งเป็นชุดข้อมูลเดียวกันกับที่ใช้ในระบบเดิม เพื่อเปรียบเทียบระยะเวลาในการประมวลผล ความแม่นยำของการสกัดข้อมูล ความสะดวกในการใช้งาน และคุณภาพของข้อมูลที่ได้ ผลจากการทดลองจะถูกวิเคราะห์เชิงปริมาณและเชิงคุณภาพ เพื่อแสดงให้เห็นถึงศักยภาพของระบบที่เสนอในการปรับปรุงกระบวนการสกัดข้อมูลข่าวภัยพิบัติให้ทันสมัย ยืดหยุ่น และนำไปใช้งานได้จริงในบริบทของการบริหารจัดการภัยพิบัติในประเทศไทย

3. การออกแบบเวิร์กโฟลว์อัตโนมัติบน n8n เพื่อควบคุมขั้นตอนการดึงและประมวลผลข้อมูลภัยพิบัติให้ทำงานประสานกันอย่างมีประสิทธิภาพโดยไม่ต้องเขียนโค้ดจำนวนมาก ระบบที่พัฒนาขึ้นได้ใช้งาน n8n ในฐานะแพลตฟอร์มควบคุมระบบงานอัตโนมัติ โดยใช้เป็นตัวควบคุมโฟลว์เชิงภาพ เพื่อเชื่อมโยง Node.js Web Scraper และโมเดลภาษาขนาดใหญ่ รายละเอียดและลำดับการประมวลผลข้อมูลแสดงไว้ดังรูปที่ 2



รูปที่ 2 หน้าจอแสดงกระบวนการทำงานและลำดับการเชื่อมต่อเวิร์กโฟลว์อัตโนมัติบนแพลตฟอร์ม n8n

ลำดับขั้นตอนการทำงานภายในเวิร์กโฟลว์บน n8n ประกอบด้วย 4 ขั้นตอนย่อย โดยเริ่มจากการกำหนดจุดเริ่มต้นการทำงาน ทำหน้าที่ตรวจจับข่าวสารใหม่และนำเข้า URL ของแหล่งข่าวจากอินพุตที่กำหนด เช่น ผ่านทางตารางรายการ เพิ่มข้อมูลภายนอก หรือการเชื่อมต่อ Webhook จากนั้นต่อด้วยการทำเว็บสแครปปิง โดยเป็น Node ประมวลผลภายในระบบงาน (โดยเขียนด้วยภาษา JavaScript/Node.js) ทำหน้าที่เข้าไปดาวน์โหลดข้อมูลหน้าเว็บ HTML ตาม URL นำเข้า จากนั้นแยกส่วนประกอบข้อความดิบออกจากโครงสร้างเว็บ และคัดกรองแท็กที่ไม่เกี่ยวข้องออกไป จากนั้นต่อด้วยการประมวลผลวิเคราะห์ธรรมชาติ โดยเชื่อมต่อไปยังตัวแบบภาษาขนาดใหญ่ (LLM) ด้วย Gemini API เพื่อสกัดมิติของข้อมูลประเภท

วันที่เกิดเหตุ สถานที่เกิดเหตุเฉพาะจุด จังหวัด และระดับความรุนแรงของภัยพิบัติผ่านกลไก Zero-shot Prompting และส่งออกในรูปแบบโครงสร้าง JSON และสุดท้ายการส่งออกและบันทึกข้อมูล โดยนำข้อมูลที่สกัดเสร็จสิ้นแล้วไปเก็บลงในระบบฐานข้อมูลอัจฉริยะ (JSON/Sheet Database) เพื่อเตรียมส่งต่อไปสร้างข้อมูลภาพสารสนเทศ

4. วิธีการทดลองและประเมินผลในการประเมินประสิทธิภาพของระบบกึ่งอัตโนมัติที่เสนอ ได้ดำเนินการทดลองเปรียบเทียบกับระบบเดิม (Python Script ร่วมกับแบบจำลองภาษา WangchanBERTa) บนชุดข้อมูลเดียวกัน โดยมีขอบเขตชุดข้อมูลและรายละเอียดสเปกเครื่องประมวลผลคอมพิวเตอร์ดังนี้

ชุดข้อมูลทดสอบ ประกอบด้วยข้อมูลข่าวอุทกภัยออนไลน์ในประเทศไทยที่รวบรวมขึ้นระหว่างปี พ.ศ. 2567 จำนวน 555 ข่าวประเด็น การประเมินเปรียบเทียบ 3 ด้าน ได้แก่ ระยะเวลาการประมวลผล โดยวัดระยะเวลาประมวลผลรวมในทุกขั้นตอน ตั้งแต่สกัด URL จนถึงการจัดเตรียมโครงสร้างฐานข้อมูลเสร็จสิ้น ประสิทธิภาพความถูกต้อง โดยวัดค่าคะแนน Precision, Recall และ F1-Score ของการสกัดพารามิเตอร์เชิงพื้นที่และเวลา เทียบกับเฉลี่ยของมนุษย์ จากกลุ่มตัวอย่างสุ่มจำนวน 100 ข่าว วัดความสามารถการสกัดมิติแวดล้อมโดยวิเคราะห์ความสามารถสกัดรายละเอียดเพิ่มเติม เช่น สรุปความเสียหายเชิงตัวเลข หรือระดับความรุนแรง ซึ่งข้อจำกัดของระบบเดิมไม่สามารถประมวลผลได้

