v1.0

AI Series: Pocket Guide

คู่มือ
Prompt Engineering

ใช้งาน AI อย่างเข้าใจ • เป็นระบบ • มีประสิทธิภาพ

ทวีศักดิ์ สมานชื่น

CBTU-MU

Prompt Engineering

ใช้งาน AI อย่างเข้าใจ • เป็นระบบ • มีประสิทธิภาพ

คู่มือ Prompt Engineering
ใช้งาน AI อย่างเข้าใจ • เป็นระบบ • มีประสิทธิภาพ

ผู้เขียน ผู้ช่วยศาสตราจารย์ ดร.ทวีศักดิ์ สมานชื่น
สังกัด กลุ่มสาขาวิชาเทคโนโลยีการจัดการระบบสารสนเทศ คณะวิศวกรรมศาสตร์
มหาวิทยาลัยมหิดล และ หอสมุดและคลังความรู้มหาวิทยาลัยมหิดล

ข้อมูลทางบรรณานุกรมของหอสมุดแห่งชาติ
ทวีศักดิ์ สมานชื่น.
คู่มือ Prompt Engineering: ใช้งาน AI อย่างเข้าใจ เป็นระบบ มีประสิทธิภาพ.-- กรุงเทพฯ: จรัลสนิทวงศ์การพิมพ์ จำกัด, 2569.
402 หน้า.-- (AI Series: Pocket Guide).

1. ปัญญาประดิษฐ์. I. ชื่อเรื่อง.

006.3
ISBN 978-616-631-394-9

จัดทำโดย ผู้ช่วยศาสตราจารย์ ดร.ทวีศักดิ์ สมานชื่น

ติดต่อผู้เขียน
อีเมล: taweesak.sam@mahidol.ac.th

พิมพ์ครั้งที่ 1: มีนาคม 2026
พิมพ์ที่ บริษัท จรัลสนิทวงศ์การพิมพ์ จำกัด
233 ซอย เพชรเกษม 102/2
แขวงบางแคเหนือ เขตบางแค กรุงเทพมหานคร 10160

กิตติกรรมประกาศ

หนังสือเล่มนี้สำเร็จลงได้ด้วยความช่วยเหลือและการสนับสนุนจากหลายฝ่าย ผมขอใช้โอกาสนี้ในการแสดงความขอบคุณอย่างจริงใจต่อทุกคนที่มีส่วนร่วมในการทำให้หนังสือเล่มนี้เป็นจริง ผมขอขอบคุณ

หน่วยงานต้นสังกัด

ขอขอบคุณ **กลุ่มสาขาวิชาเทคโนโลยีการจัดการระบบสารสนเทศ คณะวิศวกรรมศาสตร์ มหาวิทยาลัยมหิดล** และ **หอสมุดและคลังความรู้มหาวิทยาลัยมหิดล** ที่ให้โอกาสและสนับสนุนการจัดทำหนังสือเล่มนี้ เป็นส่วนหนึ่งของการพัฒนาองค์ความรู้และการเผยแพร่ความรู้สู่สาธารณะ

ผู้ทรงคุณวุฒิและผู้ตรวจทาน

ขอขอบคุณผู้ทรงคุณวุฒิและเพื่อนร่วมงานที่กรุณาสละเวลาอ่าน ให้คำแนะนำ และข้อเสนอแนะอันมีค่า ช่วยให้หนังสือเล่มนี้มีความสมบูรณ์และแม่นยำยิ่งขึ้น

ครอบครัวและเพื่อนฝูง

ขอบคุณครอบครัวที่ให้กำลังใจและเข้าใจในช่วงเวลาที่ต้องอุทิศเวลาให้กับการเขียนหนังสือเล่มนี้ ขอบคุณเพื่อน ๆ ที่คอยให้คำปรึกษา แบ่งปันประสบการณ์ และเป็นแรงผลักดันให้ก้าวต่อไป

ผู้อ่านและสมาชิก CBTU-MU

ขอบคุณผู้อ่านทุกท่านที่ให้ความสนใจติดตามผลงานของเรามาในทุกช่องทาง ทั้งหนังสือ เว็บไซต์ และโซเชียลมีเดีย และ สมาชิก CBTU-MU ทุกคนที่ร่วมกันแลกเปลี่ยนความรู้ ประสบการณ์ และสร้างพื้นที่การเรียนรู้ร่วมกัน

ทีมงาน CBTU-MU และผู้จัดพิมพ์

ขอบคุณทีมงาน CBTU-MU และผู้จัดพิมพ์ที่ให้การสนับสนุนในกระบวนการผลิตและจัดจำหน่ายหนังสือเล่มนี้ ทำให้ความรู้สามารถเข้าถึงผู้อ่านได้อย่างกว้างขวาง

ความพยายามและความทุ่มเทในการจัดทำหนังสือเล่มนี้เป็นผลมาจากการสนับสนุนและความร่วมมือจากทุกฝ่ายที่กล่าวมา ผมหวังว่าหนังสือเล่มนี้จะเป็นประโยชน์และแรงบันดาลใจให้กับผู้อ่านทุกคนในการเข้าใจและใช้งาน AI อย่างมีประสิทธิภาพในชีวิตประจำวันและการทำงานครับ หากมีข้อผิดพลาดหรือข้อบกพร่องประการใด ผมขอน้อมรับไว้แต่เพียงผู้เดียว และยินดีรับฟังข้อเสนอแนะเพื่อนำไปปรับปรุงในโอกาสต่อไป

ทวีศักดิ์ สมานชื่น
กุมภาพันธ์ 2026

คำนำ

ในยุคที่ปัญญาประดิษฐ์ไม่ใช่แค่เทคโนโลยีที่น่าตื่นเต้น แต่กำลังกลายเป็นเครื่องมือที่เปลี่ยนแปลงวิธีการทำงาน การเรียนรู้ และการแก้ปัญหาของเราอย่างถาวร จนกระทั่งจะกลายเป็นปัจจัยที่ขาดไปไม่ได้ในอนาคต เช่นเดียวกับ โทรศัพท์มือถือ หรือ อินเทอร์เน็ต การเข้าใจและใช้งาน AI อย่างมีประสิทธิภาพจึงไม่ใช่ทักษะเสริม แต่เป็นความจำเป็นพื้นฐานสำหรับผู้ที่ต้องการก้าวทันโลกที่เปลี่ยนแปลงอย่างรวดเร็ว

หนังสือเล่มนี้เกิดขึ้นจากการสังเกตช่องว่างสำคัญในแหล่งความรู้ภาษาไทยเกี่ยวกับ Large Language Model (LLM) และ Prompt Engineering ในขณะที่เนื้อหาภาษาอังกฤษมีอยู่อย่างมากมาย แต่ผู้ใช้งานจำนวนมากยังขาดแหล่งความรู้ที่อธิบายแนวคิดเชิงเทคนิคอย่างเป็นระบบและเข้าถึงได้ง่าย ผมจึงตั้งใจเขียนหนังสือเล่มนี้เพื่อเป็นสะพานเชื่อมระหว่างความรู้ทางทฤษฎีกับการประยุกต์ใช้จริง โดยไม่ตัดความลึกของเนื้อหา แต่นำเสนอในลักษณะที่ผู้อ่านสามารถเข้าใจและนำไปใช้ได้ทันที

เนื่องจากการพัฒนา AI เป็นเรื่องที่เกิดขึ้นอย่างรวดเร็ว เนื้อหาในหนังสือเล่มนี้จึงถูกอัปเดตให้ทันสมัยที่สุดเท่าที่จะเป็นไปได้ และไม่ได้ผ่านกระบวนการตีพิมพ์แบบดั้งเดิมที่อาจทำให้เนื้อหาล้าสมัยก่อนที่จะถึงมือผู้อ่าน ผมจึงเลือกที่จะเผยแพร่ในรูปแบบ Self-Published เพื่อให้สามารถอัปเดตเนื้อหาได้อย่างต่อเนื่องตามการเปลี่ยนแปลงของเทคโนโลยีและแนวปฏิบัติในวงการ AI ผมหวังว่าหนังสือเล่มนี้จะเป็นแหล่งความรู้ที่มีประโยชน์และเป็นแรงบันดาลใจให้กับผู้ที่ต้องการเข้าใจและใช้งาน AI อย่างมีประสิทธิภาพในชีวิตประจำวันและการทำงานครับ

หนังสือเล่มนี้เหมาะกับใคร

หนังสือเล่มนี้ออกแบบมาสำหรับผู้ที่ต้องการเข้าใจ AI อย่างลึกซึ้งและนำไปใช้งานได้จริง ไม่ว่าผู้อ่านจะเป็น

- **ผู้ใช้งาน AI ทั่วไป** ที่ต้องการยกระดับทักษะจากการใช้งานเบื้องต้นสู่การใช้งานอย่างมืออาชีพ
- **นักวิชาการและนักวิจัย** ที่สนใจศึกษาพื้นฐานทางเทคนิคของ AI และแนวทางการวิจัยในปัจจุบัน
- **นักพัฒนาและ Data Scientists** ที่ต้องการเข้าใจโครงสร้างภายในของ LLM ข้อจำกัด และเทคนิคการใช้งานขั้นสูง
- **ผู้บริหารและผู้ตัดสินใจ** ที่ต้องการประเมินศักยภาพและข้อจำกัดของ AI เพื่อนำมาใช้ในองค์กร

ไม่ว่าผู้อ่านจะมีพื้นฐานทางเทคนิคมากน้อยเพียงใด หนังสือเล่มนี้จะช่วยให้ผู้อ่านเข้าใจ AI ในระดับที่ลึกกว่าการใช้งานผิวเผิน และสามารถนำความรู้ไปประยุกต์ใช้ในบริบทของผู้อ่านเองได้

