
เครือข่าย GAN และโมเดลภาษา LLM

บทนำสู่ Generative AI

Generative AI คือการเปลี่ยนแปลงครั้งใหญ่ในวงการปัญญาประดิษฐ์ โดยเปลี่ยนจากโมเดลที่ทำหน้าที่แค่จำแนกหรือทำนาย ไปสู่โมเดลที่สามารถ “สร้าง” ข้อมูลใหม่ได้ โดยเรียนรู้การกระจายของข้อมูลต้นฉบับ แล้วสร้างตัวอย่างใหม่ที่มีลักษณะคล้ายกัน

โมเดลที่ทรงพลังที่สุดในกลุ่มนี้ ได้แก่:

- **GAN (Generative Adversarial Networks)**: เหมาะสำหรับการสร้างภาพหรือข้อมูลโครงสร้าง
- **LLM (Large Language Models)**: เชี่ยวชาญด้านการเข้าใจและสร้างภาษามนุษย์

แม้จะทำงานในรูปแบบต่างกัน แต่ทั้งสองต่างใช้ Deep Learning และข้อมูลขนาดมหาศาล และเมื่อรวมกันจะนำไปสู่ระบบ AI แบบหลายรูปแบบ (Multimodal AI) ที่สามารถให้เหตุผลข้ามระหว่างข้อความ ภาพ และวิดีโอ

เจาะลึก GAN – การเรียนรู้แบบแข่งขัน

GAN ประกอบด้วยเครือข่ายประสาทสองตัว:

- **Generator (G)**: สร้างข้อมูลที่คล้ายกับข้อมูลจริง
- **Discriminator (D)**: แยกแยะระหว่างข้อมูลจริงและข้อมูลปลอมที่ G สร้างขึ้น

การฝึกฝนเป็นเกมแบบ Minimax:

- G พยายามหลอก D ให้เชื่อว่าข้อมูลปลอมเป็นของจริง
- D พยายามตรวจจับข้อมูลปลอมให้แม่นยำที่สุด

กระบวนการนี้นำไปสู่การวนกลับแบบป้อนกลับ (feedback loop) ซึ่งทั้งสองเครือข่าย—Generator และ Discriminator—จะปรับปรุงประสิทธิภาพของตนเองอย่างต่อเนื่องในแต่ละรอบการฝึก โดยเริ่มจาก Generator ที่รับข้อมูลสุ่ม (โดยทั่วไปจะสุ่มจากการแจกแจงแบบเกาส์เซียนหรือแบบสม่ำเสมอ) แล้วแปลงข้อมูลนั้นให้กลายเป็นผลลัพธ์ที่มีโครงสร้างและคล้ายกับข้อมูลจริง

จากนั้น Discriminator จะทำหน้าที่ประเมินทั้งข้อมูลจริงและข้อมูลที่ Generator สร้างขึ้น พร้อมทั้งคำนวณค่าความชัน (gradient) ซึ่งถูกส่งกลับไปยัง Generator เพื่อใช้ในการปรับปรุงการเรียนรู้ให้ดีขึ้นในรอบถัดไป

ประเภทของ GAN ที่นิยม ได้แก่:

- **DCGAN**: ใช้ CNN สำหรับการสร้างภาพ
- **WGAN**: ใช้ระยะ Wasserstein เพื่อเพิ่มเสถียรภาพ
- **StyleGAN**: สร้างภาพความละเอียดสูงด้วยการควบคุมสไตล์

ความท้าทายในการฝึก GAN และแนวทางแก้ไข

ความยากในการฝึก GAN และความท้าทายหลัก

การฝึก GAN นั้นขึ้นชื่อว่า “ยาก” อย่างมาก เนื่องจากลักษณะการเรียนรู้แบบแข่งขันระหว่างสองเครือข่าย ซึ่งนำไปสู่ความไม่เสถียรและปัญหาหลายประการที่พบได้บ่อย ได้แก่:

1. การล่มของรูปแบบ (Mode Collapse)