ข้อมูลสเปกเครื่องประมวลผลและสภาพแวดล้อมที่ใช้จริง

เพื่อความเที่ยงตรงและยึดถือหลักการทดลองซ้ำได้ การทดสอบประสิทธิภาพของทั้งสองระบบดำเนินการภายใต้ฮาร์ดแวร์และซอฟต์แวร์บนเครื่องประมวลผลเดียวกัน ประกอบด้วยหน่วยประมวลผลกลาง Intel Core i5-6500T CPU @ 2.50 GHz, หน่วยความจำหลัก 12.0 GB DDR4 (2,133 MHz) และระบบปฏิบัติการ Microsoft Windows 10 Pro 64-bit โดยเครื่องทดสอบใช้หน่วยประมวลผลกราฟิกแบบออนบอร์ด Intel HD Graphics 530 (128 MB) โดยไม่มีการติดตั้งการ์ดแสดงผลแยก (Discrete GPU) ส่งผลให้ระบบเดิมต้องรันแบบจำลอง WangchanBERTa ในโหมดคำนวณด้วยซีพียูทั้งหมด (CPU-only Inference) ในขณะที่ระบบใหม่ที่น่าเสนอทำงานบนรันไทม์ Node.js v22.21.1 ร่วมกับเครื่องมือควบคุมลำดับงาน n8n Workflow Automation Engine 2.3 เพื่อส่งคำร้องสกัดข้อมูลไปยัง Google Gemini API

ผลการวิจัยและอภิปรายผล

1. เปรียบเทียบเวลาในการประมวลผล จากการทดลองใช้งานจริงพบว่า ระบบเดิมที่เป็น Python Script ร่วมกับ NER ประมวลผลที่ 1125 วินาที (ประมาณ 19 นาที) สำหรับระบบใหม่ที่พัฒนาโดยใช้ JavaScript ร่วมกับ Gemini สามารถลดระยะเวลาการประมวลผลทั้งหมด เหลือเพียง 788 วินาที (ประมาณ 13 นาที) ในการสกัดข้อมูลจากข่าวจำนวน 555 ข่าว ปัจจัยที่มีผลต่อการคำนวณคือ การสกัดข้อมูลจาก URL หากทำด้วยภาษา JavaScript จะทำได้เร็วกว่า Python นอกจากนั้นการสกัดข้อมูลด้วย NER สำหรับข้อมูลวันที่ สถานที่ ทำได้ช้ากว่าการใช้งาน API ของ LLM กระบวนการแบบอัตโนมัตินี้ช่วยลดขั้นตอน manual เช่น การจัดการ HTML การเรียกใช้สคริปต์แยกส่วน และการสรุปผลแบบแมนนวล ทำให้ระบบมีความต่อเนื่องในการทำงาน ลดภาระของผู้ใช้งาน และสามารถนำไปปรับใช้ในบริบทของงานติดตามสถานการณ์น้ำท่วมในระดับวันต่อวันได้จริง ดังตารางที่ 1

ตารางที่ 1 เวลาในการประมวลผล (durations) หน่วยเป็นวินาที (seconds)

ขั้นตอน	Python Jupyter / NER (วินาที)	JavaScript / LLM (วินาที)
1. สกัดข้อมูล URL จากเว็บ	2	2
2. สกัดข้อมูลจาก URL	252	105
3. NER / LLM	862	672
4. pivot table	8	8
5. การสร้าง Heatmap	1	1
รวมเวลา (วินาที)	1125	788

จากตารางที่ 1 พบว่าในขั้นตอนย่อยที่ 3 (NER / LLM) ระบบใหม่สามารถลดระยะเวลาในการประมวลผลลงจาก 862 วินาที เหลือ 672 วินาที ซึ่งมีนัยสำคัญที่น่าสนใจเนื่องจากการเรียกใช้งานโมเดลภาษาขนาดใหญ่ (LLM) ผ่าน API ภายนอกมักจะเผชิญกับความหน่วงทางเครือข่ายที่สูงกว่าการประมวลผลแบบเบ็ดเสร็จภายในเครื่อง เหตุผลหลักที่ทำให้ความเร็วโดยรวมของระบบใหม่เพิ่มขึ้นมีสองประการสำคัญ

ประการแรกคือการประยุกต์ใช้เทคนิค Asynchronous Concurrency ในสภาพแวดล้อม Node.js โดยแทนที่การเรียก API ที่ละหนึ่งข่าวกระทำแบบเรียงลำดับ ตัวระบบได้

ปรับใช้สถาปัตยกรรมแบบไม่บล็อกการทำงานเพื่อส่งคำขอ HTTP Requests ไปยัง Gemini API แบบขนานพร้อมกันหลายรายการโดยใช้กลไก Promise ส่งผลให้การสะสมความหน่วงเครือข่ายเกิดขึ้นในเวลาทับซ้อนกัน ต่างจากระบบ Python Script แบบเดิมที่ส่งคำร้องขอประมวลผลและรอผลลัพธ์แบบทีละข้อไปจนครบ 555 ข้อ ซึ่งส่งผลให้ความหน่วงเครือข่ายสะสมเป็นเส้นตรง