สิ่งที่คุณจะได้เรียนรู้

หนังสือเล่มนี้แบ่งเนื้อหาออกเป็น 7 บทและ 10 ภาคผนวกที่เชื่อมโยงกันอย่างเป็นระบบ

บทที่ 1: พื้นฐานปัญญาประดิษฐ์ ปูพื้นฐานความเข้าใจตั้งแต่แนวคิดพื้นฐานของ AI, Machine Learning, Deep Learning ไปจนถึงสถาปัตยกรรม Transformer ที่เป็นหัวใจของ LLM ทุกตัวในปัจจุบัน ผู้อ่านจะเข้าใจว่า AI "คิด" และ "เรียนรู้" อย่างไร และทำไมสถาปัตยกรรม Transformer จึงปฏิวัติวงการ AI

บทที่ 2: ข้อจำกัดและแนวทางการปรับปรุง เจาะลึกปัญหาที่สำคัญของ LLM เช่น การสร้างข้อมูลที่ไม่เป็นความจริง (Hallucination), Lost in the Middle (ปัญหาการจัดการบริบทขนาดใหญ่) และข้อจำกัดด้าน Memory ผู้อ่านจะได้เรียนรู้เทคนิคการแก้ไขเช่น RAG (Retrieval-Augmented Generation) และ Fine-tuning พร้อมแนวทางการสร้างระบบ AI ที่เชื่อถือได้

บทที่ 3: Prompt Engineering นำเสนอเทคนิคการเขียน Prompt อย่างเป็นระบบ ตั้งแต่พื้นฐานอย่าง Zero-shot และ Few-shot ไปจนถึงเทคนิคขั้นสูงอย่าง Chain-of-Thought, Tree of Thoughts และ ReAct Framework พร้อมกรอบการทำงานระดับองค์กรอย่าง CO-STAR ที่ช่วยสร้างความสม่ำเสมอของผลลัพธ์

บทที่ 4: เทคนิคการใช้งาน ChatGPT แนะนำการใช้งาน ChatGPT อย่างมืออาชีพ ครอบคลุมฟีเจอร์ต่าง ๆ เช่น Deep Research, Canvas, Projects, Schedules และ Memory พร้อมแนวทางการจัดการข้อมูลอย่างปลอดภัย และเทคนิคการเลือกโหมดที่เหมาะกับแต่ละประเภทงาน

บทที่ 5: เทคนิคการใช้งาน Gemini AI เรียนรู้ความสามารถพิเศษของ Gemini จาก Google โดยเฉพาะการทำงานร่วมกับ Google Workspace ครอบคลุม Deep Research, Canvas, Gems และ Connected Apps พร้อม Prompt Framework 6 ช่องที่ใช้ได้จริง

บทที่ 6: จริยธรรมปัญญาประดิษฐ์ นำเสนอประเด็นสำคัญด้าน AI Ethics ครอบคลุม Fairness และการลด Bias, Transparency และ Explainability, Privacy และ Data Protection, Accountability, Safety และ Security รวมถึงกรอบกฎหมายและแนวทางการใช้ AI อย่างมีจริยธรรมในทางปฏิบัติ

บทที่ 7: สรุปและแนวทางการศึกษาต่อ สังเคราะห์สาระสำคัญจากทุกบท เชื่อมโยงความรู้ทั้งหมดเข้าด้วยกัน และนำเสนอแนวทางการศึกษาต่อในอนาคต ครอบคลุมทั้งด้านเทคนิค การประยุกต์ใช้ และการพัฒนาตนเองในยุค AI

ภาคผนวก A–J รวบรวมตัวอย่าง Prompt ที่ใช้งานได้จริงสำหรับ 10 กลุ่มอาชีพหลัก ทุกตัวอย่างใช้ CO-STAR Framework เพื่อให้ได้ผลลัพธ์ที่มีคุณภาพ

- **ภาคผนวก A: Content Creator** การสร้าง outline บทความ การกำหนดหัวข้อวิจัย การเขียน scripts การวางแผนเนื้อหา
- **ภาคผนวก B: ข้าราชการ** การเขียนหนังสือราชการ การจัดทำโครงการขอทุน การจัดทำแผนปฏิบัติราชการ การสื่อสารและประชาสัมพันธ์หน่วยงาน
- **ภาคผนวก C: ครูและอาจารย์** การสร้างแผนการสอน การออกข้อสอบ การให้ feedback การทำวิจัย
- **ภาคผนวก D: เจ้าหน้าที่ HR** การสรรหาบุคลากร การสัมภาษณ์งาน การพัฒนาบุคลากร การสื่อสารนโยบาย การวิเคราะห์ข้อมูล HR

- **ภาคผนวก E: นักการตลาด** การวางแผน content calendar การเขียน copywriting การวิเคราะห์ตลาดและคู่แข่ง
- **ภาคผนวก F: นักวิเคราะห์ข้อมูล** การเขียน SQL การสร้าง Dashboard การวิเคราะห์แนวโน้ม A/B Testing การสร้างรายงานอัตโนมัติ
- **ภาคผนวก G: นักวิจัย** การทบทวนวรรณกรรม การออกแบบระเบียบวิธีวิจัย การเขียนบทความวิชาการ การวิเคราะห์ข้อมูล การเขียน Grant Proposal
- **ภาคผนวก H: บรรณารักษ์** การสืบค้น การจัดการทรัพยากร การให้บริการอ้างอิง การสอนสื่อสารสารนิเทศ การพัฒนาคอลเลกชัน
- **ภาคผนวก I: ผู้บริหารหรือผู้จัดการ** สรุปการประชุม เขียนรายงานผู้บริหาร วางแผนงบประมาณประจำปี การสื่อสารการเปลี่ยนแปลงภายในองค์กร
- **ภาคผนวก J: พนักงานออฟฟิศ** การเขียนอีเมล การสรุปเอกสาร การจัดทำรายงาน การสร้าง SOP

แต่ละตัวอย่างมีทั้ง Prompt แบบเต็ม พร้อม Tips และแนวทางการปรับแต่งให้เหมาะกับบริบทของผู้อ่าน

จุดเด่นของหนังสือเล่มนี้

สิ่งที่ทำให้หนังสือเล่มนี้แตกต่างจากแหล่งความรู้อื่น ๆ คือ

ความสมดุลระหว่างทฤษฎีและปฏิบัติ ไม่ใส่รายละเอียดทางคณิตศาสตร์เกินจำเป็น แต่ไม่ตัดเนื้อหาทางเทคนิคที่สำคัญ อธิบายแนวคิดซับซ้อนด้วยภาษาที่เข้าใจได้ พร้อมตัวอย่างประกอบที่ชัดเจน

เนื้อหาที่ทันสมัย อ้างอิงงานวิจัยและแนวปฏิบัติล่าสุดในปี 2025–2026 รวมถึงฟีเจอร์ใหม่ ๆ ของ ChatGPT และ Gemini ที่พึ่งเปิดตัว

ตัวอย่างและ Prompt ที่ใช้งานได้จริง มี Prompt ตัวอย่างมากกว่า 100 ชุด สำหรับ 10 กลุ่มอาชีพ ที่ผู้อ่านสามารถนำไปใช้ได้ทันที ครอบคลุมงานหลากหลายประเภท ตั้งแต่การเขียน การวิจัย ไปจนถึงการวิเคราะห์ข้อมูล

แนวทางความปลอดภัย ให้ ความ สำคัญ กับ การ จัดการ ข้อมูล อย่าง ปลอดภัย การป้องกันข้อมูลอ่อนไหว และการใช้งาน AI อย่างมี จริยธรรม

กรอบความคิดที่นำไปใช้ได้ นำเสนอ Framework และ Checklist ที่ช่วย ในการตัดสินใจและประเมินผลลัพธ์ ไม่ใช่แค่ให้ความรู้ แต่ให้เครื่อง มือในการคิดและทำงาน

วิธีการอ่านหนังสือเล่มนี้

หนังสือเล่มนี้ออกแบบให้อ่านได้หลายรูปแบบตามความต้องการของผู้อ่าน

- **อ่านตามลำดับ** สำหรับผู้ที่ต้องการสร้างความเข้าใจอย่างเป็นระบบ ตั้งแต่พื้นฐานไปจนถึงขั้นสูง
- **เลือกอ่านเฉพาะบทที่สนใจ** แต่ละบทถูกออกแบบมาให้มีความเป็น อิสระ หากผู้อ่านต้องการเรียนรู้เฉพาะเทคนิคการใช้งาน ChatGPT หรือ Gemini สามารถข้ามไปบทที่ 4–5 ได้เลย หรือหากสนใจเฉพาะ AI Ethics สามารถข้ามไปบทที่ 6 ได้โดยตรง
- **ใช้เป็นคู่มืออ้างอิง** Prompt ตัวอย่าง Framework และ Checklist ต่าง ๆ สามารถใช้เป็นแหล่งอ้างอิงในการทำงานประจำวัน

แนะนำให้ทดลองใช้ Prompt และเทคนิคต่าง ๆ ที่นำเสนอในหนังสือ ด้วยตัวเองขณะอ่าน การเรียนรู้ AI ไม่ใช่การท่องจำ แต่เป็นการทดลอง ปรับแต่ง และเข้าใจผ่านประสบการณ์จริง ผลลัพธ์ที่ได้จากการใช้ Prompt และเทคนิคต่าง ๆ อาจแตกต่างกันไปขึ้นอยู่กับหลายปัจจัย เช่น เวอร์ชัน ของโมเดล AI การตั้งค่า Parameter และบริบทการใช้งานจริง ดังนั้นควร ทดสอบและปรับแต่ง Prompt ให้เหมาะสมกับความต้องการเฉพาะของ แต่ละบุคคลหรือองค์กร

สำหรับ Prompt ในภาคผนวก ผู้อ่านสามารถคัดลอกและนำไปใช้ได้ ทันที แต่ควรปรับแต่งให้เหมาะกับบริบทของตนเองเพื่อให้ได้ผลลัพธ์ที่ดี ที่สุด ทั้งนี้ผู้อ่านสามารถขอรับส่วนของภาคผนวกได้ทางเว็บไซต์ หรือทาง อีเมลของผู้เขียน เพื่อความสะดวกในการคัดลอกและใช้งาน Prompt โดย ไม่ต้องพิมพ์เอง และจะได้มีการอัปเดต Prompt ใหม่ ๆ เพิ่มเติมในอนาคต ด้วยครับ

