



KgColPali : EFFICIENT DOCUMENT RETRIEVAL WITH VISION LANGUAGE MODELS AND KNOWLEDGE GRAPHS

Auteur

Marcos LAHOZ
Université Paris-Saclay
marcos.lahoz@universite-paris-saclay.fr

Encadrante

Céline HUDELLOT
Laboratoire MICS
celine.hudelot@centralesupelec.fr

Année universitaire 2024-2025

PLAN DU DOCUMENT

Ce rapport est structuré de manière à offrir une vue d'ensemble complète du projet de stage, depuis son contexte initial jusqu'à l'analyse des résultats obtenus. Il est organisé en onze sections principales, détaillées ci-dessous :

- 1. Contexte du stage :** Cette section introductive présente le laboratoire d'accueil, son environnement de recherche et le cadre général dans lequel s'inscrit le stage, offrant une base pour comprendre les objectifs du projet.
- 2. Contexte Général de l'Étude :** Cette partie établit le cadre théorique en introduisant les grands modèles de langage (LLMs), leurs limitations (connaissance statique, hallucination) et la justification de l'adoption des systèmes RAG par rapport au fine-tuning.
- 3. Résumé des Contributions du Stagiaire :** Ce chapitre offre un aperçu concis des contributions personnelles, couvrant l'étude théorique, le développement et l'évaluation du modèle RAG développé.
- 4. Description Détaillée du Problème :** Cette section définit les défis spécifiques du projet, notamment la gestion de documents multimodaux et l'enrichissement sémantique pour des applications médicales.
- 5. Analyse de la Littérature et Synthèse de la Bibliographie :** Une revue de l'état de l'art est proposée ici, synthétisant les avancées sur les systèmes RAG, les modèles de langage-vision (comme ColPali), les Knowledge Graphs et les benchmarks comme MIRAGE.
- 6. Développements Théoriques et Méthodologiques :** Cette partie détaille la conception du système RAG hybride, expliquant l'architecture, les rôles des composants (retriever ColPali, KG MONDO) et la méthodologie d'évaluation adoptée.
- 7. Réalisation Logicielle :** Un aperçu technique est fourni, décrivant les modules logiciels, les technologies utilisées (Python, serveur Ruche avec GPUs) et l'environnement de développement mis en œuvre.
- 8. Résultats et Discussion :** Cette section présente les résultats de l'évaluation sur le benchmark MIRAGE, comparant les performances du modèle de base et du modèle enrichi par le Knowledge Graph, suivie d'une discussion sur les conclusions et limites.
- 9. Remerciements :** Une section dédiée exprime ma gratitude envers les collaborateurs, encadrants et institutions qui ont soutenu ce projet.
- 10. Glossaire :** Un glossaire est inclus pour clarifier les termes techniques et acronymes (ex. : RAG, LLM, KG) utilisés dans le rapport.
- 11. Bibliographie :** Une liste exhaustive des références, incluant articles scientifiques et ressources en ligne, complète le document.

I. CONTEXTE DE MON STAGE

Mon encadrante est Céline Hudelot qui est la directrice du Laboratoire MICS. Son domaine de recherche est l'intelligence artificielle. Plus spécifiquement, elle travaille sur l'interprétation de données non structurées (texte, image, documents) à l'aide d'approche d'apprentissage automatique et d'approches de représentation de la connaissance et du raisonnement. Il s'agit d'un stage de recherche de quatre mois dans lequel j'ai travaillé sur les modèles RAG (Retrieval Augmented Generation). Concrètement, j'ai choisi faire mon stage sur l'application des Knowledges Graphs dans les modèles RAG pour améliorer leur performance.

Spécifiquement, j'ai accordé avec mon encadrante que l'objectif était de découvrir un environnement recherche (et donc de repartir d'un travail existant, i.e. Colpali (4)), dans un contexte d'application stratégique du laboratoire (domaine médical).

Je voudrais ajouter que pendant mon stage j'ai participé dans PFIA 14 à Dijon avec mon encadrante et j'ai pu écouter plusieurs conférences. Particulièrement, j'ai écouté des conférences sur les modèles RAG, et d'autres sur les Knowledges Graphs qui m'ont aidé à bien comprendre ces sujets.

A. *Laboratoire MICS*

Le **Laboratoire MICS** (Mathématiques et Informatique pour la Complexité et les Systèmes), rattaché à **Centrale-Supélec**, est un centre de recherche dédié à l'étude des systèmes complexes. Fondé au début des années 2000 sous le nom de MAS (Mathématiques Appliquées et Systèmes).

Le laboratoire MICS a une grande expertise en paradigmes d'apprentissage automatique à partir de données non structurées et complexes dans un cadre réaliste avec notamment la proposition de nouveaux schémas d'apprentissage capables de garder un bon niveau de performance malgré peu ou pas de données étiquetées. Ses thématiques de recherche portent sur la modélisation, la simulation, l'analyse et l'optimisation de données et de systèmes complexes, qu'elles ou qu'ils proviennent du monde industriel, du vivant, des marchés financiers, de l'information ou des réseaux.

Leurs travaux couvrent un champ large du problème : ils ont proposé des paradigmes d'apprentissage avec différents degrés de supervision (non-supervisé [Ouali et al, 2020a], semi-supervisé [Ouali et al, 2020b], auto-supervisé [Ouali et al., 2021], hybridé avec de la connaissance [Ledaguenel et al., 2024], par transfert [Tamaazousti et al., 2020], incremental [Petit et al, 2023], avec peu d'exemples [Boudiaf et al., 2023]), sur différentes tâches et différentes modalités [Pellegrain et al, 2022]. Une spécificité de leurs travaux est de se placer dans des scénarios réalistes dans lesquels les données et leur environnement sont imparfaits et dynamiques [Bennequin et al, 2021].

Le laboratoire a aussi une grande expertise en IA générative, notamment sur les grands modèles de langue avec des travaux portant sur des problématiques scientifiques liées à leur déploiement (stratégie d'apprentissage et d'adaptation) [Faysse et al, 2023], sur leur évaluation robuste [Colombo et al, 2024a] et sur leur robustesse [Gisserot-Boukhlef, 2024].

Le laboratoire a aussi été à l'initiative ou a participé à la construction et à la mise à disposition de grands modèles de langues, notamment CroissantLLM [Faysse et al., 2024], un modèle bilingue (français et anglais) souverain, TowerLLM (pour la traduction) [Guerreiro et al, 2024b] et SaulLM [Colombo et al, 2024b] (dans le domaine légal). Le MICS fait aussi partie de l'équipe à l'origine de RayDino [Moutakanni et al., 2024]. Récemment, le MICS est aussi à l'initiative d'EUROBERT [Boizard et al, 2025], une suite de modèles de type encodeurs, souverains, open source et délivrant de très bonnes performances en représentation textuelle sur les langues européennes ainsi que pour les tâches liées aux mathématiques et au code.

