6. その他資料

6-1. 全体まとめ

- 生成 AI をうまく活用するためのプロンプトエンジニアリング・コンテキストエンジニアリング
- プログラミングできなくても活用できる
- 自分の業務との掛け合わせ=**一般的な LLM だけではできない**領域

6-2. クイズ

問 1

個別の指示設計に焦点を当てる「プロンプトエンジニアリング」に対し、AI との対話全体を体系的に設計するアプローチを何と呼びますか？

A. コンテキストエンジニアリング
B. モデルチューニング
C. エージェント設計
D. RAG (検索拡張生成)

問 2

AI が自身の学習データにない最新情報や専門知識にアクセスし、回答の正確性を向上させるための技術（フレームワーク）は何ですか？

A. ファインチューニング
B. RAG (Retrieval Augmented Generation)
C. Chain of Thought (思考の連鎖)
D. マルチモーダル AI

問 3

AI エージェントを構成する 3 つの不可欠な要素として、資料で挙げられていないものはどれですか？

A. モデル (LLM)
B. ツール
C. オーケストレーション層
D. ユーザーインターフェース

問 4

従来の「あなたはプロの〇〇です」という役割を与えるプロンプトの限界を乗り越えるため、AI に「思考のレンズ」を設計するアプローチとして紹介されているものはどれですか？

A. Few-Shot Learning
B. ReAct フレームワーク
C. コグニティブ・デザイン（認知設計）
D. コンテキスト圧縮

問 5

資料で紹介されている、プログラミング不要で AI アプリケーションやワークフローを構築できる「ノーコードツール」の例として、不適切なものはどれですか？

A. Dify
B. n8n
C. FlowiseAI
D. Python

問 6

Microsoft が提唱中で、プロンプトの設計と管理を体系化するためのマークアップ言語は何ですか？

A. XML
B. HTML
C. POML (Prompt Orchestration Markup Language)
D. LCEL (LangChain Expression Language)

問 7

自作で RAG・AI Agent を開発する場合と、クラウドサービスを利用する場合の比較において、一般的に自作開発が優れている点はどれですか？

A. 開発時間
B. 運用負荷
C. カスタマイズ性
D. 最新技術へのアクセス

問 8

AI にタスクを依頼する際、「ステップバイステップで考えてください」と一言加えるだけで、複雑な問題に対する回答の精度を向上させる効果が期待できるプロンプト技術は何ですか？

A. Few-Shot Learning

B. Zero-shot Chain-of-Thought

C. ReAct

D. Role Playing

問 9

複数の AI エージェントにそれぞれ専門的な役割を割り当て、分業と協調作業によって複雑なタスクを遂行する AI エージェントのデザインパターンは何ですか？

A. Debate-based Cooperation（討論ベースの協調）
B. Cross-Reflection（相互省察）
C. Role-based Cooperation（役割ベースの協調）
D. Human Reflection（人間による省察）

問 10

コマンドライン（ターミナル）から直接 AI を利用し、スクリプト化やファイル操作の自動化を容易にするツールとして紹介されているものはどれですか？

A. LangChain

B. Cursor

C. Dify

D. GeminiCli / Claude Code CLI

解答

1. A. コンテキストエンジニアリング
2. B. RAG (Retrieval Augmented Generation)
3. D. ユーザーインターフェース
4. C. コグニティブ・デザイン（認知設計）
5. D. Python
6. C. POML (Prompt Orchestration Markup Language)
7. C. カスタマイズ性
8. B. Zero-shot Chain-of-Thought
9. C. Role-based Cooperation（役割ベースの協調）
10. D. GeminiCli / Claude Code CLI

6-3. 参考文献

論文・学術情報 (arXiv)

[1] ReAct: Synergizing Reasoning and Acting in Language Models
https://arxiv.org/abs/2210.03629

[2] The Prompt Report: A Systematic Survey of Prompt Engineering Techniques
https://arxiv.org/abs/2406.06608

[3] Agent Design Pattern (論文)
https://arxiv.org/html/2405.10467v4

[4] Constitutional AI: Harmlessness from AI Feedback
https://arxiv.org/abs/2212.08073

[5] Structure Guided Prompt: Instructing Large Language Model in Multi-Step Reasoning by Exploring Graph Structure of the Text
https://arxiv.org/abs/2402.13415

[6] When "A Helpful Assistant" Is Not Really Helpful: Personas in System Prompts Do Not Improve Performances of Large Language Models
https://arxiv.org/abs/2311.10054

技術解説・ブログ記事

[7] 生成 AI の AI エージェントを大手 3 社（AWS、Azure、Google Cloud）で徹底比較してみた
https://blog.g-gen.co.jp/entry/comparing-agent-architecture-across-cloud-vendors

[8] 生成 AI の RAG 構成を大手 3 社で徹底比較
https://blog.g-gen.co.jp/entry/comparing-rag-architecture-across-cloud-vendors

[9] 生成 AI のよくある誤解を整理して AI の業務活用を推進する
https://blog.g-gen.co.jp/entry/clearing-up-misconceptions-about-generative-ai