Generator อาจเรียนรู้ที่จะสร้างข้อมูลเพียงชุดเล็ก ๆ ที่สามารถหลอก Discriminator ได้อย่างสม่ำเสมอ ซึ่งส่งผลให้ความหลากหลายของข้อมูลลดลง และเกิดการสร้างตัวอย่างที่ซ้ำกันหรือเหมือนกันเกินไป

แนวทางแก้ไข:

สามารถใช้เทคนิคต่าง ๆ เพื่อส่งเสริมความหลากหลาย เช่น

- Minibatch Discrimination (การเปรียบเทียบหลายตัวอย่างในชุดข้อมูลย่อย)

- Unrolled GANs (การคลายลูปการฝึกเพื่อให้เห็นผลลัพธ์ล่วงหน้า)

- Feature Matching (การจับคู่คุณลักษณะแทนการแยกแยะจริง/ปลอม)

2. การสูญเสียความชัน (Vanishing Gradients)

หาก Discriminator แข็งแกร่งเกินไป Generator จะได้รับข้อมูลย้อนกลับ (feedback) ที่น้อยมากหรือไม่มีเลย ทำให้การเรียนรู้หยุดชะงัก

แนวทางแก้ไข:

WGAN (Wasserstein GAN) เข้ามาแทนที่การใช้ฟังก์ชันสูญเสียแบบ Binary Cross-Entropy ด้วยฟังก์ชันสูญเสียแบบ Wasserstein ซึ่งให้ความชันที่นุ่มนวลและต่อเนื่องมากกว่า ช่วยให้ Generator สามารถเรียนรู้ได้อย่างมีประสิทธิภาพมากขึ้น

3. ความไม่เสถียรในการฝึก (Training Instability)

GAN มักเกิดการสั่น (oscillation) หรือเบี่ยงเบน (divergence) ระหว่างการฝึก เนื่องจากลักษณะของภูมิทัศน์ฟังก์ชันสูญเสีย (loss landscape) ที่ไม่เป็นนูน (non-convex) ซึ่งทำให้การหาค่าที่เหมาะสมที่สุดเป็นเรื่องยากและไม่แน่นอน

แนวทางแก้ไข:

- ปรับค่า learning rate อย่างระมัดระวัง

- ใช้เทคนิค spectral normalization เพื่อควบคุมขนาดของพารามิเตอร์

- ปรับสมดุลจำนวนรอบการฝึกระหว่าง Generator และ Discriminator ให้เหมาะส**4. Evaluation Metrics**

GAN ไม่มีตัวชี้วัดที่ชัดเจนเหมือนโมเดลแบบ Supervised

ตัวชี้วัดที่ใช้:

- Inception Score (IS)
- Fréchet Inception Distance (FID)
- Precision และ Recall สำหรับโมเดล Generative

การประยุกต์ใช้ GAN ในโลกจริง

GAN ถูกนำไปใช้ในหลายอุตสาหกรรม:

การแพทย์

- สร้างภาพ MRI หรือ CT ปลอมเพื่อฝึกโมเดลวินิจฉัย
- เพิ่มความละเอียดของภาพสแกน

ศิลปะและการออกแบบ

- โอนสไตล์ระหว่างภาพ
- สร้างงานศิลป์ใหม่หรือต้นแบบผลิตภัณฑ์

วิทยาศาสตร์

- จำลองการชนของอนุภาคหรือการไหลของของเหลว
- สร้างข้อมูลหายากเพื่อการวิเคราะห์

เกมและโลกเสมือน

- สร้างพื้นผิว ตัวละคร และฉากที่สมจริง
- สร้างเนื้อหาแบบ Procedural

ธุรกิจและการเงิน

- สร้างข้อมูลตารางปลอมเพื่อวิเคราะห์แบบไม่ละเมิดความเป็นส่วนตัว
- จำลองสถานการณ์ตลาดเพื่อวิเคราะห์ความเสี่ยง

GAN ยังถูกใช้ในการสร้าง Deepfake ซึ่งมีประเด็นจริยธรรมเกี่ยวกับการบิดเบือนข้อมูลและการแอบอ้างตัวตน

