

บทที่ 1 • AI เดาคำตอบไปยังไง — ตัวอย่าง "แมวชอบกิน ___": ปลา 3, ข้าว 2, หนู 1, สก -1

ให้คะแนนทุก token → softmax เป็น % → ปรับด้วย temperature / top-p → สุ่มเลือก → ต่อกัน แล้ววนซ้ำ			
สิ่งที่ทำ	สูตร	จำง่ายๆ	ตัวอย่างในบทเรียน
คะแนน → %	$p_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$	ยก e กำลัง แล้วหารผลรวม	ปลา / ข้าว / หนู / สก: 65.7% / 24.2% / 8.9% / 1.2%
Temperature	$p_i = \frac{e^{z_i/T}}{\sum_j e^{z_j/T}}$	T ต่ำ = มั่นใจ/ซ้ำ · T สูง = หลากหลาย	T=0.5: 86.7% / 11.7% / 1.6% / 0.03% T=2: 47.4% / 28.7% / 17.4% / 6.4%
Top-p	เรียงมาก→น้อย สะสมจน ≥ p แล้วหารผลรวมใหม่	ตัดหางก่อนสุ่ม	p=0.9 เหลือ 3 คำ (สะสม 65.7% → 89.9% → 98.8%)
เลื่อนคะแนนทั้งชุด	$\text{softmax}(z + c) = \text{softmax}(z)$	สนใจแค่ระยะห่าง	(2, 1, 0) → 66.5% / 24.5% / 9.0%

บทที่ 2 • AI เข้าใจความหมายยังไง — แมว (0.9, 0.8) · หมา (0.8, 0.9) · สก (0.9, -0.2)

ความหมาย = เวกเตอร์ · ความคล้าย = cosine · การโฟกัส = softmax ของ dot product · ความจำ = context window · ความรู้ใหม่ = RAG			
สิ่งที่ทำ	สูตร	จำง่ายๆ	ตัวอย่างในบทเรียน
Dot product	$a \cdot b = \sum_i a_i b_i$	คูณคู่ แล้วบวก	แมว·หมา = 1.44
ความยาว	$\ a\ = \sqrt{\sum_i a_i^2}$	พิทาโกรัส	$\ \text{แมว}\ = 1.2042$
Cosine	$\cos \theta = \frac{a \cdot b}{\ a\ \ b\ }$	1 เหมือน · 0 ไม่เกี่ยว · -1 ตรงข้าม	แมว-หมา 0.9931 · แมว-สก 0.5855
Attention	$\text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V$	dot → softmax → ผสม	น้ำหนัก 67.3% / 16.4% / 16.4%
ต้นทุน attention	$\propto n^2$	ยาว 2 เท่า งาน 4 เท่า	1,000 token = 1M คู่ง · 2,000 = 4M คู่ง
RAG	ค้น chunk ที่ cosine กับคำถามสูงสุด แล้วแนบเข้า prompt	สอบบแบบเปิดหนังสือ	A 0.9899 · B 0.7746 · C 0.1414

บทที่ 3 • ข้างใน Transformer

token → เวกเตอร์ → [attention ให้คำคุณ้กัน + MLP ให้คิดต่อ] × L ชั้น บนทางหลัก residual → dot กับทุกคำ = logit			
สิ่งที่ทำ	สูตร	จำง่ายๆ	ตัวอย่างในบทเรียน
ตำแหน่ง (RoPE)	หมุน q, k ตามตำแหน่ง m, n ในทุกชั้น	คะแนนขึ้นกับระยะห่าง m - n	ห่าง 2 ช่อง: (3, 1) dot 0.5 · (5, 3) dot 0.5
Causal mask	$s_{ij} = -\infty$ เมื่อ $j > i$	ห้ามแอบดูคำอนาคต	(1, 2, 3) แถวที่ 2 → 26.9% / 73.1% / ปิด
หลายหัว (multi-head)	$d_{\text{head}} = d/h$	หลายคำถามพร้อมกัน	768 = 12 × 64
Residual + layer norm	$x \leftarrow x + f(\text{LN}(x))$	บวกเพิ่ม ไม่เขียนกับ	(0, 0.5) → (0.6, 0.6) → (0.8, 0.7)
MLP	ขยาย → ReLU($w \cdot x + b$) → ย่อ	คิดต่อทีละ token	ReLU(1 + 1 - 1.5) = 0.5
Unembedding	$\text{logit}_i = h \cdot e_i$	ทิศของ h เลือกคำ	h (2, 1) → (3, 2, 1, -1) → ปลา 65.7%
จำนวนพารามิเตอร์	$\approx 12 d^2 L$	MLP รว 2 ใน 3 ของแต่ละชั้น	GPT-2 small: d 768, L 12 → 84.9M (รวม embedding 124.3M)