L'équipe impliquée dans ce projet est l'équipe GEMILA (Groupe d'Études en Mathématiques et Informatique pour l'Apprentissage) dont les travaux visent à développer des méthodologies à l'interface de l'informatique et des mathématiques pour l'apprentissage statistique dans des cadres applicatifs à fort impact sociétal.

TABLE DES MATIÈRES

I. Contexte de mon stage	3	1. Phase d'Indexation et de Préparation du Corpus.	13
A. Laboratoire MICS	3	2. Phase de Récupération Multimodale et d'Enrichissement.	14
Table des matières	4	3. Phase de Génération	14
II. Contexte Général de l'Étude	5	VI Réalisation Logicielle	14
A. RAG versus Fine-Tuning : Deux Approches pour une même Finalité	5	A. Technologies et Outils Utilisés	14
B. Fonctionnement d'un LLM	5	B. Architecture Logicielle	15
C. Le Fonctionnement du Pipeline RAG	5	C. Étapes de Développement	15
D. Importance des Retrievers Multimodaux comme ColPali	6	D. Défis Techniques	15
E. Les Graphes de Connaissance dans le Contexte du RAG	7	E. Documentation et Reproductibilité	15
F. Problème Posé	7	VII Résultats	16
III Résumé des Contributions du Stagiaire	7	A. Protocole Expérimental	16
A. Analyse Théorique	7	B. Analyse des Performances par Configuration	17
B. Conception et Développement d'un Modèle RAG Multimodal	7	1. Performances de la Configuration COT (LLM Seule)	17
1. Schéma du Fonctionnement du Modèle RAG	7	2. Performances des Configurations KG Context	18
C. Intégration et Exploitation d'un Knowledge Graph	8	3. Performances des Configurations RAG	18
D. Évaluation avec le Benchmark MIRAGE	8	4. Performances de RAG + KG Retrieve	18
IV Description Détaillée du Problème	8	C. Comparaison avec les Résultats de Référence de MIRAGE	18
A. Le Défi de la Multimodalité et de la Structure Documentaire	8	D. Discussion	19
B. L'Enrichissement Sémantique par les Connaissances Structurées	9	IX Remerciements	21
V. Analyse de la Littérature et Synthèse de la Bibliographie	9	X. Liste des figures	21
A. Cadre Général des Systèmes RAG	9	XI Glossaire	22
1. L'Évolution des Architectures RAG	9	XII Bibliographie	23
B. Frameworks RAG Spécialisés et Pertinence Clinique	10	XIII Annexes	24
C. Récupération d'Information Multimodale	11	A. Templates Utilisés dans le Modèle	24
1. Les Défis des Documents Multimodaux en Médecine	11	B. Exemple du contexte généré par le KG	24
2. Les Modèles Vision-Langage (VLM) et le Modèle ColPali	11		
D. Évaluation Standardisée avec le Benchmark MIRAGE	11		
E. L'Apport des Knowledge Graphs	12		
1. L'Intégration de Connaissances Structurées dans les Systèmes RAG	12		
2. Les Knowledge Graphs dans le Domaine Médical	12		
3. Interprétabilité et Responsabilité	12		
VI Développements Théoriques et Méthodologiques	13		
A. Architecture Globale du Système RAG Hybride	13		

II. CONTEXTE GÉNÉRAL DE L'ÉTUDE

L'apparition des grands modèles de langage (LLMs) a été un événement très important dans l'histoire de l'intelligence artificielle, ouvrant des possibilités inédites dans la génération et la compréhension du langage naturel. Des modèles comme GPT-4 ont démontré une capacité surprenante à rédiger des textes cohérents, à résumer des documents complexes et à répondre à des questions sur un grand ensemble de sujets. Cependant, leur utilité pratique est souvent limitée par deux faiblesses inhérentes : leur **base de connaissances statique** et leur tendance à l'**hallucination**. Un LLM, bien qu'entraîné sur d'immenses quantités de données, possède une connaissance limitée à la période de son entraînement, le rendant incapable d'intégrer des informations récentes ou de corriger des faits obsolètes sans intervention externe. De plus, lorsqu'il est confronté à des questions pour lesquelles il manque d'informations précises, il peut générer des réponses plausibles mais incorrectes. Dans un contexte où la précision est non négociable, comme dans le domaine médical, ces limitations sont inacceptables. Le risque d'une réponse inexacte dans un environnement clinique peut avoir des conséquences graves, ce qui rend l'intégration de sources d'information fiables et à jour une nécessité absolue.

A. RAG versus Fine-Tuning : Deux Approches pour une même Finalité

Historiquement, l'une des méthodes principales pour adapter un LLM à un domaine spécifique est le *fine-tuning*. Cette approche consiste à continuer l'entraînement du modèle sur un corpus de données pertinent pour le domaine cible (par exemple, des textes médicaux). Le *fine-tuning* modifie les poids du modèle lui-même, lui permettant d'internaliser de nouvelles connaissances et d'adapter son style de réponse. Cependant, cette méthode présente des inconvénients majeurs qui en limitent l'applicabilité dans de nombreux scénarios :

- **Coût et Complexité** : Le *fine-tuning* est un processus coûteux en ressources informatiques (GPU) et en temps, nécessitant un jeu de données d'entraînement de grande taille et de haute qualité.
- **Stabilité de la Connaissance** : La connaissance acquise est statique, comme pour le modèle initial. Toute mise à jour de l'information nécessite de répéter le processus de *fine-tuning*, ce qui n'est pas viable pour des domaines dynamiques comme la médecine.
- **Transparence** : Les faits appris sont encapsulés dans les poids du modèle, ce qui rend difficile, voire impossible, de tracer la source d'une information spécifique ou de corriger une erreur.

C'est dans ce contexte que le paradigme de la Retrieval-Augmented Generation (RAG) a été conceptualisé comme une solution élégante. Le RAG propose une approche fondamentalement différente et plus flexible. Au lieu de mo-

difier le modèle lui-même, il introduit un mécanisme externe qui permet au LLM de rechercher des informations pertinentes et à jour dans une base de documents (*corpus*) avant de générer une réponse. Le RAG ne change pas le modèle, mais enrichit son contexte.

B. Fonctionnement d'un LLM

Un modèle de langage large (LLM) est une architecture d'intelligence artificielle basée sur des réseaux de neurones, principalement des transformers, conçus pour comprendre et générer du texte de manière semblable à un humain. Ces modèles sont entraînés sur de vastes corpus de données textuelles, souvent issus de sources diverses comme des articles scientifiques, des livres et des sites web, afin d'apprendre les structures linguistiques, les relations sémantiques et les contextes. Le fonctionnement repose sur deux phases principales : le pre-training, où le modèle acquiert une connaissance générale à travers une tâche d'auto-régulation (par exemple, prédire le mot suivant dans une phrase), et le fine-tuning, où il est adapté à des tâches spécifiques avec des données annotées, comme la génération de réponses médicales.

