

Workshop:

04-สร้าง AI ตอบคำถามจากเอกสาร (Document Q&A) ด้วยแนวคิด RAG Framework

 ผู้สอน: ดร. CJ

Walailak University

เป้าหมายของ Workshop

- เข้าใจว่า AI อ่านและเข้าใจเอกสารได้อย่างไร
- รู้จักแนวคิด RAG – Retrieval Augmented Generation
- ได้ลอง สร้างระบบถาม-ตอบจากเอกสารจริง
- สร้าง Chatbot ผู้ช่วยอ่านหนังสือ ด้วยตัวเอง

ปัญหาที่เราพบ

“อยากรู้ใจความสำคัญของบทความ แต่ไม่อยากรอ่านทั้งหมด!”

AI สามารถช่วยเราได้โดย...

1. อ่านเอกสารแทนเรา
2. ค้นหาส่วนสำคัญ
3. ตอบคำถามจากข้อมูลจริง

แนวคิดของ RAG

RAG = Retrieval + Augmented + Generation

ส่วน	หน้าที่
Retrieval	ค้นหาข้อมูลจากเอกสาร
Augmented	เสริมบริบทให้ AI เข้าใจเนื้อหา
Generation	สร้างคำตอบด้วยภาษาที่มนุษย์เข้าใจ

ตัวอย่างในชีวิตจริง

ถ้าเราถามเพื่อนว่า

"บทที่ 3 วิทยาศาสตร์ พูดถึงอะไรนะ?"

เพื่อนจะไปเปิดหนังสือ

→ อ่านบางย่อหน้า

→ แล้วสรุปตอบให้เราเข้าใจง่าย

 นั่นแหละ! คือหลักการของ **RAG Framework**

⚙️ ขั้นตอนของระบบ RAG

1. 📄 แบ่งเอกสารเป็นชิ้นเล็ก ๆ (Chunking)
2. 🧠 ให้ AI สรุปแต่ละส่วน (Context)
3. 📊 แปลงข้อความเป็นตัวเลข (Embedding)
4. 🔍 ค้นหาส่วนที่เกี่ยวข้อง (Retriever)
5. 💬 สร้างคำตอบจากข้อมูลจริง (Generative Model)

ขั้นตอนที่ 1: Chunking

AI ไม่สามารถอ่านทั้งเล่มได้
เราต้อง “หั่น” เอกสารให้เป็นส่วนย่อย ๆ

```
text = "สมองของมนุษย์มีเซลล์ประสาทมากมาย..."  
chunks = [text[i:i+50] for i in range(0, len(text), 50)]  
print(chunks)
```

 ทำให้ AI ย่อยข้อมูลได้ง่ายขึ้น

ขั้นตอนที่ 2: สร้างบริบทด้วย LLM

ใช้ ChatGPT / Llama 3 / Gemini

ช่วย “สรุป” หรือ “อธิบาย” เนื้อหาแต่ละส่วน

```
prompt = "สรุปข้อความนี้ให้นักเรียนเข้าใจง่าย:\n\n" + chunk
```

 AI จะช่วยทำให้ข้อมูลเข้าใจง่ายขึ้น

12 **34** ขั้นตอนที่ 3: แปลงข้อความเป็นตัวเลข (Embedding)

AI เข้าใจ “ความหมาย” ผ่านตัวเลข
เรียกว่า เวกเตอร์ (Vector)

```
from sentence_transformers import SentenceTransformer
model = SentenceTransformer('all-MiniLM-L6-v2')
vec = model.encode(["สมองของมนุษย์..."])
print(vec[:10])
```

✦ “ตัวเลข” เหล่านี้แทนความหมายของข้อความ

ขั้นตอนที่ 4: ค้นหาข้อมูลที่เกี่ยวข้อง (Retrieval)

```
from sklearn.metrics.pairwise import cosine_similarity  
score = cosine_similarity(query_vec, embeddings)
```

 ค่าความใกล้เคียงมาก แปลว่าเนื้อหาเกี่ยวข้องกันมาก

ขั้นตอนที่ 5: ให้ AI สร้างคำตอบ

```
prompt = f"ตอบคำถามนี้โดยอ้างอิงจากข้อมูล:\n{docs}\nคำถาม: {question}"
```

AI จะสร้างคำตอบที่อิงจากข้อมูลจริง
ไม่ได้ “เดา” จากความจำของโมเดล

เราได้ระบบ RAG แบบง่ายแล้ว!

สรุปขั้นตอนทั้งหมด

- 1 แบ่งเอกสารเป็นชิ้นเล็ก ๆ
- 2 ให้ AI สรุปหรือเสริมบริบท
- 3 แปลงข้อความเป็นเวกเตอร์
- 4 ค้นหาส่วนที่ใกล้เคียงกับคำถาม
- 5 ให้ AI สร้างคำตอบใหม่

Workshop Activity

◆ ทดลองจริงใน Google Colab

1. เปิดไฟล์ `RAG_Workshop.ipynb`

2. คัดลอกโค้ดแต่ละขั้นตอน

3. พิมพ์คำถามเช่น




- "มนุษย์คิดได้อย่างไร?"
- "ระบบประสาทคืออะไร?"

4. ดูว่า AI ตอบอย่างไร

Workshop Challenge

สร้าง “AI ผู้ช่วยอ่านหนังสือเรียน” ของคุณเอง

ตัวอย่าง:

-  สรุปเนื้อหาวิทยาศาสตร์ บท 1
-  สรุปประวัติศาสตร์ชาติไทย
-  สรุปสาระจากบทเรียนคณิตศาสตร์






สรุปสิ่งที่เรียนรู้วันนี้

ขั้นตอน	ความหมาย
Chunking	แบ่งข้อความออกเป็นส่วนย่อย
Context	ให้ AI สรุปเนื้อหาแต่ละส่วน
Embedding	แปลงข้อความเป็นตัวเลข
Retrieval	ค้นหาข้อความที่เกี่ยวข้อง
Generation	สร้างคำตอบจากข้อมูลจริง


ถามตัวเองดูสิ...

- ถ้าเรานำ RAG ไปใช้กับ “บทเรียนในห้องเรียน” จะได้อะไรบ้าง?
- AI จะช่วยให้เรา “เข้าใจ” มากกว่าแค่ “จำ” ได้อย่างไร?
- ถ้าอยากให้มันฉลาดขึ้น เราจะเพิ่มข้อมูลแบบไหน?

ภารกิจต่อยอด (Extension Ideas)

-  ทำ RAG กับ หนังสือเรียนวิทยาศาสตร์
-  ทำ Chatbot สำหรับ ดิวสอบ O-NET
-  ทำสรุปบทเรียนแบบอัตโนมัติ

สรุป Workshop วันนี้


 ทุกคนได้เรียนรู้ว่า
AI อ่านเอกสารได้ด้วยการ
"ค้นหา (Retrieve)" + "สร้างคำตอบ (Generate)"

 หลักการนี้เรียกว่า **RAG Framework**
และเป็นพื้นฐานของ ChatGPT, Gemini, และ Copilot

สร้าง AI ผู้ช่วยของคุณเอง!

“ต่อไปนี่ เวลาอ่านหนังสือสอบ
คุณไม่ต้องอ่านคนเดียวอีกต่อไป”

 ขอขอบคุณนักเรียนทุกคนที่ร่วม Workshop

 ชนันทกรณ์ จันแดง – Walailak University