LLM – โมเดลภาษาขนาดใหญ่

LLM คือเครือข่ายประสาทลึกที่ถูกฝึกให้เข้าใจและสร้างภาษามนุษย์ โดยใช้สถาปัตยกรรม Transformer และข้อมูลขนาดมหาศาล เช่น หนังสือ บทความ โค้ด และบทสนทนา

เป้าหมายในการฝึก:

- **Causal Language Modeling**: ทำนาย Token ถัดไป (เช่น GPT)
- **Masked Language Modeling**: ทำนาย Token ที่ถูกซ่อนไว้ (เช่น BERT)

LLM เรียนรู้:

- ไวยากรณ์และโครงสร้างภาษา
- ความหมายและบริบท
- ความรู้ทั่วไป
- การให้เหตุผลและการสรุป

การใช้งาน:

- แชตบอทและผู้ช่วยเสมือน
- สรุปข้อความ
- แปลภาษา
- สร้างโค้ด
- วิเคราะห์เอกสารทางกฎหมายหรือการแพทย์

สถาปัตยกรรม Transformer – เครื่องยนต์ของ LLM

Transformer คือหัวใจของ LLM โดยแทนที่การวนซ้ำด้วย Attention ทำให้สามารถประมวลผลแบบขนานและเข้าใจบริบทระยะไกลได้

ส่วนประกอบหลัก:

1. **Input Embedding**
   แปลง Token เป็นเวกเตอร์ที่มีความหมาย
2. **Positional Encoding**
   เพิ่มข้อมูลตำแหน่งให้กับ Token เพื่อรักษาลำดับ
3. **Multi-Head Self-Attention**
   ให้โมเดลสามารถโฟกัสหลายจุดในข้อความพร้อมกัน
4. **Feedforward Network**
   แปลงข้อมูลด้วยฟังก์ชันไม่เชิงเส้น เช่น ReLU หรือ GELU
5. **Layer Normalization และ Residual Connections**
   ช่วยให้การฝึกเสถียรและ Gradient ไหลได้ดี
6. **การซ้อน Layer**
   GPT ใช้ Decoder อย่างเดียว, BERT ใช้ Encoder, T5 ใช้ทั้งสอง

จุดแข็งของ Transformer:

- ประมวลผลแบบขนาน
- เข้าใจบริบททั้งข้อความ
- ขยายขนาดได้ดี

การฝึกและปรับแต่ง LLM

ขั้นตอนการฝึก LLM:

1. Pretraining

- ฝึกด้วยข้อความที่ไม่มีป้ายกำกับ
- ใช้ทรัพยากรคอมพิวเตอร์มหาศาล
- เรียนรู้ความเข้าใจทั่วไปของภาษา

2. Fine-Tuning

- ปรับให้เหมาะกับงานเฉพาะ เช่น ถามตอบ สรุป

3. Alignment

- ปรับให้ผลลัพธ์สอดคล้องกับความต้องการของมนุษย์
- ใช้เทคนิค RLHF (Reinforcement Learning from Human Feedback)

4. Parameter-Efficient Tuning

- LoRA: เพิ่ม Matrix ขนาดเล็กที่ฝึกได้
- Prompt Tuning: ปรับ Prompt โดยไม่เปลี่ยนน้ำหนักโมเดล

ตัวชี้วัด:

- Perplexity
- BLEU/ROUGE
- การประเมินโดยมนุษย์

LangChain และการเชื่อมต่อกับ LLM

LangChain คือเฟรมเวิร์กสำหรับสร้างแอปพลิเคชันที่ใช้ LLM โดยช่วยจัดการ Prompt, Memory และการเรียกใช้เครื่องมือ

จุดเด่น:

- Prompt Templates
- Chains สำหรับเรียกหลายโมเดล

- Agents ที่ใช้เหตุผลในการเลือกเครื่องมือ
- Memory สำหรับเก็บประวัติ
- เชื่อมต่อกับ API และฐานข้อมูล