การเปิดเผยข้อมูล

ผู้เขียนไม่มีความสัมพันธ์ทางการเงินหรือผลประโยชน์ทับซ้อนกับบริษัทหรือองค์กรใด ๆ ที่เกี่ยวข้องกับ AI หรือเทคโนโลยีที่กล่าวถึงในหนังสือเล่มนี้ เนื้อหาในหนังสือถูกเขียนขึ้นโดยอิงจากงานวิจัยและแนวปฏิบัติที่เป็นสาธารณะ และมีจุดมุ่งหมายเพื่อให้ความรู้และเครื่องมือในการใช้งาน AI อย่างมีประสิทธิภาพและมีจริยธรรมแก่ผู้อ่านทุกคน

การเขียนหนังสือเล่มนี้ มีการใช้ AI ในบางส่วนของกระบวนการ เช่น การช่วยในการรวบรวมข้อมูล การสร้างตัวอย่าง Prompt และการตรวจสอบความถูกต้องของเนื้อหา แต่เนื้อหาทั้งหมดได้รับการตรวจสอบและปรับแต่งโดยผู้เขียนเพื่อให้เหมาะสมกับบริบทและความต้องการของผู้อ่าน และเพื่อให้แน่ใจว่าเนื้อหามีความถูกต้องและเป็นประโยชน์สูงสุดสำหรับผู้อ่านครับ

ข้อควรตระหนัก

AI เป็นเครื่องมือที่ทรงพลัง แต่ไม่ใช่คำตอบสำหรับทุกปัญหา การใช้งาน AI อย่างมีประสิทธิภาพต้องอาศัยความเข้าใจทั้งจุดแข็งและข้อจำกัด การตรวจสอบผลลัพธ์อย่างรอบคอบ และการใช้วิจารณญาณของมนุษย์ในการตัดสินใจขั้นสุดท้าย

หนังสือเล่มนี้จะช่วยให้ผู้อ่านเข้าใจว่า AI ทำงานอย่างไร ใช้งานอย่างไร และควรระวังอะไร แต่วิธีที่ผู้อ่านจะนำความรู้เหล่านี้ไปประยุกต์ใช้ในบริบทของผู้อ่านเอง นั่นคือสิ่งที่จะสร้างความแตกต่างที่แท้จริง

ผมหวังว่าหนังสือเล่มนี้จะเป็นจุดเริ่มต้นที่ดีในการเดินทางของผู้อ่านสู่โลกแห่ง AI และการใช้งาน LLM อย่างมืออาชีพ ขอให้การอ่านเป็นประโยชน์และสนุกกับการเรียนรู้ครับ

ทวีศักดิ์ สมานชื่น
กุมภาพันธ์ 2026

สารบัญ

บทที่ 1

พื้นฐานปัญญาประดิษฐ์

โลกของเทคโนโลยีในศตวรรษที่ 21 ได้ถูกเปลี่ยนไปอย่างสิ้นเชิงด้วยความก้าวหน้าของปัญญาประดิษฐ์ (Artificial Intelligence - AI) การทำความเข้าใจเทคโนโลยีนี้ไม่ได้เป็นเพียงเรื่องของกระแสความนิยม แต่เป็นความจำเป็นที่สำคัญของผู้ใช้งาน AI ทุกคน เพื่อที่จะจำแนกความแตกต่างระหว่างความสามารถที่แท้จริงกับความคาดหวังที่เกินจริง การปูพื้นฐานความเข้าใจในสถาปัตยกรรมระดับลึก ตั้งแต่วิวัฒนาการของโครงข่ายประสาทเทียมยุคแรกเริ่มจนถึงสถาปัตยกรรม Transformer ที่ขับเคลื่อน Large Language Model (LLM) ในปัจจุบัน จึงเป็นจุดเริ่มต้นที่สำคัญ

คำว่า ปัญญาประดิษฐ์ หรือ AI เป็นคำที่ทุกคนคุ้นเคยกันอยู่แล้ว ไม่ว่าจะเป็นรูปแบบทางวิชาการ หรือรูปแบบความบันเทิงต่าง ๆ ที่ได้สร้างภาพจินตนาการของ AI ที่มีความสามารถเหนือมนุษย์ในหลาย ๆ ด้าน แต่ในความเป็นจริงแล้ว AI มีความซับซ้อนและหลากหลายมากกว่าที่เราคิด ความรู้ทางด้าน AI โดยเฉพาะในส่วนของการเรียนรู้ของเครื่อง (Machine Learning-ML) มีการเรียนการสอนกันกว้างกวางในระดับมหาวิทยาลัย แต่ก็เป็นเพียงเทคนิคที่ใช้เฉพาะด้านสำหรับนักพัฒนาและนักวิจัยเท่านั้น ยังไม่สามารถเข้าถึงผู้ใช้งานทั่วไปที่ต้องการใช้ AI ในชีวิตประจำวันได้อย่างมีประสิทธิภาพ ที่สำคัญในช่วงเวลาที่ผ่านมานั้น AI ยังมีข้อจำกัดในแง่ของความเข้าใจที่ลึกซึ้งและการงานที่มีประสิทธิภาพ โดยเฉพาะอย่างยิ่งในบริบทของภาษาและการสื่อสาร ซึ่งเป็นหนึ่งในความท้าทายที่สำคัญที่สุดของ AI จนกระทั่งการมาถึงของ ChatGPT ในปี 2022 ที่ทำให้ AI Chatbot กลายเป็นที่รู้จักและใช้งานกันอย่างแพร่หลายมากขึ้น และมีการ

แข่งขันกันอย่างดุเดือดในตลาด LLM ในช่วงปี 2023–2025 ที่ทำให้เกิดการพัฒนาอย่างรวดเร็วและการเปิดตัวฟีเจอร์ใหม่ ๆ อย่างต่อเนื่อง การทำความเข้าใจพื้นฐานของ AI จึงเป็นสิ่งสำคัญที่จะช่วยให้ผู้ใช้งานสามารถใช้ประโยชน์จากเทคโนโลยีนี้ได้อย่างเต็มที่ และหลีกเลี่ยงการตกเป็นเหยื่อของความเข้าใจผิดหรือการใช้งานที่ไม่เหมาะสม ดังนั้นในหนังสือเล่มนี้ เราจะเริ่มต้นด้วยการทำความเข้าใจคำศัพท์พื้นฐานที่เกี่ยวข้องกับ AI และ ML รวมถึงการเรียนรู้วิวัฒนาการของแบบจำลองทางภาษา ตั้งแต่ยุคของ N-Gram ไปจนถึงยุคของ RNN และ LSTM ก่อนที่จะเข้าสู่สถาปัตยกรรม Transformer ที่เป็นรากฐานของ LLM ในปัจจุบัน

1.1 คำศัพท์ควรรู้เทคโนโลยีปัญญาประดิษฐ์

ในการศึกษาโครงสร้างของ AI จำเป็นต้องเริ่มต้นจากการจำแนกประเภท (Taxonomy) ที่ชัดเจน เพื่อลดความสับสนในการใช้งานคำศัพท์ที่มักถูกใช้สลับกันไปมาในบริบททั่วไป [1], [2] ความสัมพันธ์ของเทคโนโลยีเหล่านี้มีลักษณะเป็นลำดับชั้นดังนี้

1. **ปัญญาประดิษฐ์ (Artificial Intelligence – AI)**
   AI เป็นร่มใหญ่ที่ครอบคลุมศาสตร์ในการสร้างเครื่องจักรที่มีความสามารถในการเลียนแบบสติปัญญาของมนุษย์ นิยามของ AI กว้างขวางและรวมถึงระบบที่ใช้กฎ (Rule-based systems) ซึ่งถูกโปรแกรมไว้อย่างชัดเจน เช่น ระบบผู้เชี่ยวชาญ (Expert Systems) ในยุค 1980 ที่ทำงานตามตรรกะ "If–Then" โดยไม่มีการเรียนรู้จากข้อมูลใหม่ เป้าหมายสูงสุดของ AI คือการสร้างระบบที่สามารถรับรู้ ให้เหตุผล เรียนรู้ และตัดสินใจได้เสมือนมนุษย์

2. **การเรียนรู้ของเครื่อง (Machine Learning – ML)**
   ML เป็นเทคนิคกลุ่มหนึ่งของ AI ที่เปลี่ยนแนวคิดจากการเขียนโปรแกรมที่มีกระบวนการทำงานที่ชัดเจน ไปสู่การสร้างอัลกอริทึมที่สามารถ "เรียนรู้" รูปแบบ (Patterns) และกฎเกณฑ์จากข้อมูล [1] โดยทั่วไป ML แบ่งออกเป็น 3 ประเภทหลักตามลักษณะการฝึกสอน

   - *Supervised Learning* การเรียนรู้แบบมีผู้สอน โดยใช้ข้อมูลที่มีป้ายกำกับ (Labeled Data) เช่น การทำนายราคาบ้านจากขนาดและทำเล
   - *Unsupervised Learning* การเรียนรู้แบบไม่มีผู้สอน โดยให้โมเดลหาโครงสร้างที่ซ่อนอยู่ (Hidden Structures) ในข้อมูล

เช่น การจัดกลุ่มลูกค้า (Clustering) [3], [4]

- *Reinforcement Learning* การเรียนรู้ผ่านการลองผิดลองถูกและการได้รับรางวัลหรือบทลงโทษ ซึ่งเป็นพื้นฐานสำคัญในการฝึกหุ่นยนต์หรือ AI เล่นเกม

3. **การเรียนรู้เชิงลึก (Deep Learning – DL)**
   DL เป็นวิวัฒนาการขั้นสูงของ ML ที่ได้รับแรงบันดาลใจจากโครงสร้างทางชีวภาพของสมองมนุษย์ หรือโครงข่ายประสาทเทียม (Artificial Neural Networks – ANNs) ซึ่งจัดเป็นประเภทหนึ่งของ ML ความแตกต่างที่สำคัญระหว่าง ANNs ทั่วไปกับ DL คือ "ความลึก" (Depth) ของเลเยอร์ หรือชั้นของการประมวลผล [3] ในขณะที่ ANNs แบบดั้งเดิมต้องการมนุษย์ในการสกัดคุณลักษณะ (Feature Extraction) จากข้อมูลดิบ DL สามารถเรียนรู้ที่จะสกัดคุณลักษณะเหล่านั้นได้ด้วยตนเองผ่านเลเยอร์ที่ซ้อนทับกันเป็นจำนวนมาก ทำให้มีประสิทธิภาพสูงเป็นพิเศษในการจัดการกับข้อมูลที่ไม่มีโครงสร้าง (Unstructured Data) เช่น รูปภาพ เสียง และข้อความ