ประการที่สองคือ ภาระงานในการประมวลผลบนทรัพยากรท้องถิ่น เนื่องจากระบบเดิมอาศัยการโหลดตัวแบบภาษาไทย WangchanBERTa และประมวลผลโมเดลโครงข่ายประสาทในเครื่องคอมพิวเตอร์ ซึ่งการประมวลผลโมเดลประเภท Transformer นี้สร้างภาระงานคำนวณเวกเตอร์ขนาดใหญ่แก่หน่วยประมวลผลภายในเครื่องอย่างรุนแรงหากขาดการสนับสนุนหน่วยประมวลผลกราฟิก ที่มีประสิทธิภาพสูง ในขณะที่ระบบใหม่เลือกโยนภาระการคำนวณทั้งหมดไปประมวลผลบนระบบคลาวด์คลัสเตอร์ของ Google ที่มีขีดความสามารถการคำนวณระดับสูงกว่า ผ่านสถาปัตยกรรม API ทำให้ความเร็วในการคำนวณของฝั่งเซิร์ฟเวอร์ระบบคลาวด์ชัดเจนและก้าวข้ามความหน่วงจากการรับส่งข้อมูลเครือข่ายได้อย่างมีประสิทธิภาพ

2. ความแม่นยำและประสิทธิภาพในการสกัดข้อมูลตัวตนเฉพาะในการประเมินความถูกต้องแม่นยำของกระบวนการสกัดข้อมูลตัวตนเฉพาะทางพื้นที่และเวลา งานวิจัยนี้ได้กำหนดกระบวนการวัดผลที่เป็นวิทยาศาสตร์ โดยสุ่มกลุ่มตัวอย่างแบบง่าย (Simple Random Sampling) จำนวน 100 ข้อสารภัยพิบัติจากฐานข้อมูลทั้งหมด 555 ข้อ เพื่อนำมาผ่านขั้นตอนการตรวจสอบแบบแมนนวลโดยละเอียดเกณฑ์การประเมินในการตัดสินใจว่าข้อมูลถูกต้อง ถูกตั้งไว้ที่ระดับการตรงกันแบบทุกตัวอักษร ซึ่งหมายถึงข้อมูลตัวตนที่สกัดออกมาจะต้องมีความถูกต้องสมบูรณ์ทั้งรูปคำสะกดของสถานที่และรูปแบบวันที่ประวัติการณ์โดยไม่มีส่วนขาดหรือส่วนเกิน โดยใช้ผู้ประเมินที่เป็นมนุษย์ ร่วมกันตรวจสอบจำนวน 3 คน ประกอบด้วยผู้เชี่ยวชาญด้านวิทยาการคอมพิวเตอร์และประมวลผลภาษาธรรมชาติ 2 คน ร่วมกับผู้เชี่ยวชาญด้านภูมิสารสนเทศศาสตร์ 1 คน หากมีความเห็นขัดแย้งในผลการวิเคราะห์ จะตัดสินผลลัพธ์ผ่านกลไกการลงมติเสียงข้างมาก

การเปรียบเทียบประสิทธิภาพมาตรฐานอันประกอบไปด้วย ค่าความแม่นยำ (Precision) ค่าความระลึก (Recall) และค่าคะแนนเอฟวัน (F1-Score) ระหว่างระบบเดิมที่ทำงานบน WangchanBERTa กับระบบใหม่ที่ทำงานบน Gemini API แสดงไว้ในตารางที่ 2

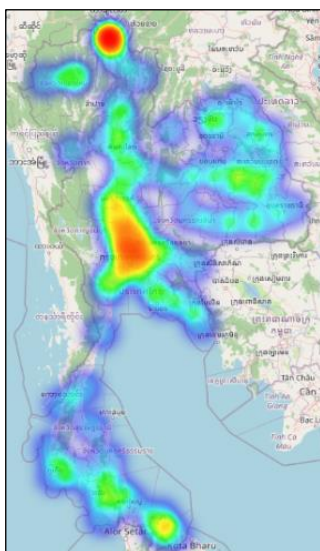
ตารางที่ 2 เปรียบเทียบประสิทธิภาพการสกัดข้อมูล Entity ของระบบเดิมและระบบใหม่
(หน่วย: ร้อยละ)

ข้อมูลจำเพาะ (Entities)	ระบบเดิม (WangchanBERTa)	ระบบใหม่ (Gemini API)
	Precision / Recall /	Precision / Recall /
	F1-Score	F1-Score
1. วันที่ (Date)	84.0 / 80.0 / 82.0	95.0 / 92.0 / 93.5
2. สถานที่ (Location)	78.0 / 75.0 / 76.5	90.0 / 88.0 / 89.0
3. จังหวัด (Province)	88.0 / 85.0 / 86.5	97.0 / 95.0 / 96.0
ค่าเฉลี่ยรวม (Average)	83.3 / 80.0 / 81.7	94.0 / 91.7 / 92.8