Au cœur du processus, les transformers utilisent un mécanisme d'attention pour pondérer l'importance des mots dans une séquence, permettant au modèle de capturer des dépendances à longue distance et de contextualiser les entrées. Par exemple, un LLM comme Gemini 2.5 Flash Lite, optimisé pour une exécution rapide, applique ces principes en combinant une architecture légère avec des techniques d'optimisation, tout en conservant une capacité à traiter des requêtes complexes. Cependant, les LLM présentent des limites inhérentes, telles qu'une dépendance aux données d'entraînement (souvent statiques) et un risque d'hallucinations, ce qui nécessite des approches comme Retrieval-Augmented Generation (RAG) pour intégrer des informations externes et améliorer la fiabilité, notamment en contexte clinique.

C. Le Fonctionnement du Pipeline RAG

Un pipeline RAG typique repose sur trois étapes principales, qui le différencient fondamentalement de l'approche par *fine-tuning*, comme illustré dans la Figure I. Ce processus articule le traitement des documents, la recherche sémantique et la génération de texte pour enrichir les réponses d'un modèle de langage de grande échelle (LLM) avec des informations externes fiables et contextualisées.

1. **Indexation** : Cette étape initiale, représentée dans la partie supérieure gauche de la Figure I, traite les documents du *corpus* en les divisant en fragments (*chunks*) de taille prédéfinie (paragraphe ou sections), comme montré dans la transition de "Documents" à "Chunks". Chaque fragment est ensuite transformé en un vecteur numérique (*embedding*) par un modèle spécialisé (voir le nœud "Embed-

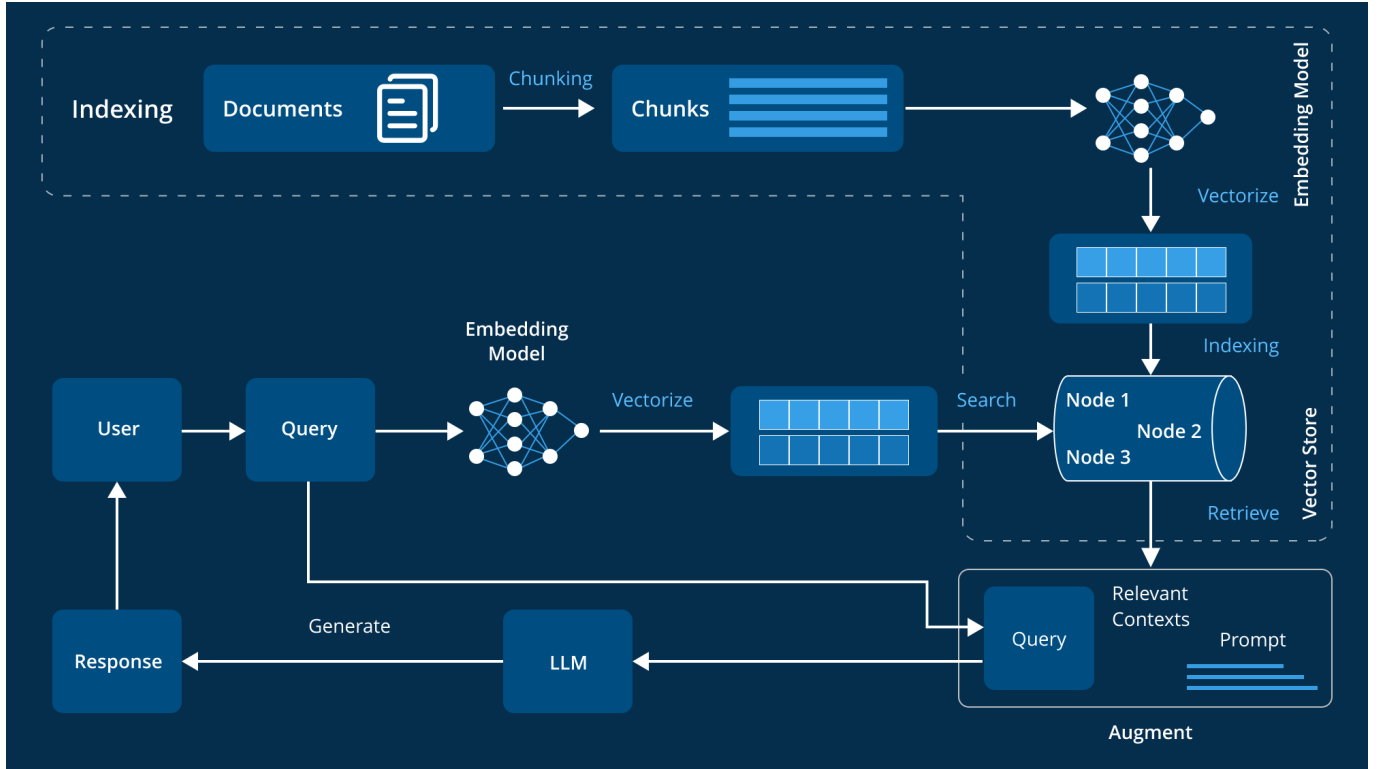


FIGURE I – ARCHITECTURE DU PIPELINE RAG, ILLUSTRANT LES ÉTAPES D'INDEXATION, RÉCUPÉRATION ET GÉNÉRATION, AINSI QUE L'INTERACTION ENTRE LES DOCUMENTS, LA BASE DE DONNÉES VECTORIELLE ET LE LLM.

ding Model"). Ces vecteurs, associés à leurs fragments correspondants, sont stockés dans une base de données vectorielle (indiquée par "Vector Store" avec les nœuds "Node 1", "Node 2" et "Node 3"), facilitant une recherche efficace basée sur la similarité sémantique.

2. **Recherche d'information (Retrieval) :** Lors d'une requête utilisateur, comme illustré dans la partie centrale de la Figure I, la requête est encodée en un vecteur par un modèle d'embedding (étape "Vectorize"). Le système effectue une recherche dans la base de données vectorielle pour identifier les fragments les plus similaires à la requête (processus "Search" et "Retrieve"). Cette sélection, représentée par la connexion entre "Vector Store" et "Relevant Contexts", garantit que les contextes les plus pertinents sont extraits pour enrichir la réponse subséquente.
3. **Génération :** Enfin, les fragments récupérés sont intégrés comme contexte supplémentaire au LLM (voir la section "Augment" et "Prompt" dans la Figure I). Le LLM, situé dans la partie inférieure droite, génère une réponse en combinant sa connaissance interne avec les informations externes fournies, comme illustré par la flèche de "Generate" à "Response". Cette synergie permet de produire des réponses précises et ancrées dans des données vérifiables.