ตัวอย่างการใช้งาน

- ตัวช่วยวิจัยอัตโนมัติ
- เครื่องมือวิเคราะห์เอกสารทางกฎหมาย
- ติวเตอร์ส่วนบุคคล
- ผู้ช่วยด้านการให้เหตุผลแบบหลายขั้นตอน

LangChain ช่วยให้นักพัฒนาสร้างระบบอัจฉริยะที่ก้าวข้ามการตอบสนองแบบคงที่—โดยสร้างปฏิสัมพันธ์ที่มีบริบทและเปลี่ยนแปลงได้ตามสถานการณ์

เปรียบเทียบ GAN กับ LLM

คุณสมบัติ	**GAN**	**LLM**
ประเภทข้อมูล	ภาพ ข้อมูลโครงสร้าง	ข้อความ โค้ด ภาษา
สถาปัตยกรรม	Generator + Discriminator	Transformer-based
วัตถุประสงค์การฝึก	หลอก Discriminator ให้เชื่อว่าเป็นจริง	ทำนายคำที่หายไปหรือคำถัดไป
ผลลัพธ์	ภาพหรือข้อมูลเทียม	ข้อความที่มีความหมายและต่อเนื่อง
การประเมินผล	ความสมจริงของภาพ, FID score	Perplexity, BLEU, feedback จากมนุษย์
การใช้งาน	ศิลปะ การจำลอง การเพิ่มข้อมูล	แชตบอท สรุปโค้ด เขียนโปรแกรม
ความท้าทาย	Mode collapse, ความไม่เสถียร	การสร้างข้อมูลเท็จ, ข้อจำกัด token

ทั้งสองเป็นโมเดลแบบ generative แต่มีกลไกและการใช้งานที่แตกต่างกัน

ทิศทางในอนาคต

อนาคตของ Generative AI อยู่ที่โมเดลแบบ multimodal ซึ่งสามารถเข้าใจและสร้างข้อมูลได้หลายรูปแบบ เช่น ข้อความ ภาพ เสียง และวิดีโอ ตัวอย่างเช่น:

- **DALL·E**: การสร้างภาพจากข้อความโดยใช้การแพร่แบบทรานส์ฟอร์เมอร์
- **GPT-4V**: ผสานการมองเห็นและภาษาเพื่อการบรรยายภาพและให้เหตุผลเชิงภาพ
- **Sora (โดย OpenAI)**: การสร้างวิดีโอจากข้อความ

โมเดล GANs และ LLMs กำลังกลายเป็นเทคโนโลยีที่เสริมกันมากขึ้นเรื่อย ๆ โดย GANs สร้างข้อมูลฝึกแบบสังเคราะห์สำหรับ LLMs ขณะที่ LLMs ใช้ในการอธิบาย ตีความ และแม้แต่ชี้นำผลลัพธ์จาก GANs

แนวโน้มที่กำลังเกิดขึ้น ได้แก่:

- สถาปัตยกรรมแบบรวมสำหรับการสร้างมัลติโมดัล
- ปัญญาประดิษฐ์เชิงตัวแทน (Agentic AI) ที่สามารถวางแผนและดำเนินงาน
- กรอบจริยธรรมสำหรับการสร้างเนื้อหาอย่างมีความรับผิดชอบ

เมื่อโมเดลเหล่านี้พัฒนาไป พวกมันจะไม่เพียงแค่ “สร้าง” แต่ยัง “ร่วมมือกัน” กลายเป็นแกนหลักของเครื่องจักรอัจฉริยะที่มีความคิดสร้างสรรค์

Additional materials

make LLM

https://medium.com/sciforce/step-by-step-guide-to-your-own-large-language-model-2b3fed6422d0

https://symbl.ai/developers/blog/a-guide-to-building-an-llm-from-scratch/

https://medium.com/@iamamellstephen/how-to-build-a-private-llm-a-comprehensive-guide-296eae0e7db9

coding

Langchain

LangChain is an open-source framework designed to simplify the development of applications powered by large language models (LLMs), allowing developers to easily integrate LLMs with external data sources and build complex, context-aware applications.

https://python.langchain.com/docs/tutorials/llm_chain/

Last update 8 September 2025