

Prompt Engineering

ผศ. ดร. ชนันทกรณ์ จันแดง

สำนักวิชาสารสนเทศศาสตร์ มหาวิทยาลัยวลัยลักษณ์

◆ ความหมายของ Prompt

- **Prompt** คือ ข้อความ คำสั่ง หรือคำถาม ที่ผู้ใช้ป้อนเข้าไปใน LLM (Large Language Model)
- คุณภาพของ Prompt มีผลโดยตรงต่อ คุณภาพของคำตอบ
- การออกแบบ Prompt อย่างมีระบบ เรียกว่า **Prompt Engineering**

◆ หลักการสร้าง Prompt ที่ดี (1)

1. ชัดเจน (Clarity)

- ระบุสิ่งที่ต้องการอย่างตรงไปตรงมา
- ตัวอย่าง:
 - ❌ ไม่ดี: “อธิบายเรื่องแมว”
 - ✅ ดี: “อธิบายวงจรชีวิตของแมวตั้งแต่เกิดจนโตเต็มวัย โดยใช้ภาษาง่าย ๆ”

◆ หลักการสร้าง Prompt ที่ดี (2)

2. มีบริบท (Context)

- ให้ข้อมูลประกอบที่จำเป็น เพื่อช่วยให้โมเดลเข้าใจสิ่งที่ต้องการ
- ตัวอย่าง:
 - “อธิบายกฎของนิวตัน ข้อที่สาม สำหรับนักเรียนมัธยมปลาย”

◆ หลักการสร้าง Prompt ที่ดี (3)

3. มีรูปแบบชัดเจน (Format)

- ระบุรูปแบบผลลัพธ์ เช่น ย่อหน้า, bullet point, ตาราง
- ตัวอย่าง:
 - “สรุปข้อดีและข้อเสียของพลังงานแสงอาทิตย์ เป็น bullet point 5 ข้อ”

4. มีข้อจำกัด (Constraints)

- กำหนดเงื่อนไข เช่น ความยาวภาษา หรือจำนวนการเขียน
- ตัวอย่าง:
 - “เขียนเรียงความเกี่ยวกับความสำคัญของน้ำสะอาด ความยาวไม่เกิน 150 คำ”

◆ เทคนิคเพิ่มเติม

- **Few-shot Prompting**

ให้ตัวอย่างก่อน เช่น:

ตัวอย่าง:

Q: $2+3$ ได้เท่าไร?

A: 5

Q: $10+4$ ได้เท่าไร?

A: 14

Q: $7+8$ ได้เท่าไร?

A:

- **Chain-of-Thought Prompting**

กระตุ้นให้โมเดลอธิบายเหตุผลทีละขั้นตอน

- ตัวอย่าง: “อธิบายวิธีแก้โจทย์คณิตศาสตร์ข้อนี้ทีละขั้นตอน”

◆ ตัวอย่างสำหรับนักเรียน ม.ปลาย

- Prompt 1: “สรุปเนื้อหาเรื่องเซลล์พืชเป็น 3 ประโยค ใช้ภาษาที่เข้าใจง่าย”
- Prompt 2: “อธิบายทฤษฎีสัมพัทธภาพของไอน์สไตน์สำหรับนักเรียนมัธยม โดยใช้ตัวอย่างจากชีวิตประจำวัน”
- Prompt 3: “ช่วยเขียนโค้ด Python ง่าย ๆ ที่หาค่าเฉลี่ยของตัวเลขในลิสต์ พร้อมคำอธิบายที่ละเอียด”

Key Takeaway

- Prompt = คำสั่งที่ใช้สื่อสารกับ LLM
- การออกแบบ Prompt ที่ดี = ได้คำตอบที่ถูกต้อง ชัดเจน และตรงตามความต้องการ
- ทักษะนี้คือ “สะพานเชื่อม” ระหว่างผู้ใช้กับความสามารถของ LLM

แบบจำลอง (Framework) สำหรับการสร้างพรอมต์

1) หลักการออกแบบพรอมต์เชิงระบบ

- ความชัดเจน (Clarity)
- บริบท (Context)
- โครงแบบคำตอบ (Output Schema)
- ตัวอย่างกำกับ (Exemplars)
- พารามิเตอร์พฤติกรรม (Behavioral Controls)
- เงื่อนไขประเมินผล (Evaluation Criteria)

2) ตัวอย่างพรอมต์ (พร้อมคัดลอกใช้งานได้ทันที)

2.1 ตัวอย่าง CRISPE

[Context] บทเรียนวิทยาศาสตร์ ม.ปลาย เรื่อง "การเรียนรู้ของเครื่องพื้นฐาน"

[Role] ทำหน้าที่เป็นครูผู้สอนที่ใช้ภาษาชัดเจน เข้าใจง่าย แต่ถูกต้องทางวิชาการ

[Input] หัวข้อ: ความแตกต่างระหว่าง Supervised/Unsupervised Learning พร้อมตัวอย่างชีวิตจริง

[Steps] (1) ให้คำจำกัดความสั้น กระชับ (2) ยกตัวอย่างละ 2 กรณี (3) เปรียบเทียบเป็นตาราง (4) สรุปข้อควรระวัง

[Parameters] รูปแบบเอาต์พุตเป็น Markdown; ความยาวรวม ~250 คำ; โทนทางวิชาการระดับมัธยมปลาย