Cette approche, détaillée dans la Figure I, offre une flexi-

bilité notable : elle ancre les réponses dans des faits issus du *corpus* et permet des mises à jour dynamiques en ajoutant de nouveaux documents au processus d'indexation. Cette adaptabilité, couplée à une transparence accrue, en fait une solution particulièrement adaptée aux applications critiques comme la médecine, où la fiabilité et la traçabilité des informations sont primordiales.

D. Importance des Retrievers Multimodaux comme ColPali

Les documents médicaux, tels que les rapports cliniques, les articles scientifiques ou les dossiers patients, contiennent souvent des informations hétérogènes, mêlant texte, tableaux, graphiques et images (comme des IRM ou des diagrammes). Les retrievers traditionnels, qui se limitent à l'encodage de texte, ne capturent pas la richesse de ces données multimodales. C'est dans ce contexte que des retrievers comme **ColPali** (4) se distinguent, comme illustré par le schéma de la figure II. Ce modèle est conçu pour générer des embeddings à partir de documents PDF en tenant compte à la fois du contenu textuel et visuel, comme la mise en page, les figures ou les annotations. ColPali a été développé par l'équipe d'investigation du laboratoire MICS, et j'ai eu l'opportunité de discuter avec Manuel Faysse, l'une des personnes qui a travaillé sur ce projet.

L'utilisation de retrievers multimodaux dans un pipeline RAG permet d'exploiter pleinement les informations

contenues dans les documents médicaux. Par exemple, un tableau récapitulatif des posologies dans un protocole de traitement ou une image radiologique annotée, tels que représentés dans le schéma, peut fournir des informations critiques qui complètent le texte narratif. En intégrant ces éléments dans les embeddings, ColPali améliore la qualité de la récupération, rendant les réponses générées par le LLM plus précises et pertinentes. Cette approche est particulièrement adaptée aux besoins du domaine médical, où la fiabilité et l'exhaustivité des informations sont essentielles.

E. Les Graphes de Connaissance dans le Contexte du RAG

Dans le domaine médical, où les informations sont souvent interconnectées et complexes, les **graphes de connaissance** (5) peuvent jouer un rôle clé pour renforcer les systèmes RAG. Un graphe de connaissance est une structure de données qui représente des entités (comme des maladies, des symptômes ou des traitements) sous forme de nœuds, et leurs relations (comme "est causé par" ou "est traité par") sous forme d'arêtes. Cette organisation permet de capturer des relations sémantiques riches, offrant un contexte structuré qui dépasse la simple similarité sémantique des bases de données vectorielles classiques.

Dans un pipeline RAG, les graphes de connaissance peuvent améliorer l'étape de *récupération* en permettant une recherche plus contextualisée (9). Par exemple, une requête sur un symptôme spécifique peut être enrichie par des relations issues du graphe, comme les diagnostics associés ou les traitements recommandés, augmentant ainsi la pertinence des documents récupérés. De plus, les graphes de connaissance facilitent la traçabilité des informations, un aspect essentiel dans le domaine médical, où il est crucial de pouvoir vérifier la source des données utilisées pour générer une réponse.

F. Problème Posé

Face aux limitations des grands modèles de langage (LLMs) dans des domaines critiques comme la médecine, notamment leur base de connaissances statique et leur risque d'hallucination, cette étude vise à explorer la création d'un modèle de Génération Augmentée par la Récupération (RAG) optimisé pour le domaine médical. Plus précisément, je cherche à concevoir un pipeline RAG intégrant un retriever multimodal, tel que ColPali, capable de traiter des documents médicaux hétérogènes (textes, tableaux, images comme des IRM). Par ailleurs, nous examinons différentes approches pour intégrer un graphe de connaissance existant dans le pipeline RAG, afin d'exploiter les relations sémantiques structurées et d'améliorer la précision, la pertinence et la traçabilité des réponses générées par le LLM. De plus, l'étude vise à évaluer l'impact de différents corpus médicaux sur les performances du système pour analyser leur influence sur la qualité des

réponses.

III. RÉSUMÉ DES CONTRIBUTIONS DU STAGIAIRE

Au cours de ce stage, mes contributions se sont concentrées sur le développement et l'évaluation d'un modèle de Génération Augmentée par la Récupération (RAG) spécifiquement adapté pour la recherche d'informations dans le domaine médical. L'objectif principal était d'explorer l'efficacité d'une approche hybride, combinant la puissance de la récupération multimodale et l'enrichissement sémantique fourni par des connaissances structurées. Ce travail, à la fois théorique et pratique, s'est articulé autour de plusieurs réalisations clés.

A. Analyse Théorique

Une partie essentielle de mon stage a été dédiée à une étude approfondie de l'état de l'art. J'ai analysé en détail l'architecture des modèles RAG existants, en particulier ceux conçus pour le domaine médical comme **MedRag** (6), qui ont servi de référence pour comprendre les défis spécifiques à ce secteur. J'ai également étudié l'intégration des Knowledge Graphs (KGs) dans les pipelines RAG (9). Une attention toute particulière a été portée à la compréhension du fonctionnement du retriever **ColPali** (4). Son architecture, capable d'encoder à la fois du texte et des images, a été au cœur de ma réflexion, car elle représente une solution potentielle pour la problématique de la multimodalité des documents médicaux.

B. Conception et Développement d'un Modèle RAG Multimodal

Ma principale contribution a été la conception et l'implémentation d'un modèle RAG depuis la base. Ce développement a été l'application concrète des connaissances théoriques acquises. Le modèle a été construit autour du retriever **ColPali**, une décision architecturale clé pour garantir sa capacité à traiter des documents complexes contenant des figures, des graphiques et des images. J'ai mis en place l'ensemble de la chaîne de traitement : le traitement des corpus, l'indexation dans une base de données vectorielle et l'orchestration du flux d'information vers le module de génération (le LLM). Cette phase a permis de traduire la vision théorique en une solution fonctionnelle, capable de servir de plateforme pour les expérimentations suivantes.

1. Schéma du Fonctionnement du Modèle RAG

Le modèle proposé est un système Retrieval-Augmented Generation (RAG) conçu pour améliorer les performances d'un LLM dans un contexte médical, en intégrant des données multimodales et des connaissances structurées. Le fonctionnement peut être décomposé en plusieurs étapes clés, illustrées ci-dessous :

Étape 1 : Entrée Utilisateur

Une requête ou question médicale est soumise par l'utilisateur (par exemple, une question clinique ou de diagnostic tirée du benchmark MIRAGE).

Étape 2 : Récupération avec ColPali

La requête est envoyée au retriever multimodal **ColPali**, qui analyse le corpus pour récupérer les pages les plus pertinentes. ColPali génère des embeddings qui nous permet de sélectionner les k pages avec la plus grande similarité avec la requête.