จากตารางที่ 2 พบว่าระบบใหม่ที่ประยุกต์ใช้งานโมเดลภาษาขนาดใหญ่ (Gemini API) ผ่านเวิร์กโฟลว์อัตโนมัติสามารถทำคะแนน F1-Score เฉลี่ยสูงถึงร้อยละ 92.8 ซึ่งมีประสิทธิภาพดีกว่าระบบเดิมที่ได้คะแนนร้อยละ 81.7 อย่างมีนัยสำคัญ เมื่อพิจารณาในรายละเอียดพบว่า ประสิทธิภาพในการสกัดข้อมูลประเภทสถานที่ มีความแตกต่างกันอย่างชัดเจนที่สุด โดยระบบเดิมได้ค่า F1-Score เพียงร้อยละ 76.5 เนื่องจากโมเดลภาษาขนาดเล็กอย่าง WangchanBERTa มักประสบปัญหาการแบ่งขอบเขตของชื่อสถานที่ในภาษาไทยที่ไม่มีการเว้นวรรคและการสลับบริบทคำกำกวม (เช่น สกัดคำว่า "ปาย" ผิดพลาดระหว่างชื่อสถานที่หรือส่วนหนึ่งของคำกริยา) ในขณะที่ระบบใหม่ (Gemini API) สามารถประเมินความสัมพันธ์เชิงบริบทได้ลึกซึ้งกว่า ทำให้สกัดสถานที่ได้อย่างถูกต้องใกล้เคียงกับเกณฑ์ Exact Match และช่วยส่งต่อข้อมูลพิกัดเชิงภูมิศาสตร์ที่ครบถ้วนมากขึ้นเพื่อนำไปสร้างแผนที่ความร้อนได้ตรงกับสภาพความเป็นจริง นอกจากนี้ ในส่วนของการสกัดข้อมูลจังหวัดซึ่งมีคลังรายการจำกัด ทั้งสองระบบสามารถทำคะแนนได้ดี แต่ Gemini API ยังคงมีขีดความสามารถที่สูงกว่าในกรณีที่โครงสร้างประโยคข่าวมีความซับซ้อนหรือใช้ชื่อย่อจังหวัดในเนื้อหาข่าวสารภัยพิบัติ

3. ความครอบคลุมของข้อมูล ระบบใหม่แสดงศักยภาพในการดึงข้อมูลเหตุการณ์ที่ระบบเดิมไม่สามารถจับได้ โดยเฉพาะข้อมูลที่อยู่ในรูปแบบภาษาธรรมชาติซึ่งมีความหลากหลายสูง เช่น ข่าวที่ใช้คำว่า “น้ำระบายไม่ทัน” หรือ “หมู่บ้านถูกตัดขาด” ซึ่งระบบ NER เดิมอาจไม่สามารถตีความว่าเป็นเหตุการณ์น้ำท่วมได้ ขณะที่ Gemini สามารถตีความและจัดประเภทเหตุการณ์ได้แม่นยำกว่า นอกจากนี้ยังสามารถรวมเหตุการณ์ซ้อนซึ่งเกิดในวันเดียวกันจากหลายแหล่งข่าวเข้าเป็นข้อมูลเดียวได้โดยอัตโนมัติ ลดความซ้ำซ้อนของข้อมูลและช่วยให้ผลการวิเคราะห์มีความครบถ้วนมากขึ้น

4. การนำเสนอข้อมูลเชิงพื้นที่ และสารสนเทศเชิงพื้นที่ ข้อมูลเชิงพื้นที่และเวลา ที่สกัดได้สำเร็จจากระบบใหม่ (ตารางที่ 3) จะถูกนำเสนอและจัดเก็บในรูปแบบโครงสร้างข้อมูลมาตรฐาน ประกอบด้วยรายละเอียดทางภูมิศาสตร์ เช่น ชื่อหมู่บ้าน ตำบล อำเภอ และจังหวัดที่เกิดเหตุ ข้อมูลเหล่านี้มีความพร้อมสำหรับการนำเข้าสู่ระบบสารสนเทศภูมิศาสตร์ (GIS) เพื่อจัดทำแผนที่ความร้อน แสดงความหนาแน่นและสถิติความถี่ของเหตุการณ์อุทกภัยจากแหล่งข้อมูลออนไลน์ได้อย่างมีประสิทธิภาพ ดังแสดงในรูปที่ 3



รูปที่ 3 แผนที่ความร้อน (Heat Map) แสดงสถิติความถี่และการกระจุกตัวของเหตุการณ์อุทกภัยจากการประมวลผลข่าวออนไลน์ ประจำปี พ.ศ. 2567

จากการวิเคราะห์การกระจายตัวทางภูมิศาสตร์ในแผนที่ความร้อน (รูปที่ 3) พบว่าพื้นที่ที่มีการกระจุกตัวของเหตุการณ์อุทกภัยในระดับสูง ซึ่งแสดงด้วยสัญลักษณ์โทนสีแดงและสีส้มเข้ม มีรายละเอียดดังนี้

พื้นที่ภาคเหนือตอนบน (จังหวัดเชียงรายและเชียงใหม่) สะท้อนให้เห็นถึงเหตุอุทกภัยครั้งใหญ่ในช่วงเดือนสิงหาคมและกันยายน พ.ศ. 2567 ซึ่งได้รับอิทธิพลทางอ้อมจากพายุหมุนเขตร้อน "ยะกิ" (Yagi) ส่งผลให้เกิดน้ำป่าไหลหลากล้นตลิ่งแม่น้ำสายเข้าท่วมพื้นที่อำเภอแม่สาย จังหวัดเชียงราย ตลอดจนการล้นเอ่ออย่างรุนแรงของแม่น้ำปิงที่เอ่อท่วมย่านเศรษฐกิจในตัวเมืองเชียงใหม่หลายระลอก