4. **โมเดลภาษาขนาดใหญ่ (Large Language Model – LLM)**
   LLM คือจุดสูงสุดของเทคโนโลยี Deep Learning ในปัจจุบัน นี้ที่ถูกออกแบบมาเพื่อประมวลผลและสร้างข้อความ โดยอาศัยสถาปัตยกรรมหลักที่เรียกว่า Transformer ซึ่งจะกล่าวถึงในรายละเอียดต่อไป โมเดลเหล่านี้ได้รับการฝึกฝนด้วยชุดข้อมูลขนาดใหญ่มาก ระดับ Petabyte (10^{15} ไบต์) ซึ่งครอบคลุมความรู้เกือบทั้งหมดบนอินเทอร์เน็ต [5] ความ "ใหญ่" ของ LLM วัดจากจำนวนพารามิเตอร์ (Parameters) ซึ่งเปรียบเสมือนจุดเชื่อมต่อประสาทในสมอง โดยโมเดลระดับแนวหน้าอย่าง GPT-5 หรือ Gemini มีพารามิเตอร์ในระดับล้านล้าน (Trillions) ทำให้เกิดความสามารถอุบัติใหม่ (Emergent Capabilities) ที่ไม่ได้ถูกสอนโดยตรง เช่น การเขียนโค้ด การแก้โจทย์คณิตศาสตร์ หรือการใช้เหตุผลเชิงตรรกะ [6]

5. **Prompt Engineering**
   เป็นเทคนิคที่ใช้ในการออกแบบและปรับแต่งคำสั่ง (Prompt) เพื่อให้ LLM ตอบสนองได้อย่างมีประสิทธิภาพและแม่นยำมากขึ้น โดยการเลือกใช้คำที่เหมาะสม การจัดรูปแบบคำถาม หรือการให้ตัวอย่างที่ชัดเจน สามารถช่วยให้โมเดลเข้าใจเจตนาของผู้ใช้ได้ดีขึ้น และลดความคลุมเครือในการตอบกลับ ทำให้ได้ผลลัพธ์ตรงตามความต้องการของผู้ใช้

6. **Pre-train**
   เป็นกระบวนการฝึกสอนโมเดลด้วยชุดข้อมูลขนาดใหญ่ที่มีความหลากหลาย เพื่อให้โมเดลเรียนรู้รูปแบบและความสัมพันธ์ในข้อมูลก่อนที่จะนำไปใช้งานในงานเฉพาะทาง เช่น การฝึกสอน GPT-5 ด้วยข้อมูลจากอินเทอร์เน็ตที่ครอบคลุมหัวข้อต่าง ๆ เพื่อให้โมเดลมีความรู้พื้นฐานกว้าง ก่อนที่จะนำไปปรับแต่งสำหรับงานเฉพาะทาง

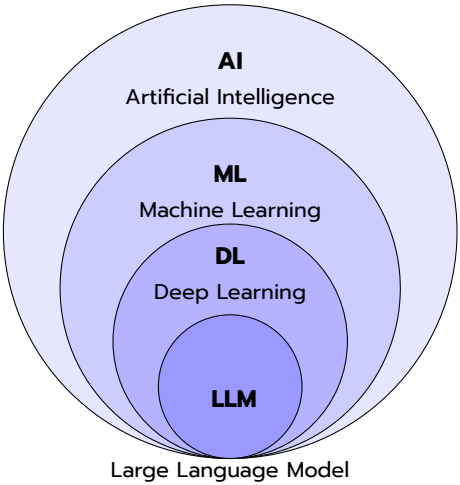
7. **Fine-tuning**
   เป็นกระบวนการปรับแต่งโมเดลที่ได้รับการฝึกฝนมาแล้ว (Pre-trained Model) ด้วยชุดข้อมูลเฉพาะทาง เพื่อให้โมเดลมีความเชี่ยวชาญในงานหรือโดเมนที่ต้องการ เช่น การปรับแต่ง GPT-5 ให้เชี่ยวชาญในการวิเคราะห์ทางการแพทย์โดยใช้ข้อมูลจากงานวิจัยและเคสทางคลินิก

8. **Tokenization**
   เป็นกระบวนการแปลงหรือแบ่งข้อความดิบให้เป็นหน่วยย่อยที่เรียกว่า Token ซึ่งอาจเป็นคำหรือส่วนของคำ (Subword) เพื่อให้โมเดลสามารถประมวลผลได้อย่างมีประสิทธิภาพ เช่น การแปลงประโยค “ฉันรักการเรียนรู้” เป็น Token เช่น “ฉัน”, “รัก”, “การ”, “เรียน รู้” หรือในบางกรณีอาจถูกแยกเป็น Subword เช่น “เ-รียน-รู้” เพื่อจัดการกับคำที่ไม่เคยเห็นมาก่อน

9. **Embeddings**
   เป็นเทคนิคที่ใช้ในการแปลง Token ให้เป็นเวกเตอร์ตัวเลขในมิติสูง (High-dimensional space) โดยคำที่มีความหมายใกล้เคียงกันจะมีค่าเวกเตอร์ที่ใกล้เคียงกัน เช่น คำว่า “แมว” และ “สุนัข” อาจมีเวกเตอร์ที่อยู่ใกล้กันในพื้นที่เวกเตอร์ เนื่องจากทั้งสองคำนี้เป็นสัตว์เลี้ยงและมีลักษณะคล้ายคลึงกัน

10. **Attention**
    เป็นกลไกที่ช่วยให้โมเดลสามารถโฟกัสไปที่ส่วนที่สำคัญของข้อมูลขณะประมวลผล เช่น ในประโยค “แมวอยู่บนโต๊ะ” โมเดลอาจให้ความสนใจมากขึ้นกับคำว่า “แมว” และ “โต๊ะ” เพื่อเข้าใจความสัมพันธ์ระหว่างสิ่งเหล่านี้ได้ดีขึ้น

11. **Transformer**
    เป็นสถาปัตยกรรมที่ถูกนำมาใช้ใน LLM ซึ่งมีความสามารถในการประมวลผลข้อมูลแบบขนาน (Parallel) และจัดการกับบริบทระยะยาวได้อย่างมีประสิทธิภาพ โดยใช้กลไก Attention เพื่อสร้างความสัมพันธ์ระหว่างคำในประโยคได้อย่างแม่นยำ

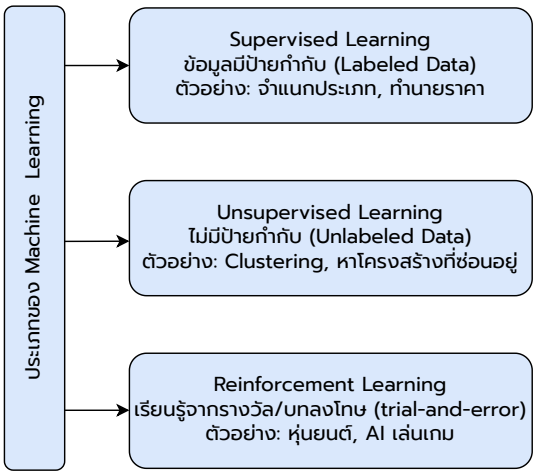


รูปที่ 1.1: ความสัมพันธ์ของคำศัพท์ AI, ML, DL และ LLM

12. **Emergent Capabilities**
เป็นความสามารถที่เกิดขึ้นโดยไม่ได้คาดคิดในโมเดลขนาดใหญ่ เมื่อจำนวนพารามิเตอร์เพิ่มขึ้นถึงระดับหนึ่ง โมเดลจะเริ่มแสดงพฤติกรรมหรือความสามารถใหม่ ๆ ที่ไม่ได้ถูกสอนโดยตรง เช่น การแก้ปัญหาทางคณิตศาสตร์ การเขียนโค้ด หรือการใช้เหตุผลเชิงตรรกะ ซึ่งเป็นสิ่งที่ไม่เคยเห็นในโมเดลขนาดเล็กมาก่อน

13. **Context Window**
เป็นจำนวนคำที่โมเดลสามารถพิจารณาได้ในครั้งเดียว เช่น LLM อาจมี context window ขนาด 4,096 หรือ 8,192 โทเคน ซึ่งหมายความว่าโมเดลสามารถเข้าใจและสร้างข้อความที่มีความยาวได้มากขึ้นโดยไม่สูญเสียบริบทสำคัญ

รูปที่ 1.1 แสดงความสัมพันธ์ของคำศัพท์ทั้ง 4 คำ คือคำว่า AI, ML, DL และ LLM ดังที่ได้กล่าวไป จะเห็นความสัมพันธ์ในลักษณะที่เรียกว่าเป็น subset ลงไปเรื่อย ๆ โดยจะมีคำว่า AI เป็นวงกลมที่ใหญ่ที่สุดอยู่ภายนอก และ LLM เป็นวงกลมที่เล็กที่สุดอยู่ภายใน ในขณะที่รูปที่ 1.2 แสดงการแบ่งประเภทของ ML หลัก ๆ ที่มีใช้ในปัจจุบัน



รูปที่ 1.2: ประเภทหลักของ Machine Learning ประกอบด้วย Supervised, Unsupervised และ Reinforcement Learning