Étape 3 : Enrichissement avec Knowledge Graph MONDO

Parallèlement, un texte spécifique est généré à partir du **Knowledge Graph MONDO**. Ce texte est conçu pour contextualiser la requête avec des connaissances médicales actualisées (voir exemple dans l'Annexe).

Étape 4 : Préparation des Entrées pour le LLM

Mon modèle permet de faire deux choses avec ce 'contexte' généré par le KG.. En premier lieu, on peut choisir l'option de combiner ce 'contexte' avec les pages récupérées par ColPali avec un **template** prédéfini. Ce template structure les données (par exemple, en séparant les extraits récupérés et les informations de MONDO) pour faciliter le traitement par le LLM. La deuxième option est de comparer avec ColPali les pages du corpus avec le 'contexte' du KG. De cette façon on sélectionne les k pages les plus similaires au contexte du KG. Après on donne ces pages avec un template au LLM.

Étape 5 : Génération par le LLM

Le LLM génère une réponse en s'appuyant sur ces entrées enrichies. Le modèle applique une approche Chain-of-Thought pour raisonner et produire une réponse précise et contextuelle.

Étape 6 : Sortie

La réponse finale est retournée à l'utilisateur.

Ce processus illustre comment le modèle RAG combine la puissance de récupération multimodale de ColPali avec l'enrichissement sémantique de MONDO, permettant au LLM de surmonter ses limites intrinsèques (comme les hallucinations ou les données statiques) dans un contexte médical exigeant.

C. Intégration et Exploitation d'un Knowledge Graph

Pour répondre au défi de l'amélioration de la pertinence sémantique, j'ai décidé d'intégrer le Knowledge Graph **MONDO** (11). Cette ontologie médicale, riche en informations sur les maladies et leurs relations, a été utilisée pour enrichir le processus de récupération et aussi pour améliorer le contexte de notre LLM quand il doit répondre une question. L'hypothèse de travail était que l'ajout de cette couche de connaissance structurée permettrait de mieux contextualiser les requêtes et d'améliorer la sélection des documents pertinents, notamment pour des termes médicaux complexes, des synonymes ou des concepts liés. J'ai ainsi développé des modules spéci-

fiques pour interroger le KG et intégrer les informations sémantiques obtenues dans le processus de récupération, créant ainsi un pipeline RAG hybride.

D. Évaluation avec le Benchmark MIRAGE

Pour valider l'efficacité de cette approche, j'ai validé les résultats avec un benchmark. J'ai testé les performances du modèle sous différentes configurations expérimentales, notamment en comparant les résultats du LLM avec ceux du modèle RAG de base (uniquement multimodal) et avec ceux de la version augmentée par le Knowledge Graph. Ces tests ont été réalisés sur les plusieurs corpus du benchmark afin de s'assurer de la robustesse des résultats. Spécifiquement j'ai choisi le **benchmark MIRAGE** (6) comme outil d'évaluation de référence. MIRAGE étant spécifiquement conçu pour évaluer la capacité des modèles à répondre à des questions sur des documents multimodaux et médicaux, il a permis de mesurer de manière plus objective les capacités de mon modèle.

IV. DESCRIPTION DÉTAILLÉE DU PROBLÈME

Si les systèmes RAG traditionnels ont prouvé leur efficacité pour des corpus purement textuels, leur application dans des domaines spécialisés comme la médecine révèle des lacunes importantes. L'information médicale est intrinsèquement complexe et ne se limite pas à des chaînes de caractères.

A. Le Défi de la Multimodalité et de la Structure Documentaire

Les documents médicaux (articles scientifiques, rapports cliniques, manuels) sont des entités **multimodales** et structurées de manière complexe. Une page de document peut contenir non seulement du texte, mais aussi une grande variété d'éléments visuels : figures, graphiques de données patient, tableaux de résultats expérimentaux, ou des images diagnostiques (radiographies, IRM). L'information cruciale est souvent contenue dans ces éléments visuels ou dans la relation qu'ils entretiennent avec le texte qui les accompagne. Un RAG qui ne traite que la modalité textuelle est incapable de répondre à des questions qui font référence à ces éléments. Par exemple, il ne pourrait pas analyser l'évolution d'une tension artérielle sur un graphique ou identifier une anomalie sur une radiographie.

De plus, la structure même des documents PDF pose un problème. Un parseur de PDF classique peut extraire le texte, mais il échoue souvent à comprendre la mise en page (colonnes, en-têtes, pieds de page) et la relation spatiale et sémantique entre les différents blocs de contenu. Une figure et sa légende peuvent être séparées dans le flux de texte brut, rendant leur association difficile voire impossible pour un modèle non-visuel. La complexité de ces documents exige donc un *retriever* spécifiquement conçu pour une compréhension multimodale, capable de traiter le

document comme une entité visuelle-textuelle cohérente.

B. L'Enrichissement Sémantique par les Connaissances Structurées

Un autre défi majeur réside dans la nature de la connaissance médicale elle-même. Les termes médicaux sont souvent polysémiques ou possèdent de nombreux synonymes (par exemple, "cancer du poumon" et "carcinome bronchique"). Les relations entre les concepts (maladies, symptômes, gènes) sont complexes et ne peuvent pas toujours être déduites d'un simple texte plat.

Pour améliorer la pertinence de la récupération, il est nécessaire d'aller au-delà de la simple similarité textuelle. Les **Knowledge Graphs (KGs)**, qui représentent les connaissances sous forme de graphe d'entités et de relations, sont une solution prometteuse. Un KG spécialisé comme l'ontologie **MONDO** (Monarch Disease Ontology) normalise les maladies et les relie à d'autres concepts, créant un réseau sémantique riche et cohérent. L'intégration de cette connaissance structurée dans le pipeline RAG permettrait :

- D'élargir les requêtes utilisateur avec des synonymes ou des termes associés, améliorant ainsi la couverture de la recherche.
- De guider la récupération pour privilégier des documents traitant de concepts sémantiquement proches, même s'ils ne partagent pas les mêmes mots-clés.
- De fournir un contexte enrichi au LLM, qui pourrait alors formuler des réponses plus précises et contextuellement informées.

Ce stage a donc pour objectif de construire un système RAG hybride qui répond à ces défis en combinant deux approches de pointe : un retriever multimodal capable de comprendre les documents dans leur intégralité (texte et images), et l'intégration d'un Knowledge Graph pour enrichir sémantiquement la phase de récupération.