บทที่ 4 • AI เรียนรู้อย่างไร

เดาคำตอบไปในข้อความจริง → วัดความผิดด้วย loss → หาความชันแล้วส่งย้อนทุกชั้น (backprop) → ขยับพารามิเตอร์ทีละก้าวเล็กๆ วนซ้ำ			
สิ่งที่ทำ	สูตร	จำง่ายๆ	ตัวอย่างในบทเรียน
วัดความผิด (loss)	$\text{loss} = -\ln p$ ของคำเฉลย	มั่นใจผิด = โดนหนัก	เฉลย ปลา / ข้าว / หนู / สก: 0.4197 / 1.4197 / 2.4197 / 4.4197
Perplexity	$e^{\overline{\text{loss}}}$	ลั้งเลที่ทาง ยิ่งต่ำยิ่งดี	เฉลย ปลา: $e^{0.4197} = 1.5215$
ทิศของคะแนน	ความชันต่อ logit = $p - y$	ที่ทาย ลบ ที่ถูก	-0.3428, +0.2418, +0.0889, +0.0120 · 1 ก้าว: ปลา 65.7% → 76.6%
ก้าวฝึก	$w \leftarrow w - \eta g$ (g = ความชัน)	ๆ พอดีๆ	สามชาไทย: η 0.5 ถึงกัน 1 ก้าว · η 1.1 ระเบิด
Backprop	คูณความชันย้อนกลับทีละชั้น (กฎลูกโซ่)	ส่งความผิดย้อนทุกชั้น	0.5 คูณกัน 10 ชั้น = 1/1,024 → เลื่อนหาย (residual ช่วย)
ค่าคำนวณการฝึก	$C \approx 6ND$	~20 token ต่อพารามิเตอร์ (คุ่มงบ)	N 1B → D 20B → C ≈ 1.2 × 10 ²⁰

บทที่ 5 · จากเครื่องเดาคำ ผู้ช่วย

base model ต่อข้อความ → chat template บอกว่าใครพูด → SFT สอนบทบาท → reward model ให้คะแนน → RLHF ใส่คะแนนแบบมีเบรค (KL)

สิ่งที่ทำ	สูตร	จำง่ายๆ	ตัวอย่างในบทเรียน
Base model	ต่อข้อความที่น่าจะตามมา ไม่ได้ทำตามคำสั่ง	จัดฉากให้ดู	"เรียน ทีมทุกคน": สั่งตรงๆ 5.2% → จัดฉากเป็นอีเมล 90.1%
SFT	$loss = -\ln p$ ของคำตอบตัวอย่าง	สูตรเดิม ข้อมูลใหม่	"เรียน ทีมทุกคน": loss 1.7128 → 0.0642
ความชอบ (Bradley-Terry)	$P(A \succ B) = \sigma(r_A - r_B)$	softmax ของสองคำตอบ	r 1.5 vs 0.2 → 78.6%
RLHF	$r - \beta \cdot KL$	ใส่คะแนนแบบมีเบรค	$\beta = 1$: สุภาพ ชัดเจน 1.20 · เอาใจ 0.40
เดาหรือไม่	ตอบเมื่อ $p > \frac{w}{1+w}$	ไม่หักคะแนนตอนผิด = เดาคุ่มเสมอ	ผิดหัก $w = 0 / 1 / 3 \rightarrow$ ต้องมั่นใจเกิน 0.0% / 50.0% / 75.0%

บทที่ 6 · ตอนมันตอบจริง — โมเดลสมมติ 8B · การ์ด 24 GB

อ่าน prompt รวดเดียว แล้วเขียนทีละ token → ความเร็วถูกจำกัดด้วยการอ่านหน่วยความจำ → KV cache, quantization, batching และ MoE ช่วยให้เร็วและถูกลง

สิ่งที่ทำ	สูตร	จำง่ายๆ	ตัวอย่างในบทเรียน
เวลาตอบ	รอคำแรก $\approx$ token prompt ÷ ความเร็ว prefill · เขียน $\approx$ token คำตอบ ÷ ความเร็ว decode	อ่านรวด เขียนทีละคำ	prompt 7,000 + ตอบ 300 token → 1.4 s + 7.5 s
งานต่อ token	$\approx 2N$	MoE นับเฉพาะส่วนที่ใช้	8B → ~16B ครั้งต่อ token
KV cache	$2 \cdot L \cdot h_{kv} \cdot d_h \cdot \text{byte} \cdot n$	โตตามความยาว × จำนวนคน	131,072 byte/token · 128K token = 16.8 GB
น้ำหนักโมเดล	$N \times \text{bit} \div 8$	บิตน้อย = ไฟล์เล็ก	8B: 16 บิต 16.0 GB → 4 บิต 4.0 GB
Quantization	$scale = \max w \div (2^{b-1} - 1)$	บิตน้อย = เขียนมาก	4 บิต: คลาดสูงสุด 0.060
ความเร็ว decode	ความเร็วหน่วยความจำ ÷ ขนาดโมเดล (× batch)	อ่านครั้งเดียว แจกหลายคน	62.5 → 500 token/s เมื่อ batch 8
หน่วยความจำการ์ด	น้ำหนัก + KV ของทุกคน + อื่นๆ ≤ ความจุ	ทุกอย่างต้องลงการ์ด	การ์ด 24 GB, 8B ที่ 16 บิต → เหลือที่ KV 53,405 token
MoE	รวม = ส่วนกลาง + E × expert · ใช้จริง = ส่วนกลาง + k × expert	ประหยัดจำนวน ไม่ประหยัดหน่วยความจำ	รวม 34B · ใช้ต่อ token 10B
ค่าใช้จ่าย + แคช	token × ราคาต่อล้าน ÷ 1,000,000	ส่วนคงที่ไว้ต้น → แคช	แคช 6,000 จาก 7,000 token → บิลลด 63.5%

