Comment fonctionne un RAG ?

Données traitées	Outils	Public cible
Texte	VectorStore LLM	Initiateur de projet

Budget	Gain	Effort
€€€	★★★	

Prérequis	Connaissance des modèles de langage (LLM), Similarité sémantique, *chunking* des documents, Connaissance basique des bases de données

Contributeur
Kajetan WOJTACKI
Ingénieur de Recherche Senior chez DecisionBrain, spécialisé en IA et apprentissage automatique. Possède un master en Physique Appliquée et un doctorat en Mécanique et Génie Civil.

PROBLÉMATIQUE

Les modèles de langage comme les LLM sont performants pour générer des réponses, mais ils peuvent fournir des informations obsolètes ou imprécises, car ils sont limités aux données sur lesquelles ils ont été entraînés. Le ***Retrieval-Augmented Generation*** **(RAG)** résout ce problème en enrichissant le modèle avec des informations externes provenant de bases de données, de documents fournis par l'utilisateur ou de moteurs de recherche. Il permet au modèle de rester **à jour**.

EXPLICATION

Les systèmes de RAG combinent les **capacités des modèles de langage** (LLM) avec des **mécanismes de recherche** d'informations en temps réel pour produire des réponses fiables et précises. L'objectif majeur est de **réduire les hallucinations** (réponses incorrectes ou inventées) en s'appuyant sur des données externes à jour. Cette approche garantit une **précision** accrue, particulièrement dans des **contextes spécifiques** à un domaine. Les entreprises peuvent ainsi intégrer des systèmes capables de fournir des **réponses adaptées**, tels qu'un support client ou un résumé de documents volumineux.

Source : ©Just_Super, Getty Images Signature via Canva.com

HUB FRANCE IA

Intelligence Artificielle Générative
Fiche pratique
Catégorie : Acculturation technique

Comment fonctionne un RAG ?

MISE EN ŒUVRE

Préparation des documents et de la base de données
Les documents sont collectés, nettoyés et préparés pour être indexés.

- Découpage des documents (*chunking*) : Les documents longs sont divisés en morceaux (*chunks*) pour améliorer la recherche et la pertinence des résultats.
- Indexation des données : Les *chunks* sont convertis en vecteurs à l'aide de modèles appelés modèles d'*embedding*, puis stockés dans une base de données (*vectorstore*)

Input utilisateur et sa vectorisation
La requête utilisateur est également convertie en vecteur via ce même processus d'*embedding* model et grâce à des techniques de similarité sémantique pour permettre une comparaison efficace avec les *chunks.*

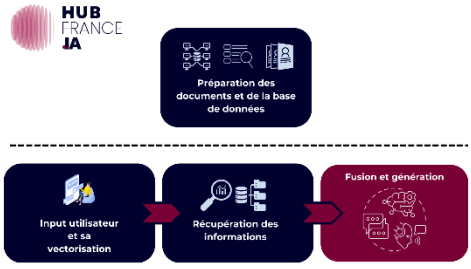
Récupération des informations
Le système interroge la base vectorielle pour récupérer les documents les plus pertinents (les k premiers où on définit k en fonction de ses besoins), en priorisant les informations sémantiquement proches pour contourner les limites de longueur de contexte des LLM.

Fusion et génération
Les documents récupérés sont injectés dans le modèle LLM en tant que contexte supplémentaire. Cela permet au modèle de générer une réponse contextuelle précise sans nécessiter de fine-tuning (c'est-à-dire un coûteux réentraînement complet du modèle).

Production de réponse
La réponse finale est produite par le modèle en combinant la requête utilisateur et les informations récupérées.

Exemple pratique

Une entreprise utilise un système RAG pour son support client.
Lorsqu'un utilisateur pose une question, le RAG récupère les informations pertinentes dans une base documentaire interne, comme des FAQ ou des manuels techniques.
Ensuite, ces informations sont fusionnées avec la requête pour générer une réponse précise.



POUR ALLER PLUS LOIN

Optimisation des requêtes : Améliorer les résultats en optimisant les algorithmes de recherche vectorielle
Graph Rag : Extension permettant une récupération hiérarchique ou relationnelle des informations.
Reranking : Étape de réorganisation des chunks pour prioriser les informations par pertinence.