Generative AI
How Machines Create

From DALL-E to the Future of AI

Nipun Batra | IIT Gandhinagar

The Journey Complete

Week	Topic	The Question
1 - 2	Foundations	How do machines learn?
3 - 4	Supervised Learning	How do we predict?
5	Neural Networks	What makes deep learning special?
6	Computer Vision	How do machines see?
7	Language Models	How do machines understand text?
8	**Generative AI**	**How do machines create?**

Today's Agenda

1. **Discriminative vs Generative** - Two paradigms
2. **Image Generation** - From GANs to Diffusion
3. **Multimodal AI** - Text + Images + More
4. **Using LLM APIs** - Building with AI
5. **Fine-tuning** - Customizing models
6. **The Future** - What's next?

Part 1: Two Paradigms

Discriminative vs Generative

Discriminative Models

Learn to DISTINGUISH between classes

$$P(\text{label}|\text{input})$$

"Given this image, what's the probability it's a cat?"

Input	Output
Image	"cat" (95%) or "dog" (5%)
Text	"spam" or "not spam"

Everything we've learned so far!

Generative Models

Learn to CREATE new data

$$P(\text{data}) \quad \text{or} \quad P(\text{data}|\text{prompt})$$

"Generate a new image that looks like a cat"

Input	Output
"A cat on a rainbow"	**New image**
"Write a poem about AI"	**New text**

The Landscape

Domain	Generative Tool	What It Creates
Text	ChatGPT, Claude	Essays, code, poems
Images	DALL-E, Midjourney	Any image from text
Audio	Suno, ElevenLabs	Music, voices
Video	Sora, Runway	Video clips
3D	DreamFusion	3D models
Code	Copilot, Cursor	Working programs

Part 2: Image Generation

From GANs to Diffusion

A Brief History

Year	Model	Key Innovation
2014	GANs	Generator vs Discriminator game
2020	VQVAE	Discrete image tokens
2021	DALL-E	Text-to-image at scale
2022	Stable Diffusion	Open-source, diffusion models
2023	DALL-E 3, Midjourney v5	Photorealistic quality
2024	Flux, SD3	Even better quality

GANs: The Generator-Discriminator Game

Generator

- Creates fake images
- Tries to fool discriminator
- Gets better at faking

Discriminator

- Tells real from fake
- Tries to catch generator
- Gets better at detecting

Both improve until generated images are indistinguishable from real!

Diffusion Models: The New King

Idea: Learn to **denoise** images

```
Training:
Real image → Add noise → Noisy image
             ↑
    Model learns to reverse this!

Generation:
Pure noise → Denoise → Denoise → ... → Final image
```

Diffusion: Step by Step

Step	Image State	What Happens
0	Pure noise	Start with random pixels
1	Mostly noise	Model removes some noise
2	Shapes emerge	Structure appears
…	…	…
50	Clear image	Final result

Each step removes a little noise!

Text-to-Image: How It Works

```
Input: "A photo of a cat wearing a hat on Mars"
       ↓

  1. Text Encoder (CLIP)                 → Text embedding
  2. Diffusion Model (guided by text)    → Denoising
  3. VAE Decoder                         → Final image

       ↓
Output: Image of a cat in a hat on Mars!
```

Using Image Generation

```python
from openai import OpenAI

client = OpenAI()

response = client.images.generate(
    model="dall-e-3",
    prompt="A serene Japanese garden with cherry blossoms",
    size="1024×1024",
    quality="standard",
    n=1,
)

image_url = response.data[0].url
```

Prompt Engineering for Images

Bad Prompt	Good Prompt
"cat"	"A fluffy orange tabby cat sleeping on a velvet cushion, soft lighting, photorealistic"
"landscape"	"Misty mountain landscape at sunrise, oil painting style, dramatic clouds, warm golden light"

Key elements: Subject, style, lighting, composition, quality modifiers

Part 3: Multimodal AI

Text + Images + More

What is Multimodal?