[Examples] ตัวอย่างตาราง: คอลัมน์ = วิธี, ข้อมูลที่ใช้, ตัวอย่าง, จุดแข็ง, ข้อจำกัด

2.2 ตัวอย่าง RTF

Role: ผู้ช่วยสอนวิชาคอมพิวเตอร์

Task: เขียนโค้ด Python ที่อ่านลิสต์ตัวเลขและคืนค่าเฉลี่ย ส่วนเบี่ยงเบนมาตรฐาน พร้อมอธิบายโค้ดที่ละบรรทัดอย่างย่อ

Format: โค้ดในบล็อก ```python``` และคำอธิบายในรายการ bullet ถัดจากโค้ด

2.3 ตัวอย่าง CO-STAR + ESCAPE

Context: มีบทความวิจัยหัวข้อการประยุกต์ใช้ RAG ในงานถามตอบเอกสาร

Objective: สรุปส่วนบทนำและงานที่เกี่ยวข้องให้เข้าใจง่ายสำหรับนักเรียน ม.ปลาย

Style/Tone: ทางวิชาการ เป็นกลาง

Audience: ผู้เรียนระดับมัธยมปลายที่มีพื้นฐาน ML เบื้องต้น

Response: สังออกเป็นหัวข้อย่อย (bullet) 6–8 ข้อ และปิดท้ายด้วย "แหล่งอ้างอิง" 2–3 รายการ

Evidence: สังเคราะห์เฉพาะจากข้อความที่ให้ หากไม่เพียงพอให้ระบุว่า "ข้อมูลไม่เพียงพอ"

Avoid: หลีกเลี่ยงการเดาข้อมูลที่ไม่มีในต้นฉบับ

Post-check: ตรวจสอบความสอดคล้องก่อนสรุป

2.4 ตัวอย่าง CRAFT

Constraints: เวลาเรียน 30 นาที; ระดับความยาก = ม.ปลาย; หลีกเลียงคำศัพท์เทคนิคเกินจำเป็น
Requirements: สร้างแบบฝึกหัด 5 ข้อ เกี่ยวกับ "Vector-Embedding-Indexing" มีเฉลยแบบสั้น
Actions: (1) นิยามย่อ (2) สร้างโจทย์ 5 ข้อ (3) ให้เฉลยกระชับแต่ถูกต้อง (4) แนบคำอธิบายข้อที่มักผิด
Format: Markdown แบ่งหัวข้อชัดเจน: นิยาม, แบบฝึกหัด, เฉลย, คำอธิบาย
Tests: ตรวจสอบว่าโจทย์ครอบคลุมทั้งนิยาม-ตัวอย่าง-การประยุกต์

2.5 พรอมต์ RAG QA

Role: ผู้ช่วยตอบคำถามจากเอกสาร

Task: ตอบคำถามโดยอ้างอิงเฉพาะเอกสารที่แนบ

Format: ส่งออกเป็น JSON:

```
{  
  "answer": "...",  
  "citations": ["ชื่อเอกสาร/หน้า"],  
  "confidence": "low|medium|high"  
}
```

Constraints: หากไม่มีหลักฐานในเอกสาร ให้ตอบ {"answer": "ข้อมูลไม่เพียงพอ", "citations": [], "confidence": "low"}

3) แนวทางแก้ปัญหาที่พบบ่อย

- คำตอบกว้าง/เลื่อนลอย → เพิ่ม Format, ความยาว, Few-shot
- ข้อเท็จจริงคลาดเคลื่อน → เพิ่ม Evidence/Citations, Post-check
- โครงสร้างไม่ถูกต้อง → ใช้ schema เช่น JSON/ตาราง
- ยาว/สั้นเกินไป → ระบุจำนวนคำ/ย่อหน้า
- สำนวนไม่เหมาะสม → กำหนด Audience และ Tone ชัดเจน

4) ชุดแม่แบบสรุปแบบพร้อมใช้

4.A สรุปบทเรียน

Role: ครูวิทยาการคอมพิวเตอร์ระดับ ม.ปลาย

Task: สรุปหัวข้อ {หัวข้อ} เป็น bullet 6–8 ข้อ โดยเชื่อมโยงกับตัวอย่างในชีวิตประจำวัน

Format: Markdown bullets; ความยาวรวม ~180–220 คำ; ปิดท้าย "คำศัพท์สำคัญ" 5 คำพร้อมคำอธิบายสั้น

4.B ออกแบบกิจกรรมห้องเรียน

Role: ผู้ออกแบบการเรียนรู้

Task: สร้างกิจกรรม 20 นาที สอนหัวข้อ {หัวข้อ} มีวัตถุประสงค์ ผลลัพธ์ที่คาดหวัง ขั้นตอน และเกณฑ์ประเมิน

Format: ตาราง 4 คอลัมน์: ส่วนประกอบ | รายละเอียด | เวลา | เครื่องมือประกอบ

4.C โค้ด + คำอธิบาย

Role: ผู้ช่วยสอนโค้ด

Task: เขียนโค้ด {ภาษา} ทำ {งาน} และอธิบายหลักการทีละขั้น

Format: บล็อกโค้ด + bullet อธิบายหลังโค้ด; หลีกเลี่ยงไลบรารีเกินจำเป็น

Constraints: ความยาวโค้ด ≤ 40 บรรทัด