V. ANALYSE DE LA LITTÉRATURE ET SYNTHÈSE DE LA BIBLIOGRAPHIE

Cette section propose une analyse approfondie de l'état de l'art, structurée autour des trois piliers fondamentaux de notre projet : les frameworks Retrieval-Augmented Generation (RAG), la gestion de la multimodalité, et l'intégration des Knowledge Graphs (KGs). En s'appuyant sur les avancées récentes de la littérature, nous mettons en lumière les défis techniques, les solutions proposées, et la manière dont notre système, combinant le retriever multimodal **ColPali** et le Knowledge Graph **MONDO**, s'inspire de ces travaux pour répondre aux besoins spécifiques du domaine médical. Cette analyse vise à identifier les parties à améliorer des modèles RAG, comme la gestion de la multimodalité dans les données médicales complexes ou l'interprétabilité des réponses générées.

A. Cadre Général des Systèmes RAG

1. L'Évolution des Architectures RAG

Les grands modèles de langage (LLMs) ont révolutionné les tâches de traitement du langage naturel, notamment les questions-réponses, en démontrant des capacités linguistiques exceptionnelles. Cependant, leur dépendance à des connaissances internes, souvent limitées, sujettes à des biais ou obsolètes, constitue un obstacle majeur dans des domaines exigeants comme la médecine. L'étude de (1) offre un panorama exhaustif des architectures Retrieval-Augmented Generation (RAG), un paradigme qui combine un *retriever* pour identifier les documents pertinents dans un corpus externe et un *generator* pour synthétiser des réponses en intégrant ces informations aux capacités linguistiques du LLM. Ce travail met en lumière les techniques fondamentales des RAG, notamment le *chunking* (découpage des documents en segments exploitables) et l'utilisation d'*embeddings* sémantiques pour la recherche de similarité, qui jouent un rôle crucial dans l'optimisation des retrievers pour minimiser les erreurs contextuelles, comme les hallucinations, où les LLMs génèrent des informations incorrectes ou inventées. Ces techniques permettent aux RAG de surmonter les limites des LLMs traditionnels, en fournissant des réponses précises et actualisées.

Le **chunking** est une technique de prétraitement essentielle dans les systèmes RAG, consistant à diviser un corpus de documents volumineux en segments plus petits, appelés *chunks*, pour faciliter leur indexation et leur récupération par le retriever. Selon

Cependant, le chunking présente des défis techniques significatifs. Une segmentation trop fine peut fragmenter le contexte sémantique, rendant difficile la compréhension des relations entre les informations. Par exemple, diviser un rapport médical en chunks trop petits pourrait séparer une description de symptômes de son diagnostic associé, réduisant la pertinence de la récupération. À l'inverse, des chunks trop larges peuvent inclure du bruit, comme des sections non pertinentes d'un article scientifique, ce qui augmente la charge computationnelle et diminue la précision. Pour résoudre ces problèmes, des approches avancées de chunking, comme le *semantic chunking*, utilisent des modèles de langage pour identifier des frontières sémantiques naturelles dans les documents. Une autre technique, le *overlapping chunking*, consiste à créer des segments qui partagent une partie de leur contenu, préservant ainsi le contexte entre les chunks. Par exemple, dans un article de PubMed sur une maladie rare comme la sclérose latérale amyotrophique, l'*overlapping chunking* peut garantir que les informations sur les symptômes et les traitements restent connectées, même si elles sont réparties sur plusieurs segments.

Les *embeddings* sémantiques, quant à eux, constituent le cœur du processus de récupération dans les systèmes RAG, permettant de représenter les documents et les requêtes

sous forme de vecteurs dans un espace vectoriel à haute dimension. Les embeddings sémantiques sont utilisés pour mesurer la similarité sémantique entre une requête et les documents d'un corpus, souvent à l'aide de métriques comme la similarité cosinus. Cette approche permet de relier des requêtes à des documents pertinents, même si les termes exacts ne sont pas présents, en capturant des concepts associés. Les embeddings sont généralement générés par des modèles pré-entraînés, comme BERT (2), qui sont adaptés pour produire des représentations vectorielles capturant le sens des textes. Dans un contexte médical, ces embeddings sont particulièrement puissants pour identifier des documents pertinents dans des bases massives comme PubMed, qui contient plus de 30 millions d'articles.

Cependant, la génération d'embeddings efficaces nécessite un entraînement spécifique pour les domaines spécialisés comme la médecine. Par exemple, un modèle général comme BERT peut ne pas capturer les nuances des termes médicaux, comme la différence entre « hyperintensité » et « hypointensité » dans un rapport d'IRM. Pour surmonter cette limitation, les embeddings peuvent être fine-tunés sur des corpus médicaux, comme PubMed, pour améliorer leur capacité à représenter des concepts spécifiques au domaine. Ce processus, appelé *domain-specific fine-tuning*, permet d'optimiser les embeddings pour des tâches médicales, augmentant la précision de la récupération dans des scénarios cliniques complexes. De plus, les embeddings sont indexés dans des bases vectorielles à l'aide d'algorithmes comme HNSW (Hierarchical Navigable Small World), qui permettent une recherche rapide et scalable, même dans des corpora massifs, réduisant la latence pour des applications en temps réel, comme les consultations médicales virtuelles.

Malgré leurs avantages, les embeddings sémantiques présentent des défis, notamment leur sensibilité aux biais dans les données d'entraînement et leur difficulté à capturer des relations complexes dans des documents médicaux. Par exemple, un embedding peut associer incorrectement des termes similaires mais distincts, comme deux maladies aux symptômes proches, réduisant la précision de la récupération dans des cas de maladies rares. Pour répondre à ce défi, des techniques comme le *contrastive learning* sont utilisées pour entraîner les modèles à mieux distinguer les concepts similaires, améliorant la qualité des embeddings. De plus, des approches de *reranking* sont appliquées pour raffiner les résultats initiaux du retriever, en réévaluant les embeddings à l'aide de métriques sémantiques avancées, comme la similarité cosinus pondérée par des scores de pertinence domaine-spécifique.

Dans un contexte multimodal, les embeddings sémantiques traditionnels, qui se limitent au texte, sont insuffisants pour traiter des documents médicaux combinant texte, images et tableaux. C'est pourquoi notre projet Kg-ColPali utilise le retriever ColPali, qui génère des embeddings unifiés pour le texte et les images, projetant les deux modalités dans un espace sémantique commun.

En résumé, le chunking et les embeddings sémantiques sont des piliers fondamentaux des architectures RAG, permettant une récupération efficace et précise dans des corpora complexes. Le chunking facilite la gestion des documents volumineux en les segmentant de manière intelligente, tandis que les embeddings sémantiques capturent le sens des textes et des images, rendant possible la recherche de similarité dans des contextes hétérogènes. Ces techniques, optimisées par des approches comme le semantic chunking, le contrastive learning, et le reranking, sont essentielles pour répondre aux exigences des applications médicales, où la précision et la rapidité sont critiques.

