Course Title:

LLM deep Dive

🕒 **Duration:**

- **Total Duration:** 16 hours
- **Schedule:** 4 hours/day over 4 days

📚 **Course Objectives:**

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

- Design advanced fine-tuning strategies for LLMs (e.g., LoRA, QLoRA) and optimize model performance for specific use cases while mitigating overfitting and cost inefficiencies.
- Build scalable RAG (Retrieval-Augmented Generation) pipelines on cloud platforms, addressing challenges like chunking, retrieval accuracy, and integration with vector databases.
- Deploy LLMs efficiently in production using cloud-based solutions (e.g., vLLM, TGI) and evaluate trade-offs between latency, cost, and scalability.
- Evaluate LLM performance rigorously using benchmarks (HELM, AlpacaEval) and metrics (accuracy, toxicity, bias) to ensure alignment with business and ethical requirements.
- Implement governance and security best practices for LLMs, including prompt injection defenses, compliance with AI regulations (EU AI Act), and multimodal model risk assessment.

👥 **Target Audience:**

- ML Engineers & AI Researchers
- Data Scientists Transitioning to LLMOps
- AI/ML Technical Leads & Architects
- Cloud/MLOps Engineers
- AI Governance & Compliance Specialists

Level: advanced

Pre-requisites:

-

ML,DL; Working Knowledge of Foundation Models in LLM; Prompt Engineering and Experience in using tensorflow/ pytorch, sklearn and hugging face libraries.

📖 **Course Delivery Methodology:**

- **Hands-on Training:** 70% of the course will be practical, focusing on hands-on labs, demos, simulations, and project work.
- **Theory/Concepts:** 30% of the course will cover theoretical foundations to provide the necessary conceptual understanding.

📖 **Proposed Course Outline:**

Day 1: LLM Fundamentals & Fine-tuning Strategies (4 Hours)

Hour 1: Introduction to LLMs & Tokenization

- **Understanding LLMs:** Brief overview, common architectures (Transformers), and their capabilities.
- **Subword Tokenizers:**
  - Why subword tokenization? (Byte-Pair Encoding, WordPiece, SentencePiece).
  - Practical examples of tokenization with popular libraries (e.g., Hugging Face Tokenizers).
  - Impact of tokenization on model performance and cost.

Hour 2: Embedding Models & Their Applications

- **What are Embeddings?**
  - Word embeddings (Word2Vec, GloVe) vs. contextual embeddings (BERT, RoBERTa, Sentence-BERT).
  - How embeddings represent meaning.
- **Using Embedding Models:**
  - Generating embeddings for text.
  - Use cases: semantic search, text similarity, clustering.

Hour 3: Architecting Fine-Tuning Strategies (Part 1)

- **Introduction to Fine-tuning:**
    - Why fine-tune? (Domain adaptation, task-specific performance).
    - Supervised fine-tuning vs. instruction fine-tuning.
    - Chain of Thought, ReAct prompting techniques
- **Data Preparation for Fine-tuning:**
    - Curating and cleaning datasets.
    - Data formats and common challenges.
- **Parameter-Efficient Fine-Tuning (PEFT) Concepts:**
    - LoRA, QLoRA, Prompt Tuning (conceptual overview).

Hour 4: Architecting Fine-Tuning Strategies (Part 2) & Optimization

- **Implementing PEFT:**
    - Choosing the right PEFT method for different scenarios.
    - Hands-on considerations for applying PEFT (e.g., using `peft` library).
- **Optimizing Model Performance (Introduction):**
    - Basic concepts of model quantization (e.g., INT8, FP4).
    - Knowledge distillation (conceptual).

Day 2: RAG Pipelines & LLM Deployment (4 Hours)

Hour 1: Building Scalable RAG Pipelines (Part 1)

- **Understanding RAG:**
    - Retrieval-Augmented Generation explained.
    - Benefits of RAG over traditional fine-tuning for certain use cases.
- **Components of a RAG Pipeline:**
    - Document ingestion and chunking strategies.
    - Vector databases: Introduction and role in RAG (e.g., Pinecone, Weaviate, ChromaDB).

Hour 2: Building Scalable RAG Pipelines (Part 2)

- **Implementing Retrieval:**
    - Query embedding and similarity search.
    - Hybrid search techniques.
- **Generation with Retrieved Context:**
    - Prompt engineering for RAG.
    - Addressing hallucinations in RAG.
- **Building Scalable RAG Pipelines on Cloud (Conceptual):**
    - Overview of cloud services for RAG components (e.g., AWS Kendra, Azure AI Search, Google Cloud Vertex AI Search).

Hour 3: Large Language Model Deployment on Cloud (Part 1)

- **Deployment Considerations:**
  - Model size, inference speed, cost, scalability.
  - Batching and serving LLMs.
- **Cloud Deployment Options:**
  - Managed services (e.g., AWS SageMaker, Azure ML, Google Cloud Vertex AI Endpoints).
  - Containerization (Docker) for LLM deployment.