พื้นที่ลุ่มน้ำยม (จังหวัดสุโขทัยและพิษณุโลก) ปรากฏความถี่ของการรายงานข่าวหนาแน่นในระดับสูง เนื่องจากลุ่มน้ำยมตอนล่างไม่มีเขื่อนหรืออ่างเก็บน้ำขนาดใหญ่สำหรับกักเก็บและชะลอน้ำหลากที่ไหลมาจากภาคเหนือตอนบน ส่งผลให้น้ำเอ่อล้นตลิ่งและคันดินกั้นน้ำชำรุดเสียหาย จนเข้าท่วมพื้นที่เกษตรกรรมและชุมชนริมน้ำเป็นบริเวณกว้างเป็นประจำทุกปี

พื้นที่ลุ่มน้ำเจ้าพระยาตอนล่าง (จังหวัดพระนครศรีอยุธยา) ในฐานะพื้นที่ลุ่มต่ำที่เป็นทางผ่านน้ำหลากจากภาคกลางตอนบนก่อนระบายออกสู่อ่าวไทย พื้นที่บริเวณนี้ได้รับผลกระทบโดยตรงจากการบริหารน้ำและการเร่งระบายน้ำของเขื่อนเจ้าพระยา ส่งผลให้แม่น้ำเจ้าพระยาและลำน้ำสาขาเอ่อล้นตลิ่งเข้าท่วมชุมชนนอกคันกั้นน้ำ รวมถึงพื้นที่ลุ่มต่ำริมน้ำเป็นระยะเวลายาวนาน

ผลการวิเคราะห์เชิงพื้นที่ข้างต้นมีความสอดคล้องอย่างมีนัยสำคัญกับรายงานสรุปสถานการณ์อุทกภัยประจำปี พ.ศ. 2567 ของกรมป้องกันและบรรเทาสาธารณภัย (ปภ.) ระบุว่าจังหวัดเชียงราย เชียงใหม่ สุโขทัย และพระนครศรีอยุธยา เป็นกลุ่มพื้นที่ที่ประสบอุทกภัยรุนแรงและยาวนานที่สุดในรอบปี (ศูนย์ข้อมูลสาธารณภัย, 2567) ความสอดคล้องดังกล่าวช่วยยืนยันความถูกต้องเชิงเนื้อหาพื้นที่และเวลา ของระบบสกัดข้อมูลกึ่งอัตโนมัติที่พัฒนาขึ้นว่าสามารถสะท้อนภาพสถานการณ์ภัยพิบัติในความเป็นจริงได้อย่างแม่นยำ ซึ่งจะเป็ประโยชน์อย่างยิ่งต่อการนำไปประยุกต์ใช้สนับสนุนระบบวางแผนและแจ้งเตือนภัยล่วงหน้าในพื้นที่เสี่ยงภัยได้อย่างมีประสิทธิภาพ

5. ความสะดวก การเปลี่ยนมาใช้โมเดลภาษาขนาดใหญ่ ในการสกัดและวิเคราะห์ข้อมูลแทนการพัฒนา NER เอง ช่วยเพิ่มความยืดหยุ่นและลดความซับซ้อนในการพัฒนาอย่างมาก นักพัฒนาไม่จำเป็นต้องสร้างหรือปรับแต่งโมเดลสำหรับแต่ละประเภทข้อมูลอีกต่อไป เพียงแค่ใช้คำสั่ง หรือ prompt ที่เข้าใจง่ายก็สามารถดึงข้อมูลที่ต้องการได้ทันที ทำให้ได้สะดวกและกระบวนการทำงานเร็วขึ้น นอกจากนี้ LLM ยังทำงานผ่าน API ได้โดยตรง ทำให้สามารถใช้งานได้กับหลากหลายภาษาโปรแกรมมิ่ง ไม่จำกัดแค่ Python แต่ยังรวมถึง JavaScript หรือภาษาอื่น ๆ ด้วย ส่งผลให้ทีมพัฒนาสามารถผนวกความสามารถของ AI เข้ากับระบบที่มีอยู่ได้อย่างรวดเร็ว ทำให้การพัฒนาเป็นไปอย่างคล่องตัวและเปิดกว้างสำหรับนักพัฒนาจากทุกสายงาน

6. ข้อดีของการใช้ n8n JavaScript Gemini การผสานการทำงานระหว่าง n8n JavaScript Gemini แสดงให้เห็นถึงศักยภาพของระบบอัตโนมัติในการจัดการข้อมูลข่าว

ภัยพิบัติอย่างมีประสิทธิภาพ โดยใช้ n8n เป็นส่วนติดต่อผู้ใช้และช่วยลดภาระในการเขียนโค้ดแบบดั้งเดิม ด้วยการสร้าง workflow แบบ visual drag-and-drop ที่สามารถตั้งค่า trigger และควบคุมการประมวลผลข้อมูลได้อย่างยืดหยุ่น ขณะที่ JavaScript และ Gemini ทำหน้าที่ในการประมวลผลข้อความและสกัดข้อมูลสำคัญจากข่าวในลักษณะที่เข้าใจบริบทภาษาไทยได้อย่างลึกซึ้ง การใช้เครื่องมือทั้งสองนี้ร่วมกันช่วยลดเวลาในการพัฒนา เพิ่มความแม่นยำของผลลัพธ์ และสามารถขยายขีดความสามารถของระบบได้โดยไม่ต้องปรับแต่งโค้ดที่ซับซ้อนมากนัก นอกจากนี้การประมวลผลผลภาษา JavaScript ได้ลดความล่าช้าในการประมวลผลอีกด้วย และการใช้งาน Gemini ก็ทำให้ไม่ยึดติดว่าการประมวลผลภาษาธรรมชาติจะต้องเป็น Python เท่านั้น