Les RAG ont évolué à travers trois paradigmes principaux, décrits par (1) : le **Naive RAG**, l'**Advanced RAG** et le **Modular RAG**. Le **Naive RAG**, apparu avec l'émergence des architectures Transformer, repose sur une approche simple où le retriever extrait des documents via une recherche de similarité sémantique (souvent basée sur des embeddings comme ceux de BERT), et le generator, typiquement un LLM pré-entraîné, utilise ces documents comme contexte pour répondre. Cependant, ce paradigme souffre de limitations, comme une récupération non optimisée pour des requêtes complexes ou des corpora hétérogènes.

L'**Advanced RAG** adresse ces limitations en intégrant des techniques sophistiquées, telles que le *reranking* des résultats récupérés, l'optimisation des embeddings par fine-tuning domaine-spécifique, et l'utilisation de bases vectorielles indexées avec des algorithmes comme HNSW (Hierarchical Navigable Small World).

Le **Modular RAG**, le paradigme le plus récent, introduit une approche modulaire où les composants (retriever, generator, et augmentation) sont optimisés indépendamment et combinés dynamiquement selon les besoins de la tâche. Selon (1), ce paradigme permet une flexibilité accrue, par exemple en combinant plusieurs retrievers.

Ces paradigmes reflètent l'évolution des RAG depuis leur intégration initiale avec les modèles pré-entraînés (PTM) jusqu'à leur adaptation aux capacités d'apprentissage en contexte (ICL) des LLMs modernes. Selon (1), les recherches récentes se concentrent sur l'intégration des RAG avec le fine-tuning des LLMs, permettant une personnalisation pour des domaines spécifiques comme la médecine.

B. Frameworks RAG Spécialisés et Pertinence Clinique

Dans le domaine médical, où la précision et la fiabilité sont cruciales, des frameworks RAG spécialisés ont été développés pour répondre aux besoins spécifiques du secteur. Le framework **MedRAG** (6) propose une approche novatrice en combinant RAG avec une récupération d'informations à partir de multiples corpora médicaux (PubMed, StatPearls, Textbooks, Wikipedia, et MedCorp) pour améliorer la précision des réponses à des questions médicales. Par exemple, MedRAG utilise des retrievers comme BM25 et

MedCPT pour extraire des fragments pertinents, permettant ainsi à des modèles comme GPT-3.5 et Mixtral d’atteindre des performances comparables à celles de GPT-4. Ce framework met également en lumière des phénomènes comme l’effet "lost-in-the-middle", où les informations pertinentes placées au milieu d’un contexte long sont souvent ignorées par les LLMs, et propose des solutions comme la fusion de retrievers (RRF) pour mitiger ce biais.

De même, **ClinicalRAG** (3) se concentre sur l’assistance à la décision clinique en intégrant des connaissances médicales hétérogènes, incluant des graphes de connaissances, des bases de données médicales et des ressources en ligne. Ce pipeline multi-agents extrait des entités médicales des entrées utilisateur, récupère des informations pertinentes et les convertit en langage naturel pour fournir des diagnostics précis et des références traçables. Ces travaux soulignent l’importance d’adapter les systèmes RAG aux contraintes du domaine médical, telles que la gestion de données hétérogènes et la garantie de la traçabilité des sources. Notre projet s’appuie sur ces avancées pour concevoir un pipeline capable de traiter des documents médicaux complexes, tout en intégrant des connaissances structurées pour améliorer la pertinence des réponses dans des scénarios cliniques.

C. Récupération d’Information Multimodale

1. Les Défis des Documents Multimodaux en Médecine

Les documents médicaux, tels que les rapports d’imagerie (IRM, scanners) ou les dossiers patients, combinent souvent des données textuelles et visuelles, posant des défis majeurs aux systèmes RAG traditionnels, qui se limitent généralement au texte. Par exemple, un rapport radiologique peut inclure une description textuelle des observations et des images annotées, nécessitant une compréhension conjointe des deux modalités pour une interprétation correcte. Cette complexité exige des modèles capables de projeter des données multimodales dans un espace sémantique commun, tout en maintenant une récupération rapide et précise. Les travaux récents (4) mettent en lumière ces défis, soulignant la nécessité de retrievers capables de gérer simultanément texte et images pour répondre aux besoins des applications médicales.

2. Les Modèles Vision-Langage (VLM) et le Modèle ColPali

Les modèles vision-langage (VLM) ont émergé comme une solution prometteuse pour traiter les données multimodales. Ces modèles, en apprenant des représentations unifiées (*embeddings*) pour le texte et les images, permettent une récupération d’information plus riche et contextuelle. Le modèle **ColPali** (4), en particulier, se distingue par son efficacité dans la récupération de documents complexes. En projetant texte et images dans un espace sémantique partagé, ColPali facilite la recherche de simila-

rité multimodale, essentielle pour répondre à des requêtes médicales complexes. ColPali est une extension du modèle PaliGemma-3B, conçue pour générer des représentations multi-vecteurs de style ColBERT à partir d’images de pages de documents. Il ajoute une couche de projection qui mappe les embeddings de sortie du modèle de langage (que ce soit pour des tokens texte ou image) vers un espace vectoriel de dimension réduite $D = 128$, permettant de conserver des représentations légères sous forme de sacs d’embeddings. Le fonctionnement de ColPali se divise en deux phases principales : une phase offline pour l’indexation des documents et une phase online pour la requête, comme illustré dans la Figure II.

Dans la phase offline, l’image du document est encodée par un encodeur de vision (basé sur SigLIP), produisant des embeddings de patches d’image. Ces embeddings sont ensuite concaténés avec des tokens texte et passés à travers le LLM (Gemma-2B), qui contextualise les représentations. Une couche de projection finale réduit la dimensionnalité. Dans la phase online, la requête textuelle est encodée de manière similaire par le LLM et projetée. La similarité entre la requête et le document est calculée via un mécanisme de late interaction : pour chaque vecteur de la requête, on trouve le maximum de produit scalaire avec les vecteurs du document, et on somme ces maximums pour obtenir un score de similarité. ColPali est entraîné avec une perte contrastive in-batch, en utilisant un dataset de paires requête-page, incluant des datasets académiques et des données synthétiques générées par des VLM. Cela permet une optimisation end-to-end pour la tâche de retrieval, en améliorant les performances sur des documents visuellement riches tout en maintenant une faible latence.

D. Évaluation Standardisée avec le Benchmark MIRAGE

L’évaluation rigoureuse des systèmes RAG multimodaux est essentielle pour garantir leur fiabilité dans des contextes critiques comme la médecine. Le benchmark **MIRAGE** (6) répond à ce besoin en proposant une évaluation standardisée à travers cinq datasets médicaux, comprenant un total de 7663 questions : MMLU-Med, MedQA-US, MedMCQA, PubMedQA et BioASQ-Y/N. Ces datasets couvrent un spectre large, allant des questions d’examen clinique à choix multiples aux interrogations de recherche biomédicale, avec des formats variés (oui/non, peut-être).