บทที่ 7 · คิดเป็นขั้น ใช้เครื่องมือ เป็น agent

คิด = เขียนกระดาษทดลง context · โหวต = ถามหลายรอบ · เครื่องมือ = เขียนใบสั่งให้โปรแกรม · agent = วนลูป และความผิดพลาดคูณสะสม

สิ่งที่ทำ	สูตร	จำง่ายๆ	ตัวอย่างในบทเรียน
Chain-of-thought	งานต่อ token $\approx 2N$	เขียนขั้นกลาง = ได้คิดเพิ่ม	ตอบ 60 token ได้คำนวณ 30 เท่าของตอบ 2 token
Self-consistency	$p^3 + 3p^2(1-p)$	$p > 0.5$ โหวตช่วย · $p < 0.5$ โหวตทำร้าย	p 0.7: ถาม 1 รอบ 70.0% → โหวต 3 รอบ 78.4%
Tool calling	JSON → โปรแกรมทำ → ผลต่อท้าย context	โมเดลไม่ได้กดปุ่มเอง	วันลาคงเหลือทั้งทีม: เครื่องมือตอบ 15.5 วัน
Agent	$P = p^n$	ขั้นเยอะ ต้องมีจุดตรวจ	95% × 10 ขั้น → 59.9% · มีจุดตรวจ → 93.0%
Multimodal	$\frac{W}{16} \times \frac{H}{16}$	ภาพใหญ่ = token เยอะ	ภาพ 224×224 → 196 แผ่น
Prompt injection	คำสั่ง = ข้อมูล = token	จำกัดสิทธิ์ + คนอนุญาติ	ข้อความในเอกสารภายนอกสั่งโมเดลได้ ถ้าไม่กั้น
Eval	ผ่านที่ข้อจาก 20–50 ข้อ	ต่าง 1 ข้อ ยังสรุปไม่ได้	17/20 = 85.0% vs 18/20 = 90.0% · ต่าง 5.0 จุด แต่คลาดได้ ~10.4 จุด

ใช้กับงานจริงพร้อมนี้

งานข้อเท็จจริง / ตัวเลข / ข้อโต้แย้ง → temperature ต่ำ · ระดมไอเดีย → สูงขึ้น แล้วขอหลายแบบมาเลือก	กันมั่ว → แบบเอกสาร (RAG) · ขอให้อ้างอิง · อนุญาตให้ตอบว่า "ไม่ทราบ" · ตรวจสอบชื่อ ตัวเลข วันที่เสมอ
เอกสารยาว → วางคำถามไว้ท้าย (หรือทั้งต้นและท้าย) และตัดเฉพาะส่วนที่เกี่ยวข้อง	ถามแบบเป็นกลาง → ไม่ชี้นำคำตอบ และใส่ข้อมูลที่ต้องใช้ให้ครบ อย่าให้ AI เติมเอง
ประหยัดเวลาและค่าใช้จ่าย → บอกความยาวคำตอบที่ต้องการ · วางส่วนที่คงที่ไว้ต้น prompt	งานหลายขั้น → เปิดโหมดคิดนาน และให้แสดงขั้นตอนก่อนสรุป เพื่อเช็คตัวเลขกลางทาง
Agent → ให้สิทธิ์น้อยที่สุด และกดยืนยันเองก่อนส่งหรือลบอะไร	เปลี่ยน prompt → ลองกับชุดทดสอบ 20 ข้อจากงานจริงทุกครั้ง ต่าง 1 ข้อยังสรุปไม่ได้

ตัวเลขเป็นตัวอย่างจำลองที่ตั้งขึ้นให้คิดตามได้ (ยกเว้นขนาด GPT-2 small ซึ่งมาจากงานที่เผยแพร่) โมเดลจริงใหญ่มาก มีคลังคำนับแสน token และเวกเตอร์หลายพันมิติ · เนื้อหาเพื่อการศึกษามาก่อน · ไม่เกี่ยวข้องกับการจ้างหรือองค์กรใด · ChatGPT, Claude, Gemini เป็นเครื่องหมายการค้าของเจ้าของ · ตัวละครมีจุด · ภาพสร้างด้วย AI · © 2026 ZENITH