[10] 【プロンプト技術 58 種類の 9 割は「無駄」だった...】実例 2,000 件から学ぶ！LLM 時代の"強いプロンプトテンプレート"設計術 (GMO インターネットグループ)
https://recruit.gmo.jp/engineer/jisedai/blog/prompt_template/

[11] 「あなたはプロの〇〇です」をもうやめたい (Qiita)
https://qiita.com/makotosaekit/items/0eccb562bf7d3f66fbfa

[12] Context Engineering for Agents (LangChain Blog)
https://blog.langchain.com/context-engineering-for-agents/

[13] Context Engineering Guide (Prompting Guide)
https://www.promptingguide.ai/guides/context-engineering-guide

[14] Context Engineering with LLMs (Phil Schmid's Blog)
https://www.philschmid.de/context-engineering

[15] LLM へのプロンプトを構造化された文書で管理する POML (azukiazusa.dev)
https://azukiazusa.dev/blog/poml-prompt-structured-document/

[16] AWS Bedrock の Knowledge base で RAG アプリを実装してみた (NRI ネットコム)
https://tech.nri-net.com/entry/aws_bedrock_rag_app

[17] Azure OpenAI Service の「On Your Data」とは？ (NTT 東日本)
https://business.ntt-east.co.jp/content/cloudsolution/column-628.html

[18] MCP(Model Context Protocol)の紹介 (CloudNative Inc. Blog)
https://blog.cloudnative.co.jp/27994/

[19] Agent Communication Protocol について (Qiita)
https://qiita.com/okikusan-public/items/196f1a781b3bf33536aa

[20] Agent Design Pattern の解説 (Qiita)
https://qiita.com/Chi_corp_123/items/cf26215878cad285599b

各種ガイド・資料

[21] Vellum AI LLM Leaderboard
https://www.vellum.ai/llm-leaderboard

[22] 特許用プロンプト例 (Google Sheets)
https://docs.google.com/spreadsheets/d/1bh1X3BGT8x99OSos19TrRpOUQGN9kqhyKK2v-r3tcWI/edit?usp=sharing

[23] GPT-5 prompting guide (OpenAI Cookbook)
https://cookbook.openai.com/examples/gpt-5/gpt-5_prompting_guide

[24] Claude 4 プロンプトエンジニアリングのベストプラクティス (Anthropic)
https://docs.anthropic.com/ja/docs/build-with-claude/prompt-engineering/claude-4-best-practices

[25] Google prompting guide 101 (Google)
https://services.google.com/fh/files/misc/gemini-for-google-workspace-prompting-

[26] RAG From Scratch (GitHub)
https://github.com/langchain-ai/rag-from-scratch

[27] Google Cloud Architecture Center - Generative AI for RAG
https://cloud.google.com/architecture/ai-ml/generative-ai-rag?hl=ja

[28] Google Codelabs - Building AI Agents with Vertex AI
https://codelabs.developers.google.com/devsite/codelabs/building-ai-agents-vertexai?hl=ja#2

[29] patentsearchagent (GitHub)
https://github.com/niship2/patentsearchagent

[30] POML (Prompt Orchestration Markup Language) (GitHub)
https://github.com/microsoft/poml

[31] Playwright MCP を使って特許調査
https://www.enlighton.co.jp/post/playwright-mcpを使って特許調査

[32] FlowHunt USPTO Patent MCP
https://www.flowhunt.io/integrations/uspto/

[33] Google Patents MCP Server
https://github.com/KunihiroS/google-patents-mcp

[34] プロンプトジェネレータツール比較
https://miralab.co.jp/media/prompt_generator_recommendations/

[35] 知財バージョン/私作成
https://ip-prompt-generator-for-llms-222593271342.us-west1.run.app/

[36] プロンプト研究の論文紹介
https://note.com/rami_engineer/n/n8ecd7ac00cc4

[37] Baby AGI: Task-driven Autonomous Agent
https://yoheinakajima.com/task-driven-autonomous-agent-utilizing-gpt-4-pinecone-and-langchain-for-diverse-applications/

[38] 現場で活用するための AI エージェント実践入門
https://www.kspub.co.jp/book/detail/5401408.html

[39] Deep Agents (LangChain Blog)
https://blog.langchain.com/deep-agents/

[40] Deep Agents with LangGraph (無料コース)
https://academy.langchain.com/courses/deep-agents-with-langgraph

[41] Deep Agents from Scratch (GitHub)
https://github.com/langchain-ai/deep-agents-from-scratch/blob/main/notebooks/4_full_agent.ipynb

[42] X 投稿: RAG システムの工夫点
https://x.com/athleticKoder/status/1968658987756224919

[43] GitHub: marp
https://github.com/nishiq2/marp

[44] ChatGPT
https://chat.openai.com

[45] Claude
https://claude.ai

[46] Gemini
https://gemini.google.com

[47] Google AI Studio
https://aistudio.google.com/

[48] AI0W - AI Agents
https://ai0w.com/agents/

[49] Dify
https://dify.ai/

[50] n8n
https://n8n.io/

[51] FlowiseAI
https://flowiseai.com/

[52] LangFlow
https://www.langflow.org/

[53]Effective context engineering for AI agents
https://www.anthropic.com/engineering/effective-context-engineering-for-ai-agents