

LLM Architectures, Prompt Engineering, and AI Ethics

Lecture Overview

This lecture covers modern LLM architectures, prompt engineering techniques, and important ethical considerations in AI development.

Topics Covered:

- LLM scaling laws
- Modern architectures
- Prompt engineering techniques
- AI ethics and bias
- AI safety and alignment

1. Large Language Model Architectures

1.1 Scaling Laws

```
print("LLM Architectures and Scaling")
print("="*70)

print("Scaling Laws (Kaplan et al., 2020):")
print("  Performance improves predictably with:")
print("    - Model size (parameters)")
print("    - Dataset size (tokens)")
print("    - Compute (FLOPs)")

print("\nEmpirical findings:")
print("  Loss ~ C^(-0.05) where C is compute budget")
print("  Larger models are more sample efficient")

print("\nModel sizes:")
print(f"{'Model':<20} {'Params':>12} {'Year':>8}")
print("-" * 42)
for m, p, y in [
    ("GPT-2", "1.5B", "2019"),
    ("GPT-3", "175B", "2020"),
    ("PaLM", "540B", "2022"),
    ("LLaMA 2", "70B", "2023"),
    ("GPT-4", "~1.8T*", "2023"),
    ("Claude 3", "Unknown", "2024"),
]:
    print(f"{m:<20} {p:>12} {y:>8}")
print("*Estimated, uses Mixture of Experts")

print("\nKey architectural innovations:")
print("  - Rotary Position Embeddings (RoPE)")
print("  - Group Query Attention (GQA)")
print("  - SwiGLU activation")
print("  - RMSNorm instead of LayerNorm")

LLM Architectures and Scaling
=====
Scaling Laws (Kaplan et al., 2020):
  Performance improves predictably with:
    - Model size (parameters)
    - Dataset size (tokens)
    - Compute (FLOPs)

  Empirical findings:
    Loss ~ C^(-0.05) where C is compute budget
    Larger models are more sample efficient

  Model sizes:
  Model           Params      Year
  -----          -----
  GPT-2           1.5B       2019
  GPT-3           175B      2020
  PaLM            540B      2022
  LLaMA 2         70B       2023
  GPT-4           ~1.8T*    2023
  Claude 3        Unknown   2024
  *Estimated, uses Mixture of Experts

  Key architectural innovations:
    - Rotary Position Embeddings (RoPE)
    - Group Query Attention (GQA)
    - SwiGLU activation
    - RMSNorm instead of LayerNorm
```

2. Prompt Engineering

2.1 Prompting Techniques

```
print("Prompt Engineering")
print("="*70)

print("Zero-shot prompting:")
print("""
    Prompt: "Classify the sentiment: 'This movie was amazing! ''"
    Output: "Positive"
""")

print("Few-shot prompting:")
print("""
    Prompt: "Classify sentiment:
        'Great product!' -&gt; Positive
        'Terrible service' -&gt; Negative
        'This movie was amazing!' -&gt; "
    Output: "Positive"
""")

print("Chain-of-Thought (CoT):")
print("""
    Prompt: "Roger has 5 tennis balls. He buys 2 cans of 3.
        How many does he have? Let's think step by step."
    Output: "Roger starts with 5 balls.
        2 cans of 3 balls = 6 balls.
        5 + 6 = 11 balls total."
""")

print("\nPrompt engineering tips:")
print("  1. Be specific and clear")
print("  2. Provide examples (few-shot)")
print("  3. Use delimiters for structure")
print("  4. Specify output format")
print("  5. Ask for step-by-step reasoning")

Prompt Engineering
=====
Zero-shot prompting:

Prompt: "Classify the sentiment: 'This movie was amazing! ''"
Output: "Positive"

Few-shot prompting:

Prompt: "Classify sentiment:
        'Great product!' -&gt; Positive
        'Terrible service' -&gt; Negative
        'This movie was amazing!' -&gt; "
Output: "Positive"

Chain-of-Thought (CoT):

Prompt: "Roger has 5 tennis balls. He buys 2 cans of 3.
        How many does he have? Let's think step by step."
Output: "Roger starts with 5 balls.
        2 cans of 3 balls = 6 balls.
        5 + 6 = 11 balls total.

Prompt engineering tips:
  1. Be specific and clear
  2. Provide examples (few-shot)
  3. Use delimiters for structure
  4. Specify output format
  5. Ask for step-by-step reasoning
```

2.2 Advanced Prompting

```
print("Advanced Prompting Techniques")
print("="*70)
```

```
print("Self-consistency:")
print("  - Generate multiple CoT paths")
print("  - Take majority vote on answers")
print("  - Improves reasoning accuracy")

print("\nTree of Thoughts (ToT):")
print("  - Explore multiple reasoning paths")
print("  - Backtrack if path seems wrong")
print("  - Good for complex planning")

print("\nRetrieval-Augmented Generation (RAG):")
print("  1. Query vector database for relevant docs")
print("  2. Include retrieved context in prompt")
print("  3. Generate answer based on context")

print("\nSystem prompts:")
print('  "You are a helpful AI assistant specialized in coding.')
print('    Always explain your code and follow best practices."')

print("\nPrompt templates:")
print("  template = ''")
print("  Context: {context}")
print("  Question: {question}")
print("  Answer the question based only on the context.")
print("  ''")
```