7. ความเป็นไปได้ในการนำไปต่อยอดกับระบบแจ้งเตือนอัตโนมัติ ผลลัพธ์จากระบบที่พัฒนาขึ้นสามารถนำไปต่อยอดได้อย่างชัดเจนในบริบทของระบบแจ้งเตือนภัยอัตโนมัติ เช่น การตั้งค่าให้ส่งข้อความแจ้งเตือนเหตุการณ์น้ำท่วมผ่านไลน์ OA หรือ Telegram โดยอ้างอิงจากข้อมูลข่าวที่สกัดใหม่แบบเรียลไทม์ และสามารถประเมินพื้นที่เสี่ยงผ่านข้อมูลเชิงพื้นที่ที่สกัดได้จากข่าว ระบบยังสามารถทำงานร่วมกับเครื่องมือตรวจวัดภาคสนาม เช่น IoT Sensor หรือข้อมูลจากกรมอุตุนิยมวิทยา เพื่อสร้างการแจ้งเตือนที่มีความแม่นยำสูง และมีศักยภาพในการประยุกต์ใช้กับภัยพิบัติอื่น ๆ เช่น ดินถล่มหรือไฟป่าในอนาคตได้อีกด้วย

8. ประเด็นด้านความปลอดภัย ความเป็นส่วนตัว และข้อจำกัดของ LLM แม้ระบบที่นำเสนอจะมีศักยภาพสูง แต่ก็ยังมีข้อพึงระวังด้านความปลอดภัยและความเป็นส่วนตัว เช่น การใช้ LLM อย่าง Gemini ต้องอาศัยการส่งข้อมูลผ่านอินเทอร์เน็ต ซึ่งอาจมีความเสี่ยงต่อการรั่วไหลของข้อมูล หากไม่มีมาตรการควบคุมที่เหมาะสม นอกจากนี้ LLM ยังมีข้อจำกัดด้านความแม่นยำในกรณีที่มีข้อมูลมีความกำกวม หรือใช้ภาษาท้องถิ่นเฉพาะกลุ่ม และยังไม่สามารถรับรองผลลัพธ์ได้ 100% ซึ่งเป็นข้อพึงระวังสำคัญอีกข้อในการประยุกต์ใช้โมเดลภาษาขนาดใหญ่ (Ji et al., 2023) โดยเฉพาะเมื่อใช้ในงานที่มีผลกระทบต่อการตัดสินใจเชิงนโยบาย ดังนั้นควรมีการออกแบบระบบให้สามารถตรวจสอบย้อนกลับได้ และใช้ LLM เป็นเพียงเครื่องมือเสริมการตัดสินใจมากกว่าการพึ่งพาเพียงอย่างเดียว นอกจากนี้ค่าใช้จ่ายหลักของการใช้งาน LLM ที่ผู้ใช้งานส่วนใหญ่ต้องเผชิญ โดยปกติจะคิดตามจำนวนโทเคน (tokens) ที่ใช้ ซึ่งก็คือหน่วยย่อยของข้อความ (เช่น คำหรือส่วนของคำ) ยิ่งป้อนข้อมูลเข้าไปมาก หรือให้โมเดลสร้างข้อมูลออกมามาก ค่าใช้จ่ายก็จะสูงขึ้นตามไปด้วย

9. เนื่องจากต้นทุนและประสิทธิภาพการประมวลผลด้วยโมเดลภาษาขนาดใหญ่ (LLM) แปรผันโดยตรงตามจำนวนโทเคน (Tokens) ทั้งในฝั่งขาเข้า (Input Tokens) และขาออก (Output Tokens) การวิเคราะห์ในส่วนนี้จึงสรุปเชิงปริมาณครอบคลุม 2 ประเด็นหลัก คือ ประเด็นปริมาณโทเคนที่ใช้ในการประมวลผล และ ประเด็นราคาและค่าใช้จ่ายที่ต้องจ่าย

จากการทดสอบประมวลผลสกัดข้อมูลและสรุปเนื้อหาจากชุดข้อมูลข่าวอุทกภัย ทั้งหมดจำนวน 555 บทความ โดยใช้แบบจำลองการตัดคำของ Gemini พบว่าระบบใช้ปริมาณโทเคนรวมทั้งสิ้น 900,842 โทเคน (ประมาณ 0.90 ล้านโทเคน) โดยแบ่งเป็นปริมาณ ฝั่งขาเข้า จำนวน 872,318 โทเคน (คิดเป็นสัดส่วนสูงถึงร้อยละ 96.84) และโทเคนฝั่งขาออกจำนวน 28,524 โทเคน (คิดเป็นร้อยละ 3.16) ซึ่งแสดงให้เห็นว่าภาระการประมวลผลส่วนใหญ่อยู่ในฝั่งข้อมูลขาเข้า ขณะที่ฝั่งข้อมูลขาออกใช้โทเคนเพียงส่วนน้อยเนื่องจากสถาปัตยกรรมคำสั่งกำหนดให้โมเดลตอบกลับเฉพาะโครงสร้างข้อมูล JSON ที่มีความกระชับสูง

