Faysal Aziz

Architecting Intelligent Systems with LLMs and Agentic AI

From Prompt Engineering to Multi-Agent Orchestration with Python, Linux, LoRA, QLoRA, and Kafka - 3 days

FA-0108 · 3 days

Agentic & Generative AI · Software Development · DevOps, Cloud & Infrastructure

---

Introduction

Modern AI systems are no longer limited to static chatbot interfaces or isolated machine learning models. Enterprises are rapidly moving toward autonomous and semi-autonomous AI architectures capable of reasoning, coordinating tasks, invoking tools, interacting with distributed systems, and operating at scale. This course is designed to bridge the gap between foundational LLM usage and production-grade Agentic AI systems built using modern open-source tooling.

Participants will move beyond basic prompting into advanced prompt engineering methodologies, parameter-efficient fine-tuning techniques such as LoRA and QLoRA, retrieval-enhanced architectures, AI agent frameworks, and multi-agent orchestration patterns. The training emphasizes practical implementation using Python on Linux environments, with Kafka-based communication layers for scalable agent collaboration and distributed event-driven AI workflows.

The course is delivered from an industry implementation perspective rather than an academic research perspective. The instructor brings over 30 years of industry experience and focuses heavily on architectures, workflows, tooling, deployment patterns, and operational realities currently demanded in enterprise AI environments.

Learning outcomes

By the end of the course, participants will be able to:

- Understand modern LLM architectures and inference workflows
- Design advanced prompts for reasoning, planning, tool usage, and agent execution
- Implement structured prompting and chain-based reasoning workflows
- Build and evaluate Retrieval-Augmented Generation (RAG) pipelines
- Understand tokenization, embeddings, attention mechanisms, and transformer fundamentals
- Fine-tune open-source LLMs using LoRA and QLoRA
- Work with quantization techniques for efficient model deployment
- Prepare datasets for supervised fine-tuning and instruction tuning
- Evaluate and benchmark fine-tuned models
- Build autonomous AI agents using Python
- Implement tool-calling and function-calling architectures
- Design memory-enabled and stateful AI agents
- Build multi-agent systems with coordinated workflows
- Implement Kafka-driven event-based agent orchestration
- Design scalable distributed AI agent architectures on Linux environments
- Integrate vector databases, APIs, and external systems into agent pipelines
- Understand observability, tracing, governance, and operational concerns for AI agents

Prerequisites

- Strong Python programming skills
- Intermediate Linux command-line proficiency
- Experience with virtual environments and package management
- Understanding of REST APIs and JSON
- Familiarity with Git and Git workflows
- Basic understanding of machine learning concepts
- Experience using Docker containers
- Working knowledge of asynchronous programming concepts
- Familiarity with message queues or distributed systems concepts
- Basic understanding of GPU concepts and CUDA environments
- Comfortable reading technical documentation

Training outline

1. Foundations of Large Language Models

Evolution of Generative AI

1. Traditional NLP vs Transformer Architectures
2. Emergence of Foundation Models
3. Open-Source vs Closed Models
4. Instruction-Tuned Models
5. Reasoning Models
6. Multimodal LLMs

Transformer Architecture Fundamentals

1. Attention Mechanisms
2. Self-Attention
3. Multi-Head Attention
4. Positional Encoding
5. Feed Forward Networks
6. Context Windows
7. KV Cache Concepts
8. Token Prediction Workflows

Tokenization and Embeddings

1. BPE Tokenization
2. SentencePiece
3. Embedding Spaces
4. Semantic Similarity
5. Vector Representations
6. Embedding Models
7. Chunking Strategies

LLM Inference Fundamentals

1. Temperature

2. Top-K Sampling
3. Top-P Sampling
4. Repetition Penalties
5. Beam Search
6. Streaming Inference
7. Speculative Decoding
8. Quantized Inference

Open-Source LLM Ecosystem

1. Llama Family
2. Mistral Models
3. DeepSeek Models
4. Gemma Models
5. Qwen Models
6. Phi Models
7. Model Selection Criteria
8. Hardware Considerations