Hour 4: Large Language Model Deployment on Cloud (Part 2) & Different Modals

- **Advanced Deployment Topics:**
  - Load balancing and auto-scaling for LLMs.
  - Monitoring deployed LLMs.
- **Different Modalities of LLMs:**
  - **Multi-modal Models:** Introduction to models that handle text, image, audio, etc. (e.g., CLIP, DALL-E, Gemini, GPT-4V).
  - Use cases and challenges of multi-modal LLMs.

Day 3: LLM Evaluation, Governance & Advanced Topics (4 Hours)

Hour 1: LLM Performance Metrics

- **Traditional NLP Metrics for LLMs:**
  - BLEU, ROUGE (limitations for open-ended generation).
- **LLM-Specific Evaluation Metrics:**
  - Fluency, coherence, relevance, factual consistency.
  - Human evaluation vs. automated evaluation.
- **Benchmarking Best Practices:**
  - Common LLM benchmarks (e.g., MMLU, HELM).
  - Setting up internal benchmarks.

Hour 2: LLM Security & Governance (Part 1)

- **LLM Governance:**
  - Data privacy and compliance (GDPR, HIPAA implications).
  - Responsible AI principles for LLMs.
  - Model cards and documentation.
- **LLM Security:**
  - Prompt injection attacks.
  - Data leakage and privacy concerns during inference.

Hour 3: LLM Security & Governance (Part 2) & Reinforcement Learning for Alignment

- **Mitigation Strategies:**
  - Input/output filtering.
  - Guardrails and safety mechanisms.
  - Bias detection and mitigation in LLMs.
- **Reinforcement Learning for LLM Alignment:**
  - **RLHF (Reinforcement Learning from Human Feedback):**
    - Conceptual understanding of how RLHF aligns LLMs with human preferences.
    - Role of reward models.
  - **Direct Preference Optimization (DPO):** An alternative to RLHF.

Hour 4: Q&A, Case Studies & Next Steps

- **Recap of Key Concepts:**
  - Review of fine-tuning, RAG, deployment, evaluation, and governance.
- **Open Discussion & Q&A:**
  - Addressing specific questions from participants.
- **Real-world Case Studies (Brief):**
  - Examples of LLMs in production environments across different industries.
- **Future Trends & Resources:**
  - Brief discussion on emerging LLM research and tools.
  - Providing resources for continued learning.

Day 4: Introduction to Agentic AI, Agentic AI Applications in Retail

Hour 1: Introduction to Agentic AI & Retail Use Cases

- **What is Agentic AI?**
  Overview of Agentic AI principles: autonomy, goal-oriented behavior, contextual awareness.

- **Key Agent Properties:**
  Perception, memory, planning, action execution, interaction.

- **Agentic AI in Retail:**

    - Use cases: intelligent shopping assistants, personalized recommendations, inventory agents, dynamic pricing bots.
    - Real-world examples from Amazon, Instacart, Shopify.
- **Agent Architecture Overview:**
  Introduction to the Perceive-Plan-Act loop and decision trees in agents.

Hour 2: Agent Development Frameworks & Tooling

- **Agentic Development Frameworks:**
    - Overview of LangGraph (graph-based agent flow control)
    - Google ADK (Agent Development Kit): tooling for goal-driven agents
    - Optional mentions: Meta's AutoGen, CrewAI, Semantic Kernel
- **Toolchain Integration in Azure AI Foundry or local environments:**
    - LangChain, OpenAI Function Calling, ReAct agents
    - API integrations (e.g., product search, user profile DB, vector DB)
- **Design Patterns:**
    - Single-agent vs. Multi-agent vs. Orchestrator patterns
    - REST vs. Event-driven agents

Hour 3: Multi-Agent Systems & Communication Protocols

- **Understanding MAS (Multi-Agent Systems):**
  - Decentralized vs. centralized coordination
  - Task decomposition and distribution among agents
- **Agent-to-Agent Communication (A2A):**
  - Use of message-passing and asynchronous queues
  - Protocol design for collaboration
- **Multi-Agent Coordination Protocol (MCP):**
  - Define roles, intent signaling, negotiation
  - How to use LangGraph for MCP-style task routing
- **Use Case Examples:**
  - Retail assistant coordinating with a price-check agent and inventory validator agent

---

Hour 4: Reasoning, Planning & Lab: Build a Retail Shopping Assistant

- **Advanced Agent Features:**
  - Short-term and long-term memory integration (e.g., vector store, Redis)
  - Planning with ReAct, Tree of Thought, LangGraph DAG
  - Incorporating feedback loops and state transitions

- **Lab Objective:**
  🔧 **Build a Shopping Assistant Agent**
  - Input: User request like "Find me running shoes under ₹5000"
  - Agent Actions:
    - Query a dummy product catalog (CSV/JSON/public API)
    - Filter based on price, size, brand preferences
    - Rank and return options
    - Escalate to a secondary agent for inventory check
- **Tools:** LangChain or LangGraph, OpenAI/GPT API, dummy catalog JSON or public datasets
- **Bonus:** Use of memory and retriever in the agent for follow-up questions
-