1.2 วิวัฒนาการของแบบจำลองทางภาษา

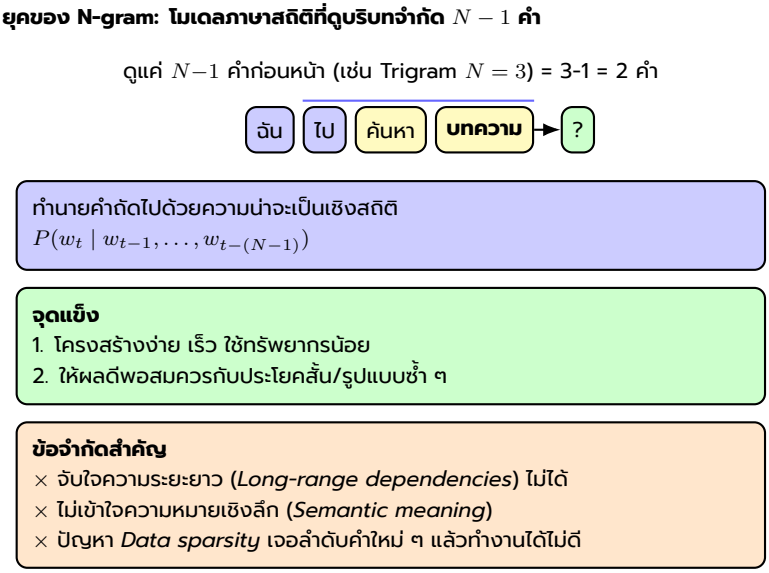
ก่อนที่ LLM จะเข้ามามีบทบาท งานประมวลผลภาษาธรรมชาติ (NLP) ซึ่งเป็นสาขาหนึ่งของปัญญาประดิษฐ์ได้พัฒนามาอย่างต่อเนื่องยาวนาน โดยมีเป้าหมายสำคัญคือทำให้คอมพิวเตอร์ "เข้าใจ" และ "สื่อสาร" ด้วยภาษามนุษย์ได้ใกล้เคียงความเป็นจริงมากขึ้น ไม่ว่าจะเป็นการทำความเข้าใจความหมายของประโยค การตอบคำถาม การสรุปข้อความ การแปลภาษา หรือการวิเคราะห์ความรู้สึกจากข้อความ อย่างไรก็ตาม ความท้าทายที่สำคัญที่เป็นเหมือนกำแพงของงานด้านภาษาตลอดมาคือ "การเข้าใจบริบทของภาษา" เพราะภาษาไม่ได้มีความหมายจากคำเดี่ยว ๆ แต่เกิดจากความสัมพันธ์ของคำกับคำอื่น ๆ สถานการณ์ ผู้พูด และความรู้พื้นหลังที่ซ่อนอยู่ เช่น คำพ้องความหมาย สำนวน การประชด หรือการอ้างอิงสิ่งที่กล่าวมาก่อนหน้า ทำให้ระบบจำนวนมากในยุคก่อนต้องพึ่งพากฎ (rule-based) หรือสถิติจากข้อมูลที่จำกัด จึงมักทำงานได้ดีเฉพาะกรณีที่รูปแบบชัดเจน แต่มักผิดพลาดเมื่อเจอประโยคซับซ้อนหรือบริบทที่ต้องเชื่อมโยงข้ามหลายประโยค ด้วยเหตุนี้ งานวิจัย NLP ในช่วงก่อนยุค LLM จึงมุ่งคิดค้นแบบจำลองและเทคนิคใหม่ ๆ เพื่อเพิ่มความสามารถในการจับความสัมพันธ์ของคำในระดับวลี–ประโยค–ย่อหน้า ลดความกำกวมของภาษา และค่อย ๆ ขยับจาก "การนับความถี่คำ" ไปสู่การแทนความหมายและบริบทที่ลึกขึ้น ซึ่งปูทางไปสู่ยุคของโมเดลเชิงลำดับและการเรียนรู้เชิงลึกในเวลาต่อมา

1.2.1 ยุคของ N-Gram และสถิติ

ในยุคแรก โมเดลภาษาอาศัยหลักการทางสถิติเป็นหลัก เช่น N-gram models ซึ่งทำนายคำถัดไปโดยดูจากความน่าจะเป็นของคำที่ปรากฏก่อนหน้าจำนวน $N-1$ คำ [7] แม้จะมีความแม่นยำในระดับหนึ่งสำหรับประโยคสั้น ๆ แต่ N-gram ทำงานผิดพลาดไม่สามารถใช้งานได้ในการจับใจความระยะยาว (Long-range dependencies) และขาดความเข้าใจในความหมายเชิงลึก (Semantic Meaning) ของคำ

รูปที่ 1.3 แสดงยุคแรกของงานประมวลผลภาษาธรรมชาติ (1990s–ต้น 2000s) โมเดลภาษานิยมใช้แนวคิด N-gram ซึ่งเป็นวิธีทำนาย "คำถัดไป" ด้วยสถิติแบบง่าย ๆ โดย มองย้อนกลับไปแค่ $N-1$ คำก่อนหน้าเท่านั้น

ตัวอย่างในภาพคือ Trigram ที่มี $N=3$ ดังนั้นโมเดลจะดูเพียง 2 คำล่าสุด ($3-1=2$ คำ) เพื่อทายคำถัดไป เช่นประโยคตัวอย่าง "ฉัน



รูปที่ 1.3: N-gram language models ใช้บริบทจำกัด $N-1$ คำ จึงเหมาะกับประโยคสั้น แต่มีข้อจำกัดกับ long-range dependencies และ semantic meaning

ไป ค้นหา บทความ $\rightarrow$?" ถ้าเป็น Trigram โมเดลจะใช้บริบทแค่ "ค้นหา บทความ" แล้วดูจากข้อมูลว่า หลังคำคู่แบบนี้ "มักตามด้วยคำอะไร" เช่น "วิจัย" "ออนไลน์" "ใหม่" เป็นต้น แล้วเลือกคำที่มีโอกาสเกิดหรือเป็นไปได้สูงสุด

แนวคิดนี้สามารถเขียนเป็นสมการได้ว่า

$$P(w_t \mid w_{t-1}, \ldots, w_{t-(N-1)}) \tag{1.1}$$

ซึ่งหมายถึง "ความน่าจะเป็นของคำถัดไป w_t เมื่อรู้คำก่อนหน้า $N-1$ คำ" พูดให้เข้าใจง่ายคือ นับสถิติจากข้อมูลจริงว่า 'คำชุดนี้' มักตามด้วย 'คำอะไร'

ข้อดีของ N-gram คือ โครงสร้างง่าย คำนวณเร็ว ใช้ทรัพยากรน้อย และให้ผลพอใช้ได้กับประโยคสั้นหรือรูปแบบซ้ำ ๆ แต่ข้อจำกัดสำคัญคือ โมเดล จับบริบทระยะยาวไม่ได้ เพราะดูได้แค่ไม่กี่คำ ไม่เข้าใจความหมายเชิงลึก แค่รู้ว่าคำไหนมักอยู่ใกล้กัน และมีปัญหา data sparsity คือถ้าเจอลำดับคำใหม่ ๆ ที่ไม่ค่อยปรากฏในข้อมูล โมเดลจะไม่มีสถิติพอทำให้ทำนายได้ไม่ดี

1.2.2 ยุคของ RNN และ LSTM

การนำ โครง ข่าย ประสาท เทียม มา ใช้ กับ ภาษา เริ่ม ต้น อย่าง จริงจัง ด้วย Recurrent Neural Network (RNN) ซึ่งถูกออกแบบมาเพื่อจัดการกับข้อมูลแบบลำดับ (Sequential Data) โดยเฉพาะ RNN มีกลไก "หน่วยความจำ" (Hidden State) ที่ส่งต่อข้อมูลจากคำก่อนหน้าไปยังคำถัดไป ทำให้โมเดลเริ่มมีความเข้าใจในบริบท [8]

อย่างไรก็ตาม RNN ประสบปัญหาทางคณิตศาสตร์ที่ร้ายแรงคือ Vanishing Gradient Problem กล่าวคือ ในประโยคที่มีความยาวมาก ข้อมูล Gradient ที่ใช้ในการปรับปรุงน้ำหนักของโมเดลจะค่อย ๆ ลดค่าลงจนเข้าใกล้ศูนย์เมื่อมีการส่งผ่านย้อนกลับ (Backpropagation) หลายขั้นตอน ทำให้โมเดลไม่สามารถเรียนรู้ความสัมพันธ์ของคำที่อยู่ในตำแหน่งที่ห่างกันมาก ๆ ได้ [8]

เพื่อแก้ไขปัญหานี้ สถาปัตยกรรมที่มีชื่อว่า Long Short-Term Memory (LSTM) และ Gated Recurrent Unit (GRU) ถูกพัฒนาขึ้นมาในช่วงทศวรรษที่ 1990 และ 2014 ตามลำดับ LSTM ประกอบด้วยเซลล์หน่วยความจำ (Memory Cell) ที่สามารถเก็บข้อมูลระยะยาวได้อย่างมีประสิทธิภาพ มีประตู (Gates) ที่ควบคุมการไหลของข้อมูลระหว่างเซลล์

ต่าง ๆ ในเครือข่าย ทำให้สามารถเลือกที่จะเก็บหรือทิ้งข้อมูลได้อย่างชาญฉลาด [9], [10] ประกอบไปด้วย Input Gate, Output Gate และ Forget Gate ที่ช่วยจัดการกับข้อมูลอย่างมีประสิทธิภาพ ส่วน GRU เป็นเวอร์ชันที่เรียบง่ายกว่า LSTM โดยรวมประตูบางส่วนเข้าด้วยกัน มีประตูเหลือแค่สองประตูคือ Update Gate และ Reset Gate ทำให้มีจำนวนพารามิเตอร์น้อยลงและฝึกฝนได้เร็วขึ้นในบางกรณี [10] วิวัฒนาการจาก RNN สู่ LSTM และ GRU แสดงในภาพที่ 1.4

1.3 Transformer: จุดเริ่มต้น AI ยุคใหม่

จุดเปลี่ยนครั้งประวัติศาสตร์เกิดขึ้นในปี 2017 เมื่อทีมวิจัย Google Brain นำเสนอบทความวิจัยเรื่อง *"Attention Is All You Need"* ซึ่งแนะนำสถาปัตยกรรม Transformer [11], [12] สถาปัตยกรรมนี้ได้ปลดล็อกข้อจำกัดของ RNN/LSTM โดยสิ้นเชิง ด้วยการละทิ้งการประมวลผลแบบลำดับและหันมาใช้กลไก Self-Attention