2. Advanced Prompt Engineering

Prompt Engineering Fundamentals

1. Instruction Design
2. Prompt Structure
3. Context Injection
4. Output Formatting
5. System Prompts
6. Role-Based Prompting

Advanced Prompting Techniques

1. Zero-Shot Prompting
2. Few-Shot Prompting
3. Chain-of-Thought Prompting
4. Self-Consistency Prompting
5. Tree-of-Thought Prompting

6. ReAct Prompting
7. Step-Back Prompting
8. Deliberate Reasoning Patterns

Structured Output Engineering

1. JSON Output Constraints
2. Schema Enforcement
3. Function Calling
4. Tool Invocation Formats
5. Response Validation
6. Guardrails

Prompt Optimization

1. Prompt Compression
2. Context Window Optimization
3. Prompt Versioning
4. Prompt Evaluation
5. Prompt Benchmarking
6. Hallucination Reduction

Prompt Security

1. Prompt Injection Attacks
2. Jailbreak Techniques
3. Data Leakage Risks
4. Prompt Isolation
5. Output Sanitization
6. Safety Constraints

3. Retrieval-Augmented Generation (RAG)

RAG Architecture Fundamentals

1. Retrieval Pipelines
2. Embedding Generation

3. Semantic Search
4. Hybrid Retrieval
5. Re-ranking Pipelines

Vector Databases

1. Vector Storage Concepts
2. Similarity Metrics
3. Metadata Filtering
4. Indexing Strategies
5. ANN Search

Data Preparation for RAG

1. Document Parsing
2. Chunking Strategies
3. Metadata Enrichment
4. Embedding Optimization
5. Data Cleaning

Advanced RAG Architectures

1. Parent-Child Retrieval
2. Graph RAG
3. Agentic RAG
4. Contextual Compression
5. Multi-Vector Retrieval
6. Adaptive Retrieval

RAG Evaluation

1. Retrieval Accuracy
2. Context Precision
3. Faithfulness Evaluation
4. Hallucination Metrics
5. Benchmarking Frameworks

4. Fine-Tuning Fundamentals

Fine-Tuning Concepts

1. Pretraining vs Fine-Tuning
2. Instruction Tuning
3. Supervised Fine-Tuning
4. Domain Adaptation
5. Continual Learning

Fine-Tuning Workflows

1. Dataset Collection
2. Dataset Cleaning
3. Instruction Dataset Design
4. Prompt-Completion Formatting
5. Conversation Formatting
6. Dataset Validation

Hugging Face Ecosystem

1. Transformers Library
2. Datasets Library
3. PEFT Library
4. Accelerate
5. Safetensors
6. Model Hub Workflows

LoRA Fundamentals

1. Low-Rank Adaptation Concepts
2. Rank Decomposition
3. Adapter Layers
4. Trainable Parameters
5. Hyperparameter Selection
6. Merge Operations

QLoRA Fundamentals

1. Quantization Concepts
2. 4-Bit Quantization
3. NF4 Quantization
4. Double Quantization
5. Memory Optimization
6. Quantized Training Pipelines

Parameter-Efficient Fine-Tuning

1. LoRA
2. QLoRA
3. Prefix Tuning
4. Prompt Tuning
5. Adapter Tuning
6. IA3

Fine-Tuning Infrastructure

1. GPU Selection
2. CUDA Environments
3. VRAM Optimization
4. Mixed Precision Training
5. Gradient Checkpointing
6. Multi-GPU Training

Training Optimization

1. Batch Sizing
2. Gradient Accumulation
3. Learning Rate Scheduling
4. Optimizers
5. Loss Functions
6. Early Stopping

Evaluation and Benchmarking

1. Validation Pipelines
2. Perplexity
3. BLEU Metrics
4. ROUGE Metrics
5. Human Evaluation
6. Task-Based Evaluation

Model Packaging and Deployment

1. GGUF Conversion
2. Quantized Model Export
3. Ollama Integration
4. vLLM Deployment
5. Text Generation Inference
6. API Serving

Introduction to Agentic AI

Agentic AI Fundamentals

1. AI Agents vs Traditional Applications
2. Reactive vs Autonomous Agents
3. Goal-Oriented Architectures
4. Agent Lifecycles
5. Decision-Making Workflows

