# **知識記憶管理設計**

以下に、**RAG（Retrieval-Augmented Generation）による知識記憶管理設計**について、目的に応じた構成案と改善点を整理します。

## **1. 課題の要約**

| **項目** | **内容** |
| --- | --- |
| 目的 | LLMの通常の会話文脈保持に加え、**ユーザの発言から得た知識を外部DBにベクター形式で保持・再利用**したい |
| 要件 | キーワード抽出 → ベクター変換 → 類似検索 → RAG的補助文脈としてLLMに投入 |
| 応用 | パーソナライズ学習、記憶的対話、長期記憶の代替ストレージ |

## **2. 技術的観点：RAGによる記憶管理設計構成**

### **🔁 標準構成の流れ**

発言

↓

[1] 意味解析（NLP/NER/感情分析）

↓

[2] キーワード or 埋め込みベクター抽出（LLMまたはEmbedding Model）

↓

[3] ベクターストアに保存（pgvector, Weaviate, Qdrant, FAISSなど）

↓

[4] 類似発言・知識を再検索（類似記憶・概念補完）

↓

[5] LLMへ再プロンプト（retrieved contextとともに）

↓

[6] 応答生成

## **3. 改善・最適化設計ポイント**

| **項目** | **改善点／代替案** | **補足** |
| --- | --- | --- |
| **ベクター化** | 単語レベルより文脈レベル（SentenceBERT / OpenAI Embedding）推奨 | キーワード抽出より意味保持が重要 |
| **分類構造** | 「発言の意図」や「発言の感情状態」を別スロットに分離 | 類似文検索に加え「状態での補完」可能 |
| **記憶の時間タグ管理** | timestamp や conversation\_id を保存 | 時系列推定や「直近記憶」制御に有効 |
| **記憶の分類（短期/長期）** | 類似度やアクセス頻度により昇格（LRUまたは閾値） | メモリ管理戦略として有効 |
| **メタ情報の記憶** | 「誰の発言か」「反応はあったか」「感情状態」など保存 | LLMへのプロンプト補完に重要 |
| **知識圧縮（スパース化）** | テキストクラスタリング＋要約 | 定期バッチ処理で記憶容量を抑制 |

## **4. 推奨アーキテクチャ（構成図）**

┌──────────────┐

│ ユーザ発言 │

└────┬─────┘

│

▼

┌──────────────┐

│ 文解析（NER/感情）│

└────┬─────┘

▼

┌──────────────┐

│ ベクター変換（Embedding） │

└────┬─────┘

▼

┌──────────────┐

│ ベクターストア（pgvector等）│

└────┬─────┬─────┐

│ │ │

▼ ▼ ▼

[近似検索] [時間フィルタ] [メタ補助]

│ │ │

└─────┴─────┘

▼

┌─────────────────────┐

│ LLM（RAGによる文脈補完）│

└─────────────────────┘

## **5. 使用技術例**

| **機能** | **ツール／技術** |
| --- | --- |
| Embedding生成 | OpenAI Embeddings / HuggingFace models / Sentence-BERT |
| ベクターDB | pgvector / Weaviate / Qdrant / FAISS |
| 類似検索 | cosine / Euclidean / inner product |
| データ形式 | PostgreSQL (with pgvector) / JSON schema / Redis（短期記憶） |

## **6. 設計・構築時の留意点**

* 類似検索による知識抽出には**質と構造が重要**
* プロンプト汚染（irrelevant memory）の回避には**閾値設定**と**分類管理**が必要
* 再学習（reinforcement）ではなく**補助コンテキスト**として扱うのが原則（RAG）

## **7. 参考資料一覧（URL付き）**

* OpenAI Embedding Docs:<https://platform.openai.com/docs/guides/embeddings>
* pgvector:<https://github.com/pgvector/pgvector>
* Qdrant:<https://qdrant.tech>
* Weaviate:<https://weaviate.io>
* LangChain Memory機能:<https://docs.langchain.com/docs/modules/memory>

## **8. 備考**

本出力はAIによるものであり、最終判断は人間にて行ってください。

設計に応じて、ベクターDBスキーマ、検索クエリ、プロンプト構成例などの具体実装も対応可能です。必要であれば続けて指示ください。