จากการตรวจสอบอัตราค่าบริการมาตรฐานของการใช้งาน Gemini API สำหรับโมเดล Gemini 2.5 Flash (Google Developers, 2024; กำหนดอัตราบริการโทเคนฝั่งขาเข้า \$0.300 USD/1M Tokens และฝั่งขาออก \$2.500 USD/1M Tokens อัตราแลกเปลี่ยนอ้างอิง 1 USD = 36.00 THB) คิดเป็นค่าบริการฝั่งขาเข้า สำหรับชุดข้อมูลข่าวทั้งสิ้น 555 ข่าว (872,318 โทเคน) เท่ากับ \$0.2617 USD (~9.42 บาทไทย) และค่าบริการฝั่งขาออก (28,524 โทเคน) เท่ากับ \$0.0713 USD (~2.57 บาทไทย) ส่งผลให้ต้นทุนค่าใช้จ่ายรวมทั้งสิ้นสำหรับการสกัดและสรุปข่าวข่าวท่วมตลอดทั้งชุดข้อมูล คิดเป็นเงินเพียง \$0.3330 USD หรือประมาณ 11.99 บาทไทย เท่านั้น (ตกเฉลี่ยเพียงประมาณ 0.0216 บาทต่อ 1 บทความข่าว)

ในด้านความเป็นไปได้เชิงเศรษฐศาสตร์และการนำไปประยุกต์ใช้งานจริง หากออกแบบระบบรับและประมวลผลข่าวสารกึ่งเรียลไทม์ ระบบจะสามารถทำงานภายใต้โควตาใช้งานฟรีของ Google AI Studio ได้โดยไม่มีค่าใช้จ่าย (0 บาท) นอกจากนี้ เมื่อเปรียบเทียบกับระบบ NER เดิมบน Python ที่ต้องมีภาระต้นทุนในการจัดหาและบำรุงรักษาเครื่องเซิร์ฟเวอร์พร้อมการ์ดประมวลผลกราฟิก (GPU) แบบ On-Premise ตลอด 24 ชั่วโมง สถาปัตยกรรมใหม่ที่ใช้ JavaScript (Node.js) ร่วมกับ n8n Workflow และ Gemini API จึงสามารถลดต้นทุนการดำเนินงานลงได้ และมีความคุ้มค่ายิ่งสำหรับหน่วยงานภาครัฐหรือองค์กรปกครองส่วนท้องถิ่นที่มีข้อจำกัดด้านงบประมาณ

สรุปผลการวิจัย

การวิจัยครั้งนี้แสดงให้เห็นถึงความก้าวหน้าอย่างมีนัยสำคัญในการปรับปรุงกระบวนการสกัดข้อมูลข่าวน้ำท่วมจากแหล่งข่าวออนไลน์ โดยการเปลี่ยนจากระบบแบบดั้งเดิมที่ต้องใช้โค้ดจำนวนมากและมีขั้นตอนซ้ำซ้อน มาเป็นระบบกึ่งอัตโนมัติที่ใช้เครื่องมือ n8n ร่วมกับ JavaScript และ Gemini ส่งผลให้สามารถลดเวลาในการประมวลผลลงได้อย่างมาก เพิ่มความแม่นยำในการสกัดข้อมูล และลดภาระของผู้ใช้งาน ระบบใหม่นี้ยังสามารถนำเสนอผลลัพธ์ในรูปแบบเชิงพื้นที่ที่ชัดเจนและต่อยอดได้ในระดับนโยบายหรือการปฏิบัติจริง

จากความสำเร็จในการใช้เทคนิคดังกล่าวกับข่าวน้ำท่วม งานวิจัยนี้เสนอให้ขยายระบบไปสู่การติดตามภัยพิบัติรูปแบบอื่น เช่น แผ่นดินไหว ดินถล่ม และไฟป่า ซึ่งมักมีการรายงานผ่านข่าวในทำนองเดียวกัน การฝึกโมเดล LLM เพิ่มเติมให้สามารถเข้าใจคำศัพท์เฉพาะของแต่ละประเภทภัยพิบัติจะช่วยเพิ่มความแม่นยำของระบบ และสามารถพัฒนาเป็นแพลตฟอร์มกลางสำหรับติดตามสถานการณ์ภัยพิบัติแบบครอบคลุม ทั้งในด้านเวลา พื้นที่ และความรุนแรง เพื่อให้ระบบมีประโยชน์อย่างเป็นรูปธรรมในระดับนโยบาย ควรมีการผนวกรวมข้อมูลที่สกัดได้จากข่าวเข้ากับระบบเตือนภัยที่มีอยู่ในระดับจังหวัดหรือองค์กรปกครองส่วนท้องถิ่น เช่น ศูนย์บัญชาการเหตุการณ์ หรือแอปพลิเคชันเตือนภัยของภาครัฐ การผนวกนี้สามารถทำได้ผ่าน API หรือ Dashboard กลางที่เชื่อมโยงกับข้อมูลเชิงพื้นที่และเชิงเวลาแบบเรียลไทม์ ซึ่งจะช่วยเพิ่มความเร็วในการรับรู้สถานการณ์และตัดสินใจเชิงรุกของหน่วยงานได้อย่างมีประสิทธิภาพมากขึ้น