Agent Components

1. Planning Modules
2. Reasoning Modules
3. Tool Execution Layers
4. State Management
5. Memory Systems

Agent Design Patterns

1. Single-Agent Architectures
2. Multi-Agent Architectures
3. Planner-Executor Patterns
4. Supervisor Patterns
5. Hierarchical Agents
6. Collaborative Agents

Tool Calling Architectures

1. Function Calling
2. External API Invocation
3. Tool Registries
4. Tool Routing
5. Dynamic Tool Selection

Memory Architectures

1. Short-Term Memory
2. Long-Term Memory
3. Episodic Memory
4. Semantic Memory
5. Vectorized Memory Stores

Agent Frameworks

1. LangChain
2. LangGraph
3. CrewAI
4. AutoGen
5. Semantic Kernel
6. OpenAI Agent SDK Concepts

Building AI Agents with Python

Linux-Based AI Development Environment

1. Python Environment Management
2. CUDA Configuration
3. GPU Driver Management
4. Containerized AI Workflows
5. Dependency Isolation

Building Single AI Agents

1. Agent Initialization
2. Prompt Templates
3. Tool Registration
4. Action Routing
5. Response Parsing
6. State Persistence

Tool-Integrated Agents

1. REST API Integration
2. Database Connectivity
3. File System Operations
4. Shell Command Execution
5. Web Retrieval
6. Search Integration

Stateful Agent Architectures

1. Session Persistence
2. Conversation Tracking
3. Context Compression
4. Memory Retrieval
5. Agent Reflection

Planning and Reasoning Systems

1. Task Decomposition
2. Recursive Planning
3. Reflection Loops
4. Self-Correction
5. Dynamic Replanning

Agent Observability

1. Tracing
2. Logging
3. Token Monitoring
4. Cost Tracking
5. Execution Graphs
6. Failure Diagnostics

Multi-Agent Orchestration

Multi-Agent System Fundamentals

1. Agent Collaboration Models
2. Distributed Agent Architectures
3. Agent Specialization
4. Delegation Strategies
5. Swarm Coordination

Agent Communication Patterns

1. Synchronous Communication
2. Asynchronous Communication
3. Event-Driven Messaging
4. Pub/Sub Architectures
5. Request-Reply Patterns

Apache Kafka Fundamentals

1. Kafka Architecture
2. Brokers
3. Topics
4. Partitions
5. Consumer Groups
6. Replication

Kafka for Agentic AI

1. Event-Driven Agent Workflows
2. Agent Messaging Pipelines
3. Streaming AI Architectures
4. Distributed Task Coordination
5. Event Sourcing
6. Durable Agent Memory Streams

Kafka Integration with Python

1. Kafka Producers
2. Kafka Consumers
3. Async Processing
4. Serialization Formats
5. Avro and JSON Messaging
6. Error Handling

Multi-Agent Orchestration Frameworks

1. LangGraph Multi-Agent Workflows
2. CrewAI Orchestration
3. AutoGen Group Chats
4. Directed Agent Graphs
5. Dynamic Routing

Distributed AI Workflow Design

1. Event Choreography
2. Workflow Orchestration
3. Agent Dependency Graphs
4. Distributed Retries
5. Fault Tolerance
6. Idempotent Processing

Scalable Agent Infrastructure

1. Containerized Agents
2. Docker-Based Deployments
3. Kubernetes Concepts
4. Horizontal Scaling
5. GPU Scheduling
6. Distributed Inference

Security and Governance

1. Agent Permissions
2. Secret Management
3. API Security
4. Isolation Strategies
5. Governance Controls
6. Auditability

Production Considerations

1. Latency Optimization
2. Throughput Optimization
3. Cost Optimization
4. Monitoring and Alerting
5. Failure Recovery
6. Disaster Recovery

Disclaimer: This course outline is intended as a high-level training framework and guideline. The trainer reserves the right to amend, reorganize, expand, reduce, or substitute topics, tools, technologies, labs, and delivery approaches where necessary to accommodate participant skill levels, time constraints, platform updates, software version changes, or evolving industry practices without prior notice.

© 2026 Faysal Aziz · Kuala Lumpur