Advanced Prompting Techniques

=====

Self-consistency:

- Generate multiple CoT paths
- Take majority vote on answers
- Improves reasoning accuracy

Tree of Thoughts (ToT):

- Explore multiple reasoning paths
- Backtrack if path seems wrong
- Good for complex planning

Retrieval-Augmented Generation (RAG):

1. Query vector database for relevant docs
2. Include retrieved context in prompt
3. Generate answer based on context

System prompts:

"You are a helpful AI assistant specialized in coding.
Always explain your code and follow best practices."

Prompt templates:

```
template = ''
Context: {context}
Question: {question}
Answer the question based only on the context.
'''
```

3. AI Ethics and Safety

3.1 Ethical Considerations

```
print("AI Ethics and Safety")
print("=*70)

print("Key ethical concerns:")

print("\n1. Bias and Fairness")
print(" - Training data reflects historical biases")
print(" - Models can amplify discrimination")
print(" - Need diverse datasets and evaluation")

print("\n2. Privacy")
print(" - Models may memorize training data")
print(" - Personal information leakage risks")
print(" - Differential privacy techniques")

print("\n3. Misinformation")
print(" - LLMs can generate convincing false info")
print(" - Deepfakes and synthetic media")
print(" - Need for detection and watermarking")

print("\n4. Environmental Impact")
print(" - Training large models: massive energy use")
print(" - GPT-3 training: ~1,300 MWh")
print(" - Need for efficient architectures")

print("\n5. Job Displacement")
print(" - Automation of cognitive tasks")
print(" - Need for reskilling programs")
print(" - Human-AI collaboration")

AI Ethics and Safety
=====
Key ethical concerns:

1. Bias and Fairness
- Training data reflects historical biases
- Models can amplify discrimination
- Need diverse datasets and evaluation

2. Privacy
- Models may memorize training data
- Personal information leakage risks
- Differential privacy techniques

3. Misinformation
- LLMs can generate convincing false info
- Deepfakes and synthetic media
- Need for detection and watermarking

4. Environmental Impact
- Training large models: massive energy use
- GPT-3 training: ~1,300 MWh
- Need for efficient architectures

5. Job Displacement
- Automation of cognitive tasks
- Need for reskilling programs
- Human-AI collaboration
```

3.2 AI Safety

```
print("AI Safety and Alignment")
print("=*70)

print("Alignment: Making AI do what we want")

print("\nRLHF (Reinforcement Learning from Human Feedback):")
print(" 1. Train reward model from human preferences")
print(" 2. Fine-tune LLM to maximize reward")
```

```

print(" 3. Used by ChatGPT, Claude, etc.")

print("\nConstitutional AI:")
print(" - Define principles (constitution)")
print(" - Model self-critiques against principles")
print(" - Reduces need for human labeling")

print("\nRed teaming:")
print(" - Adversarial testing before deployment")
print(" - Find harmful behaviors and fix them")
print(" - Ongoing process")

print("\nResponsible AI practices:")
print(" - Transparency about capabilities/limitations")
print(" - Human oversight for high-stakes decisions")
print(" - Regular auditing and monitoring")
print(" - Clear documentation and model cards")

print("\nAs AI practitioners:")
print(" - Consider societal impact of your work")
print(" - Test for bias in your models")
print(" - Be transparent about limitations")
print(" - Engage with ethics discussions")

AI Safety and Alignment
=====
Alignment: Making AI do what we want

RLHF (Reinforcement Learning from Human Feedback):
1. Train reward model from human preferences
2. Fine-tune LLM to maximize reward
3. Used by ChatGPT, Claude, etc.

Constitutional AI:
- Define principles (constitution)
- Model self-critiques against principles
- Reduces need for human labeling

Red teaming:
- Adversarial testing before deployment
- Find harmful behaviors and fix them
- Ongoing process

Responsible AI practices:
- Transparency about capabilities/limitations
- Human oversight for high-stakes decisions
- Regular auditing and monitoring
- Clear documentation and model cards

As AI practitioners:
- Consider societal impact of your work
- Test for bias in your models
- Be transparent about limitations
- Engage with ethics discussions

```

Summary

Key Takeaways:

- LLM performance scales predictably with compute
- Modern LLMs use RoPE, GQA, SwiGLU
- Few-shot and CoT improve performance
- RAG grounds responses in retrieved context
- AI systems can perpetuate biases
- RLHF aligns models with human preferences
- Responsible AI requires ongoing vigilance

Practice Exercises:

1. Compare zero-shot vs few-shot prompting
2. Implement chain-of-thought prompting
3. Build simple RAG system
4. Audit a model for bias
5. Write a model card for your project