Modality = Type of data (text, image, audio, video)

Multimodal = Understanding/generating multiple types

Model	Modalities
GPT-4V	Text input + Image input → Text output
DALL-E	Text input → Image output
Gemini	Text + Image + Audio → Text
GPT-4o	Text + Image + Audio ↔ Text + Audio

Vision-Language Models

Input: Image + Text question

Output: Text answer

```
response = client.chat.completions.create(
    model="gpt-4-vision-preview",
    messages=[{
        "role": "user",
        "content": [
            {"type": "text", "text": "What's in this image?"},
            {"type": "image_url", "image_url": {"url": image_url}}
        ]
    }]
)
```

Use Cases

Task	Input	Output
Image Captioning	Photo	Description
Visual QA	Photo + Question	Answer
OCR + Understanding	Document image	Extracted info
Code from Screenshot	UI mockup	Working code

Part 4: Using LLM APIs

Building with AI

The OpenAI API Pattern

```
from openai import OpenAI

client = OpenAI()

response = client.chat.completions.create(
    model="gpt-4",
    messages=[
        {"role": "system", "content": "You are a helpful assistant."},
        {"role": "user", "content": "Explain quantum computing"}
    ],
    temperature=0.7
)

print(response.choices[0].message.content)
```

Message Roles

Role	Purpose	Example
system	Set behavior	"You are a Python tutor"
user	User input	"How do I read a file?"
assistant	Model response	"You can use open()..."

```
messages = [
    {"role": "system", "content": "Be concise."},
    {"role": "user", "content": "What is Python?"},
    {"role": "assistant", "content": "A programming language."},
    {"role": "user", "content": "What's it used for?"}
]
```

Key Parameters

Parameter	Controls	Range
temperature	Randomness	0.0 (deterministic) to 2.0 (random)
max_tokens	Response length	1 to context limit
top_p	Nucleus sampling	0.0 to 1.0
frequency_penalty	Repetition	-2.0 to 2.0

Prompt Engineering Basics

Technique	Example
Be specific	"Write a 3-paragraph summary" not "Summarize"
Give examples	"Format: Name: X, Age: Y"
Role-play	"You are an expert data scientist..."
Step-by-step	"Think through this step by step"

Building Applications

```python
def analyze_sentiment(text):
    response = client.chat.completions.create(
        model="gpt-4",
        messages=[
            {"role": "system", "content": """
                Analyze sentiment of the text.
                Return JSON: {"sentiment": "positive/negative/neutral",
                              "confidence": 0.0-1.0}
            """},
            {"role": "user", "content": text}
        ],
        temperature=0
    )
    return json.loads(response.choices[0].message.content)
```

Part 5: Fine-tuning

Customizing Models

When to Fine-tune?

Scenario	Use...
General task	Prompt engineering
Specific style/format	Fine-tuning
Domain knowledge	RAG (Retrieval)
Custom behavior	Fine-tuning

Fine-tuning Overview

```
1. PREPARE DATA
   - Format: {"messages": [{"role": …, "content": …}]}
   - Need 50-1000+ examples

2. UPLOAD DATA
   - Upload to OpenAI/Hugging Face

3. TRAIN
   - Fine-tune on your data
   - Usually takes minutes to hours

4. USE
   - Call your custom model
```

Fine-tuning with OpenAI

```
# 1. Upload training file
file = client.files.create(
    file=open("training_data.jsonl", "rb"),
    purpose="fine-tune"
)

# 2. Create fine-tuning job
job = client.fine_tuning.jobs.create(
    training_file=file.id,
    model="gpt-3.5-turbo"
)

# 3. Use your model
response = client.chat.completions.create(
    model="ft:gpt-3.5-turbo:org:custom-name:id",
    messages=[ ... ]
)
```

Hugging Face: Open Models

```python
from transformers import AutoModelForCausalLM, AutoTokenizer

# Load model
model_name = "meta-llama/Llama-2-7b"
tokenizer = AutoTokenizer.from_pretrained(model_name)
model = AutoModelForCausalLM.from_pretrained(model_name)

# Generate
inputs = tokenizer("Hello, how are", return_tensors="pt")
outputs = model.generate(**inputs, max_length=50)
print(tokenizer.decode(outputs[0]))
```

RAG: Retrieval-Augmented Generation

Problem: LLMs don't know your private data

Solution: Retrieve relevant documents, add to context

```
1. User asks question
2. Search your documents for relevant chunks
3. Add chunks to prompt
4. LLM answers using retrieved context
```

Part 6: The Future

What's Next?

