Summer Training TR-103 Prompt Engineering

Day 10 Report

What is LangChain?

LangChain is an open-source framework that makes it easier to build AI applications using language models for multi-step and dynamic tasks. It combines prompts, chains, memory, and tools into structured workflows.

Key Strengths

• **Orchestration:** Connects LLMs, tools, and memory for complex workflows.

• **Modularity:** Independent design of components like prompts, chains, and tools.

• **Agent Support:** Real-time interaction and decision-making with APIs and tools.

Core Concepts

• **LLM:** The language model that generates text.

• **Chain:** A sequence of steps combining logic and prompts.

• **Tool:** An external function such as a calculator or search API.

• **Memory:** Maintains past interactions for multi-turn conversations.

• **Agent:** Chooses which tools to use and how to respond.

LangChain Setup

• Install LangChain via pip.

• Configure environment variables for API keys.

• Import modules and set up LLMs.

Chains in LangChain

Built simple chains to see how different prompts can be linked together to solve tasks. This included:

• Sequential chains (A → B → C).

• Summarization followed by Q&A.

• Query + calculation + summarization workflows.

Tools – Extending LLMs

LangChain lets LLMs use tools such as:

• Python REPL

• Web Search APIs

• Calculator

• Document Loaders

Agents

Agents can:

• Decide step-by-step actions.

• Choose appropriate tools.

• Manage long-form conversations.

Use Cases

• Conversational assistants.

• Automated data retrieval.

• Task-based agents like travel booking or summarization.

Building an Agent with Tools

We created a simple agent using:

• Gemini LLM

• Calculator and search tools

• Prompt templates

The agent was able to interpret queries, select tools, and return results.

Multi-Turn Conversation

• By using memory components, the agent could:

• Remember names.

• Continue previous topics.

• Maintain smooth conversational flow

Advanced Agent Patterns

We explored:

• ReAct (Reasoning + Action)

• Conversational ReAct

• Tool-Calling Patterns

What is Prompt Evaluation?

Prompt evaluation is the process of testing the quality, performance, and reliability of prompts used in LLM applications.

Main Prompt Evaluation Techniques

• Human Evaluation: Manually rating outputs.

• Automatic Evaluation: Using AI models or scoring functions.

• Metric-Based Evaluation: BLEU, ROUGE, BERTScore comparisons.

• A/B Testing: Comparing two prompt versions.

• Prompt Robustness Testing: Checking small changes in phrasing.

• Prompt Sensitivity Testing: Changing input structure/order.

• Response Diversity Testing: Looking at variability in responses.

• Cost and Speed Testing: Token usage and latency monitoring.

Code Implementation Using Google AI Studio API

• Python scripts created to test prompt variations.

• Logged outputs, token usage, and response times.

• Used A/B testing to compare prompts.

• Automated evaluations with scoring scripts and system prompts.