ภาพที่ 1.5 แสดง สถาปัตยกรรมของโมเดล Transformer แบบเต็ม (Encoder–Decoder) ซึ่งนิยมใช้ในงานแปลภาษาและงานสร้างข้อความ โดยแบ่งเป็น 2 ส่วนหลักคือ Encoder (ฝั่งซ้าย) และ Decoder (ฝั่งขวา) และทั้งสองส่วนประกอบด้วยบล็อกที่ซ้ำ ๆ กันหลายชั้น (stack)

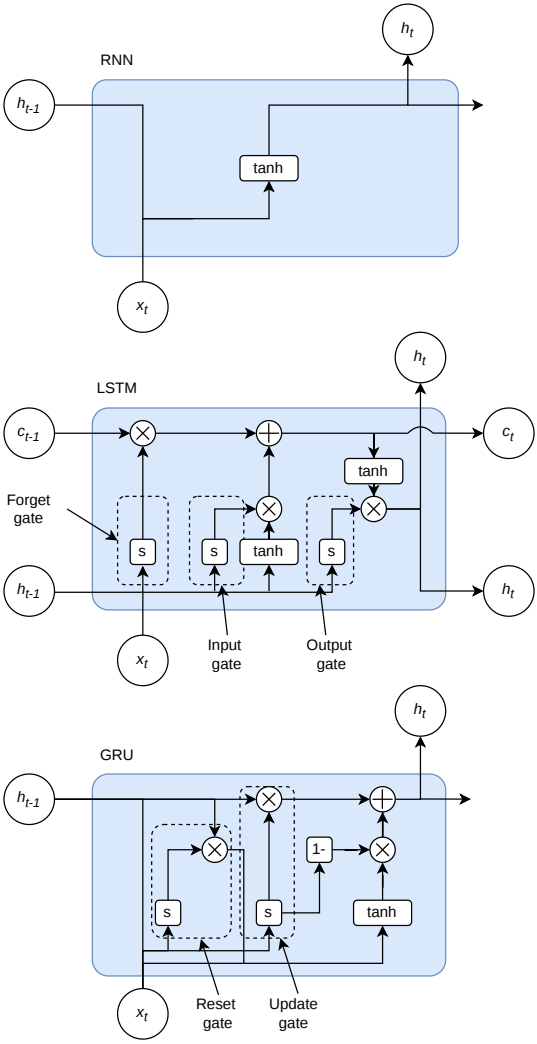
(1) ส่วนล่าง: Embeddings และข้อมูลตำแหน่ง

ข้อความ (คำ/โทเคน) จะถูกแปลงเป็นเวกเตอร์ด้วย Embeddings จากนั้นเติมข้อมูลตำแหน่งของคำ มักเรียกว่า Positional Encoding โดยสัญลักษณ์คล้ายคลื่นในภาพสื่อถึงการใส่ข้อมูลตำแหน่งและเครื่องหมาย "+" หมายถึงการบวกเวกเตอร์ (embedding + positional) เพื่อให้โมเดลรับรู้ลำดับคำในประโยค

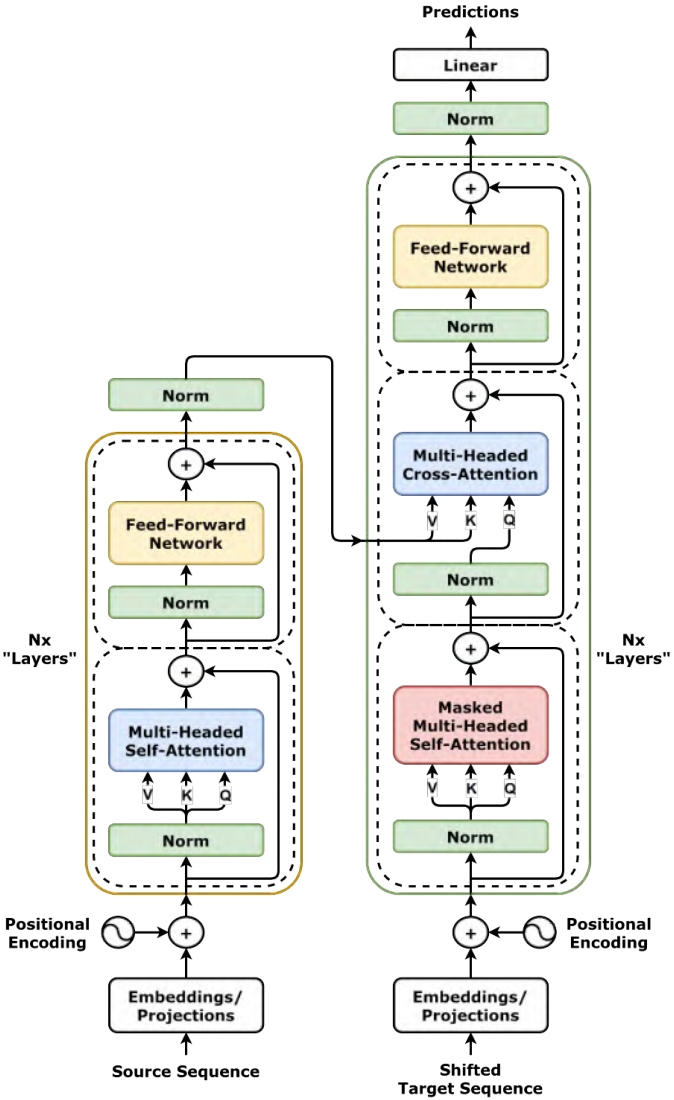
(2) Encoder (ฝั่งซ้าย)

Encoder ทำหน้าที่ "อ่าน" ประโยคต้นทางและสร้างตัวแทนความหมายเชิงบริบท (context representations) ภายใน 1 ชั้นของ Encoder มีองค์ประกอบหลัก 2 ส่วนคือ

1) **Multi-Headed Self-Attention** ให้แต่ละคำพิจารณาคำอื่น ๆ ใน



รูปที่ 1.4: วิวัฒนาการของแบบจำลองภาษา: จาก RNN สู่ LSTM และ GRU ที่มีประสิทธิภาพมากขึ้นในการจัดการบริบทระยะยาว



รูปที่ 1.5: สถาปัตยกรรม Transformer โดย Google [11]

ประโยคเดียวกันเพื่อจับความสัมพันธ์และบริบท โดย *multi-head* หมายถึงการมองความสัมพันธ์หลายมุมพร้อมกัน

2) **Feed-Forward Network (FFN)** เครือข่ายย่อยที่แปลงเวกเตอร์แบบรายตำแหน่ง เพิ่มความสามารถในการเรียนรู้รูปแบบที่ซับซ้อน

ในแต่ละบล็อกจะมี Norm (Layer Normalization) เพื่อให้การฝึกเสถียร และเครื่องหมาย "+" หมายถึง Residual ที่นำอินพุตเดิมมาบวกกับเอาต์พุตของบล็อก ช่วยให้โมเดลเรียนรู้แบบลึกได้ง่ายขึ้น ผลลัพธ์สุดท้ายของ Encoder จะถูกส่งต่อให้ Decoder ใช้ในขั้นตอน Cross-Attention

(3) Decoder (ฝั่งขวา)

Decoder ทำหน้าที่เขียนหรือสร้างข้อความปลายทางทีละคำ โดยอาศัยทั้งคำที่สร้างไปแล้ว และข้อมูลบริบทจาก Encoder ภายใน 1 ชั้นของ Decoder มีองค์ประกอบหลัก 3 ส่วนคือ

1) **Masked Multi-Headed Self-Attention** มีการทำงานคล้าย self-attention แต่มีการ *mask* เพื่อ "ห้ามมองคำในอนาคต" ทำให้การสร้างคำเป็นแบบ autoregressive คือเห็นได้เฉพาะคำก่อนหน้า

2) **Multi-Headed Cross-Attention** เป็นกลไกเชื่อม Encoder–Decoder เพื่อให้ Decoder "หันไปสนใจ" ส่วนที่เกี่ยวข้องของประโยคต้นทาง โดยทั่วไป **Query** (Q) มาจากฝั่ง Decoder ขณะที่ **Key** (K) และ **Value** (V) มาจากฝั่ง Encoder

3) **Feed-Forward Network (FFN)** แปลงเวกเตอร์ต่อเพื่อเพิ่มความสามารถไม่เป็นเชิงเส้น

เช่นเดียวกับ Encoder แต่ละบล็อกมี Norm และ Residual เพื่อช่วยให้การฝึกมีเสถียรภาพและรองรับโครงข่ายที่ลึก

(4) Linear หรือ Softmax

เอาต์พุตจาก Decoder จะผ่านชั้น Linear เพื่อแปลงเป็นคะแนนของคำในคลังคำศัพท์ (logits) และโดยปกติจะตามด้วย Softmax เพื่อได้ความน่าจะเป็นของ "คำถัดไป" ก่อนเลือกคำที่เหมาะสมในการสร้างประโยคต่อไป

Encoder ทำหน้าที่อ่านประโยคต้นทางและสร้างความหมายเชิงบริบทของทุกคำ ส่วน Decoder สร้างคำปลายทางทีละคำ โดย (1) มองคำก่อน

หน้าแบบ masked self-attention และ (2) อ้างอิงข้อมูลจาก Encoder ผ่าน cross-attention ก่อนแปลงเป็นคำถัดไปผ่าน linear/softmax

1.3.1 กลไกการทำงานของ Transformer

ความมหัศจรรย์ของ Transformer อยู่ที่ความสามารถในการประมวลผลคำทุกคำในประโยคได้ "พร้อมกัน" (Parallel) และสามารถสร้างความสัมพันธ์ระหว่างคำคู่ใด ๆ ในประโยคได้โดยตรง ไม่ว่าจะอยู่ห่างกันแค่ไหนก็ตาม โดยมีองค์ประกอบเชิงลึกดังนี้