กิตติกรรมประกาศ

ทีมวิจัยขอแสดงความขอบคุณต่อ สำนักงานคณะกรรมการส่งเสริมวิทยาศาสตร์ วิจัย และนวัตกรรม (สกสว.) ที่ได้ให้ทุนสนับสนุนการวิจัยประจำปีงบประมาณ 2567 นอกจากนี้ขอขอบคุณทุกท่านที่ได้ให้ข้อมูล ข้อเสนอแนะ และคำแนะนำอันมีค่า ซึ่งมีส่วนช่วยอย่างมากต่อความสำเร็จของการวิจัยครั้งนี้

เอกสารอ้างอิง

- ชวลิต โควีระวงศ์, รัตพล อ่างมณี, ไพรินทร์ มีศรี และ ดารวธ วัระพันธ์. (2568). *การพัฒนาและประยุกต์ใช้เทคนิคเว็บสแครปปิงและการประมวลผลภาษาธรรมชาติสำหรับการรวบรวมและวิเคราะห์ข้อมูลข่าวสารเกี่ยวกับน้ำท่วมในประเทศไทย*. ใน: การประชุมวิชาการระดับชาติและนานาชาติครั้งที่ 4 ด้านวิทยาศาสตร์และเทคโนโลยี. วันที่ 7 มีนาคม 2568. คณะวิทยาศาสตร์และเทคโนโลยี มหาวิทยาลัยราชภัฏเพชรบูรณ์, เพชรบูรณ์. 314-323.
- ศูนย์ข้อมูลสาธารณภัย. (2567). *สรุปสถิติสาธารณภัยประจำปี พ.ศ. 2567 ของกรมป้องกันและบรรเทาสาธารณภัย (ปภ.)*. สืบค้นเมื่อวันที่ 8 สิงหาคม 2569, จาก <https://datacenter.disaster.go.th/datacenter/cms/8670?id=132971>.
- Deelman, E., Vahi, K., Juve, G., Rynge, M., Callaghan, S., Maechling, P. J., Mayani, R., Chen, W., Ferreira da Silva, R., Livny, M., & Wenger K. (2015). Pegasus, a workflow management system for science automation. *Future Generation Computer Systems*, 46, 17-35. doi:10.1016/j.future.2014.10.008.
- Eze, E., Feyisetan, O., & Omojola, A. (2020). Named entity recognition for disaster management: A review and framework. *International Journal of Disaster Risk Reduction*, 48, 101612. doi:10.1016/j.ijdr.2020.101612.
- Google Developers. (2024). *Google AI Studio Pricing and Model Documentation*. Google AI for Developers. Retrieved June 25, 2026 from <https://ai.google.dev/pricing>.
- Google Gemini Team. (2023). *Gemini: A family of highly capable multimodal models*. Retrieved August 1, 2026 from <https://arxiv.org/pdf/2312.11805>.
- Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Jin Bang, Y., Madotto, A., & Fung, P. (2023). Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12), 1-38. doi:10.1145/3571730.
- Lei, K., Ma, Y., & Tan, Z. (2014). *Performance comparison and evaluation of Web development technologies in PHP, Python, and Node.js*. In: Proceedings of IEEE International Conference on Computational Science and Engineering, 19-21 December, 2014, NW Washington, DC United States. 661-668. doi:10.1109/CSE.2014.142.

- Lowphansirikul, L., Polpanumas, C., Jantrakulchai, N., & Nutanong, S. (2021). *WangchanBERTa: Pretraining transformer-based Thai Language Models*. Retrieved July 12, 2026 from <https://arxiv.org/pdf/2101.09635>.
- Makarov, I. S., Larin, D. V., Vorobeve, E. G., Emelin, D. P., & Kartashov, D. A. (2025). The impact of asynchronous and multithreaded query processing models on the performance of server-side web applications. *Software Systems and Computational Methods*, (1), 13–20. doi.org/10.7256/2454-0714.2025.1.73665.
- Phatthiyaphaibun, W., Chaovavanich, K., Polpanumas, C., Suriyawongkul, A., Lowphansirikul, L., Chormai, P., Limkonchotiwat, P., Suntornitip, T., & Udomcharoenchaikit, C. (2023). *PyThaiNLP: Thai Natural Language Processing in Python*. In: Proceedings of the 3rd Workshop for Natural Language Processing Open Source Software, December 6, 2023, 25-36. doi:10.18653/v1/2023.nlposs-1.4.
- Revilla-Romero, B., Hirpa, F. A., Thielen-del Pozo, J., Salamon, P., Pappenberger, F., & De Groeve, T. (2015). On the use of global flood forecasts and satellite-derived inundation maps for flood monitoring in data-sparse regions. *Remote Sensing*, 7(11), 15702-15728. doi:10.3390/rs71115702.
- Singrodia, V., Mitra, A., & Paul, S. (2019). *A review on web scraping and its applications*. In: International Conference on Computer Communication and Informatics (ICCCI), 23-25 January, 2019, Coimbatore, India. 1-6. doi:10.1109/ICCCI.2019.8821809.
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 5998-6008.
- Wei, X., Cui, X., Cheng, N., Wang, X., Zhang, X., Huang, S., Xie, P., Xu, J., Chen, Y., Zhang, M., Jiang, Y., & Han, W. (2023). *ChatIE: Zero-Shot Information Extraction via Chatting with ChatGPT*. Retrieved August 8, 2026 from <https://arxiv.org/pdf/2302.10205>.