

warning "Mise à jour du backend (octobre 2026)" L'inférence GitHub Models a été retirée le 30 juillet 2026. Les références ci-dessous sont conservées uniquement comme contexte historique pendant la migration de ce lab. Utilisez le Lab 013 ? Backends LLM gratuits pour configurer Foundry Local ou Ollama avant de continuer.

Lab 026 : Pattern RAG agentique

Niveau : L200 Parcours : RAG Durée : ~40 min ? Coût : GitHub Free + Docker (local)

Ce que vous apprendrez

Pourquoi le RAG naïf échoue sur les questions complexes Réécriture de requêtes ? améliorer la récupération avant la recherche Hypothetical Document Embeddings (HyDE) ? générer pour récupérer RAG multi-hop ? récupération itérative pour les questions en plusieurs parties

Auto-réflexion ? l'agent évalue la qualité de sa propre réponse

Introduction

RAG naïf : vectoriser la requête ? chercher ? générer. Fonctionne bien pour les questions simples. Échoue pour :

Questions vagues : *« Parle-moi de ça »* (qu'est-ce que « ça » ?) Multi-hop : *« Qu'est-ce qui est moins cher ? le matériel de camping ou d'escalade ? Combien j'économiserais ? »*

Lacunes de connaissances : *« Quel est le produit le plus récent ? »* (peut nécessiter de connaître la date actuelle) Hallucination : le modèle invente des faits absents du contexte

Le RAG agentique ajoute des boucles de raisonnement autour de la récupération. L'agent *décide* comment chercher, *évalue* les résultats, et *réessaie* si nécessaire.

Prérequis

Avoir terminé le Lab 022 (pgvector en fonctionnement + documents ingérés) GITHUB_TOKEN configuré Conteneur pgvector en fonctionnement : docker start pgvector-rag

tip "Données d'exemple déjà chargées ?" Si vous avez exécuté l'étape d'ingestion du Lab 022 avec le jeu de données, vous avez déjà 42 documents dans pgvector prêts pour ce lab. Sinon, exécutez d'abord l'étape 3 du Lab 022.

Exercice du lab

Étape 1 : Réécriture de requêtes

Avant de chercher, demandez au LLM de réécrire la question de l'utilisateur en meilleures requêtes de recherche.

[diagram or code sample]

Étape 2 : Récupérer avec plusieurs requêtes et dédupliquer

[diagram or code sample]

Étape 3 : HyDE ? Hypothetical Document Embeddings

Au lieu de vectoriser la question, générez une *réponse hypothétique* et vectorisez-la.

Cela correspond souvent mieux au vrai document.

[diagram or code sample]

Étape 4 : RAG multi-hop

Pour les questions complexes, récupérer ? générer une réponse partielle ? récupérer à nouveau.

[diagram or code sample]

Étape 5 : Auto-réflexion (vérification de la qualité de la réponse)

[diagram or code sample]

Résumé de l'architecture RAG agentique

[diagram or code sample]

Prochaines étapes

Mémoire d'agent entre sessions : ? Lab 027 ? Patterns de mémoire d'agent Évaluer la qualité du RAG à grande échelle : ? Lab 035 ? Évaluation d'agent

Kit d'accompagnement

Le build du site génère des téléchargements PDF et ZIP stables. Le ZIP contient le guide du lab et tous les fichiers d'accompagnement déclarés.

[Télécharger le PDF généré](#) [Télécharger le ZIP généré](#)

Supporting file: CHEATSHEET.md

Fichiers d'accompagnement: Lab 026 : Pattern RAG agentique

Commencez par le parcours local ou gratuit documenté. Consignez les hypothèses et les preuves de validation. Gardez les services hébergés facultatifs sauf exigence explicite du lab.

Supporting file: README.md

Lab 026 : Pattern RAG agentique

Utilisez ces fichiers avec le guide du lab et le kit généré.