1. **Tokenization และ Embeddings**
   กระบวนการเริ่มต้นด้วยการแปลงข้อความดิบให้เป็นหน่วยย่อยเรียกว่า Token ซึ่งอาจเป็นคำ หรือส่วนของคำ จากนั้นแต่ละ Token จะถูกแปลงเป็น Embedding Vector ซึ่งเป็นตัวแทนเชิงตัวเลขในมิติสูง (High-dimensional space) โดยคำที่มีความหมายใกล้เคียงกันจะมีค่าเวกเตอร์ที่ใกล้เคียงกัน [13]

2. **Positional Encoding**
   เนื่องจาก Transformer ประมวลผลข้อมูลแบบขนานและไม่มีกลไกการรับรู้ลำดับเวลาเหมือน RNN จึงจำเป็นต้องมีการเติมข้อมูลตำแหน่ง (Positional Information) เข้าไปใน Embedding เพื่อให้โมเดลทราบว่าคำใดมาก่อนหรือหลัง โดยใช้วิธีการทางคณิตศาสตร์ เช่น คลื่นไซน์/โคไซน์ (Sine/Cosine functions) เพื่อเข้ารหัสตำแหน่งสัมพัทธ์ [13]

3. **กลไก Self-Attention**
   นี่คือหัวใจสำคัญที่สุด กลไกนี้ช่วยให้โมเดลตัดสินใจได้ว่าควร "ให้ความสนใจ" (Attend) กับคำไหนมากที่สุดในขณะที่ประมวลผลคำคำหนึ่ง ในทางคณิตศาสตร์ กลไกนี้ทำงานโดยการสร้างเวกเตอร์ 3 ชุดสำหรับแต่ละ Token ได้แก่ Query (Q), Key (K), และ Value (V) [11] สมการของ Scaled Dot-Product Attention คือ

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \qquad (1.2)$$

   โดยมีความหมายดังนี้

   - Q คือตัวแทนของคำที่กำลังถูกพิจารณา (สิ่งที่กำลังค้นหาบริบท)

- K คือตัวแทนของคำอื่น ๆ ทั้งหมดในประโยค (สิ่งที่ถูกนำมาเทียบ)
- V คือเนื้อหาข้อมูลที่แท้จริงของคำนั้น ๆ

กระบวนการการทำงานคือการนำ Q ไปคูณกับ K แบบ Dot Product เพื่อหาความคล้ายคลึงกัน ผลลัพธ์ที่ได้คือคะแนนความสนใจ (Attention Score) ซึ่งบอกว่าคำสองคำมีความสัมพันธ์กันมากน้อยเพียงใด จากนั้นนำไปหารด้วยรากที่สองของมิติ ($\sqrt{d_k}$) เพื่อปรับสเกล และผ่านฟังก์ชัน Softmax เพื่อแปลงเป็นความน่าจะเป็น สุดท้ายนำไปคูณกับ V เพื่อดึงข้อมูลบริบทที่สำคัญออกมา [12] ตัวอย่างเช่น ในประโยค "The animal didn't cross the street because it was too tired", Self-Attention จะให้คะแนนความสัมพันธ์ระหว่าง "it" กับ "animal" สูงกว่า "it" กับ "street" ทำให้โมเดลตีความได้ถูกต้อง

4. **Multi-Head Attention**
   Transformer ไม่ได้ทำ Self-Attention เพียงครั้งเดียว แต่ทำหลาย ๆ ครั้งพร้อมกันในรูปแบบขนาน เรียกว่า "Heads" แต่ละ Head จะเรียนรู้ความสัมพันธ์ในมิติที่แตกต่างกัน เช่น Head หนึ่งอาจจับความสัมพันธ์ทางไวยากรณ์ อีก Head อาจจับความสัมพันธ์ทางความหมาย หรือการอ้างอิงสรรพนาม ผลลัพธ์จากทุก Head จะถูกนำมารวมกันเพื่อให้ได้ความเข้าใจที่สมบูรณ์และรอบด้าน [14]

1.3.2 เปรียบเทียบ RNN/LSTM และ Transformer

ตารางที่ 1.1 เปรียบเทียบความแตกต่างเชิงแนวคิดระหว่างแบบจำลองภาษาแบบลำดับอย่าง RNN/LSTM กับสถาปัตยกรรม Transformer ในหลายมิติสำคัญ เริ่มจากวิธีการประมวลผลของระบบ โดย RNN/LSTM ทำงานแบบเรียงลำดับทีละคำ (sequential) จึงไม่สามารถประมวลผลทั้งประโยคพร้อมกันได้ ขณะที่ Transformer รองรับการคำนวณแบบขนาน (parallel) ทำให้ประมวลผลโทเคนจำนวนมากได้พร้อมกัน [13] ประเด็นต่อมาคือ การจัดการบริบทระยะยาว ซึ่ง RNN/LSTM มักประสบปัญหา vanishing gradient เมื่อประโยคยาวขึ้น ส่งผลให้การจดจำบริบทของคำที่อยู่ห่างไกลกัน ทำได้จำกัด ในทางตรงกันข้าม Transformer ใช้กลไก attention ที่เชื่อมโยงโทเคนทุกตำแหน่งได้โดยตรง จึงรักษาและดึงบริบทระยะยาวได้ดีกว่าอย่างชัดเจน [8] ด้าน ความเร็วในการฝึกฝน การทำงานแบบ sequential ของ RNN/LSTM ทำให้ใช้ประสิทธิภาพของ GPU ได้ไม่เต็มที่และใช้เวลาฝึกนานกว่า ขณะที่ Transformer เหมาะกับการคำนวณแบบเมทริกซ์ขนาดใหญ่ จึงฝึกได้เร็วมากและขยายได้ดีบน GPU/

ตารางที่ 1.1: เปรียบเทียบคุณสมบัติของ RNN/LSTM และ Transformer

คุณสมบัติ	RNN/LSTM	Transformer
การประมวลผล	แบบลำดับ (Sequential) ทีละคำ	แบบขนาน (Parallel) ทั้งประโยคพร้อมกัน [13]
การจัดการบริบทระยะยาว	มีปัญหา vanishing gradient คือคำที่เกิดก่อนมานานจะลดความสำคัญลงมาก	ดีเยี่ยม คือเข้าถึงทุกคำได้โดยตรงผ่าน Attention [8]
ความเร็วในการฝึกฝน	ช้า (ไม่สามารถใช้ GPU เต็มประสิทธิภาพ)	เร็วมาก (ใช้ประโยชน์จาก GPU/TPU ได้เต็มที่) [14]
ความเข้าใจบริบท	ทางเดียว (หรือสองทางใน Bi-LSTM)	รอบทิศทาง Contextual awareness

TPU [14] สุดท้ายในมิติความเข้าใจบริบท RNN/LSTM โดยทั่วไปมองบริบทแบบทางเดียวคือซ้ายไปขวา หรือสองทางในกรณี Bi-LSTM ขณะที่ Transformer สามารถเรียนรู้บริบทแบบรอบทิศทางได้ในระดับประโยค ทำให้การแทนความหมายของคำมีความละเอียดและยืดหยุ่นมากขึ้น

1.4 กระบวนการสร้างและฝึกฝน LLM

การสร้าง LLM ระดับโลกอย่าง GPT, Gemini หรือ Claude ไม่ได้เกิดขึ้นในขั้นตอนเดียว แต่ผ่านกระบวนการที่ซับซ้อนและใช้ทรัพยากรในการฝึกฝนสูง โดยสามารถแสดงเป็นขั้นตอนได้ดังนี้

1. **Pre-training (การเรียนรู้แบบ Self-Supervised)**
   นี่คือขั้นตอนที่ใช้พลังงานการคำนวณมากที่สุด โมเดลจะถูกป้อนด้วยข้อมูลข้อความมหาศาล เช่น เว็บไซต์ Common Crawl, Wikipedia, GitHub และได้รับภารกิจให้ทำนายคำที่หายไป หรือที่เรียกว่า Masked Language Modeling เช่นใน BERT หรือทำนายคำถัดไปที่เรียกว่า Causal Language Modeling เช่นใน GPT [6] โดยไม่ต้องมีมนุษย์คอยเฉลยคำตอบ โมเดลจะเรียนรู้ไวยากรณ์ ข้อเท็จจริง และตรรกะทางภาษาโดยอัตโนมัติจากสถิติของข้อมูล ผลลัพธ์คือ Base Model ที่มีความรู้กว้างขวางแต่ยังควบคุมยากและอาจไม่ทำตามคำสั่ง

2. **Fine-tuning (การปรับแต่งเพื่อการใช้งานจริง)**
เพื่อให้โมเดลสามารถตอบคำถามหรือปฏิบัติตามคำสั่งได้ จำเป็นต้องผ่านกระบวนการ Instruction Tuning หรือ Supervised Fine-Tuning (SFT) โดยใช้ชุดข้อมูลที่มีคุณภาพสูงซึ่งประกอบด้วยคู่คำถาม–คำตอบ (Prompt–Response pairs) ที่เขียนโดยมนุษย์ [5]

3. **Reinforcement Learning from Human Feedback (RLHF)**
ขั้นตอนสุดท้ายที่ทำให้ LLM มีความ "เป็นมนุษย์" มากขึ้น ปลอดภัย และมีประโยชน์ คือการใช้เทคนิค RLHF โดยให้มนุษย์จัดอันดับคำตอบของโมเดลว่าคำตอบไหนดีกว่ากัน ข้อมูลนี้จะถูกนำไปฝึก *"Reward Model"* เพื่อนำไปปรับจูน LLM หลักอีกที วิธีนี้ช่วยลดแนวโน้มที่โมเดลจะสร้างเนื้อหาที่เป็นพิษหรือผิดเพี้ยนไปจากความต้องการของผู้ใช้ [15]

1.5 ความสำคัญของข้อมูลและทรัพยากรในการสร้าง LLM

การสร้าง LLM ที่มีประสิทธิภาพสูงไม่ใช่แค่เรื่องของสถาปัตยกรรมที่ดีเท่านั้น แต่ยังต้องอาศัยข้อมูลที่มีคุณภาพและทรัพยากรการคำนวณที่มหาศาลด้วย ความสำเร็จของ LLM สมัยใหม่เป็นผลมาจากการผสมผสานระหว่างสถาปัตยกรรมขั้นสูง ข้อมูลฝึกฝนขนาดใหญ่ และพลังการคำนวณที่ก้าวกระโดด ซึ่งทั้งสามปัจจัยนี้เป็นเสมือนขาตั้งสามขาที่ขาดไม่ได้ [16], [17]