Current Capabilities

Task	State
Text generation	Excellent
Code generation	Very good
Image generation	Excellent
Video generation	Emerging
Audio generation	Good
Reasoning	Improving rapidly

Emerging Trends

Trend	What It Means
Agents	AI that takes actions, uses tools
Reasoning models	o1/o3 - think before answering
Multimodal	Seamless text/image/audio
Smaller models	Run on phones, edge devices
Open weights	Llama, Mistral, etc.

AI Agents

Traditional LLM: Answer questions

AI Agent: Take actions!

```
# Agent can:
# - Search the web
# - Run code
# - Send emails
# - Book appointments
# - Write and execute programs
```

Reasoning Models (o1/o3)

Standard LLM: Immediate response

Reasoning model: Think step-by-step internally

Model	Math Score	Science Score
GPT-4	52%	64%
o1	83%	78%
o3	91%	87%

Challenges Ahead

Challenge	Why It Matters
Hallucinations	Models make up facts
Bias	Reflects training data biases
Alignment	Ensuring helpful, safe behavior
Cost	Training = millions of dollars
Environment	Massive energy consumption
Jobs	Automation concerns

Responsible AI

Principle	Implementation
Transparency	Disclose AI use
Fairness	Test for bias
Privacy	Don't train on private data
Safety	Content filtering
Accountability	Human oversight

Course Summary

What We Learned

Your AI Journey

Week	You Learned
1-2	ML fundamentals, data, train/test
3	Linear/Logistic Regression, Trees, KNN
4	Cross-validation, Ensembles, Clustering
5	Neural networks, PyTorch
6	CNNs, Object detection, YOLO
7	Embeddings, Attention, Transformers
8	Generative AI, APIs, Future

The Core Ideas

1. **ML = Learning from data** (not explicit programming)
2. **Supervised learning** is most common
    - Classification (categories) vs Regression (numbers)
3. **Neural networks** can learn complex patterns
    - CNNs for images, Transformers for sequences
4. **Attention is all you need**
    - Modern AI is built on transformers
5. **Generative AI** creates new content
    - Text, images, audio, video

The Skills You Built

Skill	Tools
ML basics	sklearn, pandas, numpy
Deep learning	PyTorch
Computer vision	CNNs, YOLO
NLP	Transformers, APIs
Generative AI	OpenAI API, Hugging Face

Where to Go Next

Deepen Understanding

- Fast.ai courses
- Stanford CS229, CS231n
- Coursera/Udacity

Build Projects

- Kaggle competitions
- Personal projects
- Open source contributions

Stay Current

- arXiv papers
- AI newsletters
- Twitter/X AI community

Specialize

- Computer Vision
- NLP
- Reinforcement Learning
- AI Safety

Key Resources

Resource	What It Offers
Hugging Face	Pre-trained models, datasets
Papers With Code	Latest research + implementations
Kaggle	Competitions, notebooks, data
Fast.ai	Practical deep learning course
3Blue1Brown	Visual math intuition
Andrej Karpathy	Deep learning from scratch

Key Takeaways

1. **AI is pattern recognition at scale**
2. **Data is everything** — garbage in, garbage out
3. **Start simple** — complex ≠ better
4. **Evaluate properly** — test set is sacred
5. **AI is a tool** — you decide how to use it

Congratulations!

You Now Understand Modern AI

"The best way to predict the future is to create it."

— Alan Kay

Go build something amazing!

Questions?