1.5.1 ข้อมูลฝึกฝน: ความหลากหลายและคุณภาพ

ข้อมูลที่ใช้ในการฝึก LLM ต้องมีความหลากหลายและครอบคลุมหลายโดเมนเพื่อให้โมเดลมีความรู้กว้างขวางและสามารถตอบสนองต่อคำถามได้หลากหลาย ชุดข้อมูลที่นิยมใช้ประกอบด้วยแหล่งข้อมูลที่หลากหลาย เช่น

- **เว็บข้อมูลขนาดใหญ่** Common Crawl ซึ่งเป็นคลังข้อมูลเว็บไซต์ขนาด Petabyte ที่รวบรวมเนื้อหาจากอินเทอร์เน็ตทั่วโลก

- **หนังสือและบทความวิชาการ** เช่น Wikipedia, arXiv, PubMed เพื่อความรู้ที่มีโครงสร้างและความน่าเชื่อถือสูง
- **โค้ดโปรแกรม** GitHub และ StackOverflow เพื่อความสามารถในการเขียนและเข้าใจโค้ด
- **การสนทนาและโซเชียลมีเดีย** Reddit, Twitter/X เพื่อรูปแบบภาษาในชีวิตประจำวันและการสนทนา

คุณภาพของข้อมูลมีความสำคัญไม่น้อยไปกว่าปริมาณ ข้อมูลที่มีคุณภาพต่ำ เช่น ข้อมูลที่มีอคติ (bias) ข้อมูลที่ผิดพลาด หรือเนื้อหาที่เป็นอันตราย อาจส่งผลให้โมเดลสร้างเนื้อหาที่ไม่เหมาะสมหรือเผยแพร่ข้อมูลที่ผิดพลาด [18] ดังนั้น การคัดกรอง (filtering) และการทำความสะอาดข้อมูล (data cleaning) จึงเป็นขั้นตอนที่สำคัญในกระบวนการเตรียมข้อมูล

1.5.2 ทรัพยากรการคำนวณ: พลังงานและต้นทุนสูง

การฝึก LLM ขนาดใหญ่ต้องใช้พลังงานการคำนวณที่สูงมากและมีต้นทุนสูงลิ่ว โมเดลอย่าง GPT-3 ที่มี 175 พันล้านพารามิเตอร์ต้องใช้ทรัพยากรดังนี้ [16], [19]

- **GPU/TPU** ต้องใช้ GPU หรือ TPU หลายพันถึงหมื่นตัวทำงานพร้อมกันเป็นเวลานาน
- **เวลาในการฝึก** อาจใช้เวลาหลายสัปดาห์ถึงหลายเดือนในการฝึกโมเดลเพียงครั้งเดียว
- **พลังงาน ไฟฟ้า** การ ฝึก โมเดล หนึ่ง ครั้ง อาจ ปล่อย ก๊าซคาร์บอนไดออกไซด์เทียบเท่ากับรถยนต์ที่วิ่งไปมาข้ามทวีปหลายเที่ยว
- **ต้นทุนทางการเงิน** ค่าใช้จ่ายในการฝึกอาจสูงถึงหลายล้านถึงหลายสิบล้านดอลลาร์สหรัฐ

ปัจจัยเหล่านี้ทำให้การพัฒนา LLM ขนาดใหญ่จำกัดอยู่เพียงองค์กรที่มีทรัพยากรมากเท่านั้น เช่น บริษัทเทคโนโลยีขนาดใหญ่ เช่น OpenAI, Google, Meta, Anthropic และสถาบันวิจัยชั้นนำ [17] อย่างไรก็ตาม ชุมชนนักวิจัยกำลังพยายามพัฒนาวิธีการที่มีประสิทธิภาพมากขึ้น เช่น การใช้เทคนิค model compression, quantization และ เทคนิคอื่น ๆ เพื่อลดต้นทุนและผลกระทบต่อสิ่งแวดล้อม [20], [21]

1.5.3 Trade-off ระหว่างขนาดโมเดลและประสิทธิภาพ

การวิจัยในช่วงหลายปีที่ผ่านมาแสดงให้เห็นความสัมพันธ์ที่เรียกว่า *"Scaling Laws"* ซึ่งบอกว่าประสิทธิภาพของโมเดลมีความสัมพันธ์แบบ power-law กับขนาดของโมเดล ปริมาณข้อมูล และพลังการคำนวณ [17] กล่าวคือ การเพิ่มขนาดโมเดลและข้อมูลอย่างมีสัดส่วนจะทำให้ประสิทธิภาพดีขึ้นอย่างต่อเนื่อง แต่ในทางปฏิบัติ ต้องมีการหาจุดสมดุลระหว่าง

- ประสิทธิภาพในการทำงาน (Performance)
- ความเร็วในการตอบสนอง (Latency)
- ต้นทุนในการฝึกและใช้งาน (Cost)
- ผลกระทบต่อสิ่งแวดล้อม (Carbon footprint)

ดังนั้น การลงทุนในทรัพยากรเหล่านี้เป็นสิ่งจำเป็นเพื่อให้ได้ LLM ที่มีประสิทธิภาพและสามารถใช้งานได้จริงในหลากหลายแอปพลิเคชัน เช่น การสร้างเนื้อหา การตอบคำถาม การแปลภาษา การวิเคราะห์ข้อมูล และการช่วยเหลือในการเขียนโค้ด ซึ่งทั้งหมดนี้เป็นผลมาจากการผสมผสานระหว่างข้อมูลคุณภาพสูง สถาปัตยกรรมที่มีประสิทธิภาพ และทรัพยากรการคำนวณที่เพียงพอ [16], [22]

1.6 LLM สร้างข้อความได้อย่างไร

หลังจากที่เราได้เรียนรู้และเข้าใจหลักการทำงานของสถาปัตยกรรม Transformer และกระบวนการสร้าง LLM แล้ว คำถามสำคัญที่ตามมาคือ LLM สร้างข้อความได้อย่างไรในทางปฏิบัติ กระบวนการนี้แม้จะดูซับซ้อน แต่สามารถอธิบายได้ในเชิงแนวคิดผ่านขั้นตอนหลักที่เชื่อมโยงกันอย่างมีระบบได้ดังนี้ [23], [24]

LLM สมัยใหม่ที่ใช้กันส่วนใหญ่ใช้วิธีการสร้างข้อความที่เรียกว่า Autoregressive Generation ซึ่งหมายถึงการสร้างข้อความทีละคำ (หรือทีละ Token) โดยอาศัยคำที่สร้างไปแล้วทั้งหมดเป็นบริบทในการทำนายคำถัดไป [25] กระบวนการนี้ประกอบด้วยขั้นตอนดังนี้

1. **การรับ Prompt**
   ผู้ใช้จะป้อนข้อความ คำสั่ง คำถาม หรือเรียกรวม ๆ ว่า Prompt

เข้าสู่ระบบ เช่น "อธิบายหลักการทำงานของปัญญาประดิษฐ์" Prompt นี้จะถูกแปลงเป็น Token ผ่านกระบวนการ Tokenization และแปลงเป็น Embedding Vector พร้อมกับข้อมูลตำแหน่ง (Positional Encoding)

2. **การประมวลผลผ่าน Transformer**
Embedding Vector ทั้งหมดของ Prompt ที่ได้รับและแปลงมาจะถูกส่งเข้าสู่สถาปัตยกรรม Transformer โดยผ่านกระบวนการ Self-Attention และ Feed-Forward Network หลายชั้น ในแต่ละชั้น โมเดลจะวิเคราะห์ความสัมพันธ์ระหว่างคำต่าง ๆ และสร้างการแทนความหมายเชิงบริบท (Contextualized Representation) ที่ซับซ้อนยิ่งขึ้น [26]

3. **การทำนายคำถัดไป**
หลังจากประมวลผลแล้ว ชั้นสุดท้ายของโมเดล (Output Layer) จะสร้างการกระจายความน่าจะเป็น (Probability Distribution) สำหรับคำถัดไปที่เป็นไปได้ทั้งหมดในคลังคำศัพท์ (Vocabulary) ซึ่งอาจมีหลักหมื่นถึงแสนคำ โดยใช้สมการ

$$P(w_t \mid w_1, w_2, \ldots, w_{t-1}) = \text{Softmax}(W \cdot h_t + b) \qquad (1.3)$$

โดยที่ h_t คือ Hidden State ณ ตำแหน่ง t, W และ b คือพารามิเตอร์ที่ได้จากการฝึกฝน

4. **การเลือกคำ**
โมเดลจะเลือกคำถัดไปจากการกระจายความน่าจะเป็นโดยใช้กลยุทธ์ต่าง ๆ เช่น [27]

   - **Greedy Decoding** เลือกคำที่มีความน่าจะเป็นสูงสุดเสมอ ทำให้ผลลัพธ์คาดเดาได้ แต่อาจซ้ำซากเมื่อสร้างข้อความจากคำสั่งเดียวกันหลายครั้ง
   - **Beam Search** พิจารณาหลายเส้นทางที่เป็นไปได้พร้อมกัน แล้วเลือกเส้นทางที่มีความน่าจะเป็นรวมสูงสุด
   - **Temperature Sampling** ปรับระดับการกระจายความน่าจะเป็นในการสุ่ม โดย Temperature ต่ำ ($T < 1$) จะทำให้โมเดลเลือกคำที่แน่นอน ขณะที่ Temperature สูง ($T > 1$) จะเพิ่มความหลากหลายของคำที่ถูกเลือก
   - **Top-k และ Top-p Sampling** เป็นการจำกัดการเลือกเฉพาะคำที่มีความน่าจะเป็นสูง ๆ เท่านั้น เพื่อสมดุลระหว่างความสร้างสรรค์และความสมเหตุสมผล