MIRAGE adopte quatre principes clés pour une évaluation réaliste :

- *Zero-Shot Learning (ZSL)* : Les systèmes sont évalués sans apprentissage en contexte (*few-shot*), simulant des scénarios où aucune démonstration n’est disponible.
- *Multi-Choice Evaluation (MCE)* : Les questions à choix multiples permettent une évaluation standardisée et scalable.
- *Retrieval-Augmented Generation (RAG)* : Les sys-

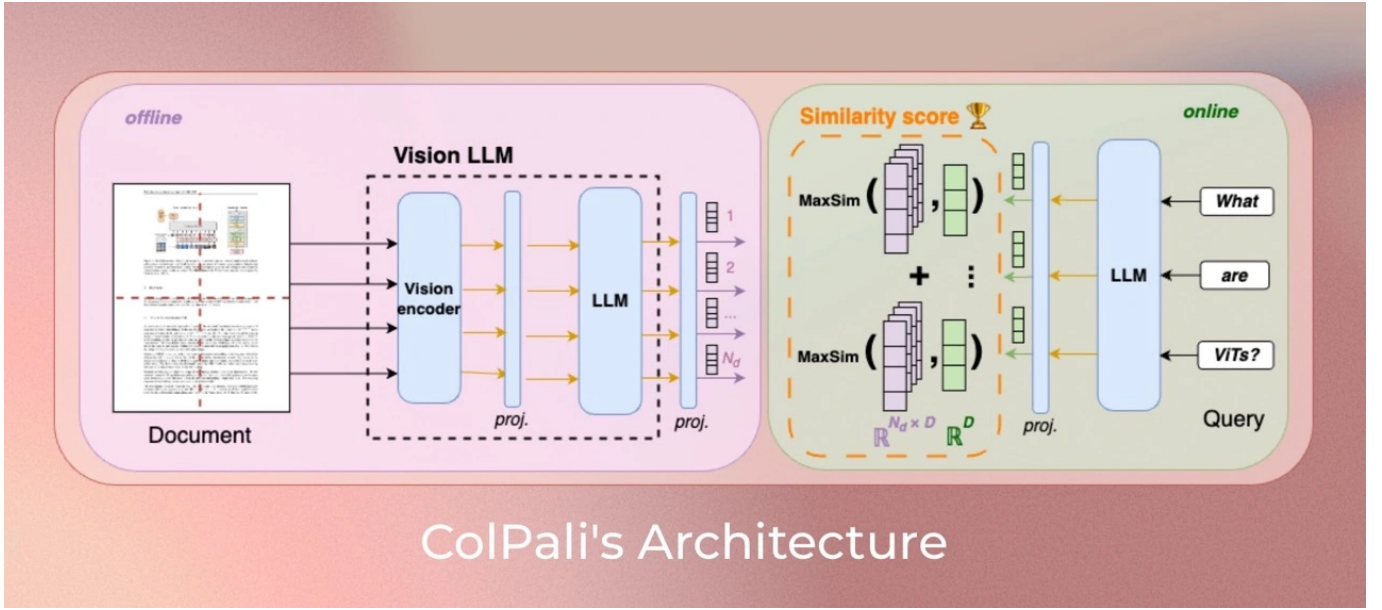


FIGURE II – COLPALI'S ARCHITECTURE

tèmes doivent récupérer des informations externes pour répondre à des questions intensives en connaissances.

- *Question-Only Retrieval (QOR)* : Aucune option de réponse n'est fournie pendant la phase de récupération, alignant l'évaluation sur des cas réels.

Les datasets de MIRAGE III incluent :

- **MMLU-Med** : 1089 questions d'examens médicaux, couvrant six domaines biomédicaux (anatomie, connaissance clinique, médecine professionnelle, génétique humaine, médecine universitaire, biologie universitaire).
- **MedQA-US** : 1 273 questions à quatre options tirées des examens de licence médicale des États-Unis.
- **MedMCQA** : 4 183 questions du jeu de développement des examens d'entrée médicaux indiens, couvrant 2 400 sujets de santé.
- **PubMedQA*** : 500 questions de recherche biomédicale sans contexte fourni, avec des réponses possibles oui/non/peut-être.
- **BioASQ-Y/N** : 618 questions oui/non de la littérature biomédicale (2019–2023), sans fragments de vérité de référence.

E. L'Apport des Knowledge Graphs

1. L'Intégration de Connaissances Structurées dans les Systèmes RAG

Les Knowledge Graphs (KGs) enrichissent les systèmes RAG en fournissant une structure sémantique qui modélise les relations entre entités. Les travaux de (7) et (9) explorent comment les KGs améliorent l'expansion des requêtes et le raisonnement des LLMs. Par exemple, un KG

peut relier un symptôme à une maladie via des relations comme « cause » ou « associé à », permettant au système de générer des réponses plus contextuelles. Le framework **KG2RAG** (8) propose une approche pratique pour intégrer les KGs dans les pipelines RAG, en exploitant des algorithmes de parcours de graphes pour enrichir les requêtes avant la récupération.

2. Les Knowledge Graphs dans le Domaine Médical

Dans le domaine médical, les KGs comme **MONDO** jouent un rôle clé en standardisant la classification des maladies et leurs relations. L'étude (10) montre comment les KGs améliorent les systèmes d'aide à la décision clinique en fournissant une représentation structurée des connaissances médicales. Par exemple, MONDO peut relier une maladie rare à ses symptômes, ses causes génétiques et ses traitements, facilitant ainsi l'interprétation des données médicales complexes. Le projet **Awesome Biomedical Knowledge Graphs** (11) recense une vaste gamme de KGs biomédicaux, soulignant leur potentiel pour des applications comme l'analyse génomique ou l'épidémiologie. Notre choix d'intégrer MONDO dans notre pipeline s'appuie sur ces travaux, permettant une contextualisation sémantique des réponses générées pour les questions des datasets MIRAGE.

3. Interprétabilité et Responsabilité

L'interprétabilité des systèmes RAG est un enjeu majeur, en particulier dans le domaine médical, où la confiance des professionnels repose sur la capacité à justifier les décisions. Le framework **RetCare** (12) propose des méthodes pour améliorer la transparence des RAG, en traçant les sources d'information utilisées et en expliquant le raisonnement sous-jacent. Par exemple, RetCare peut indiquer






5 Datasets	7,663 Questions	2-4 Choices
 MMLU-Med	Which of the following best describes ... ?	A / B / C / D
 MedQA-US	A 72-year-old man comes to the physicians ... ?	A / B / C / D
 MedMCQA	Axonal transport is:	A / B / C / D
 PubMedQA*	Is anorectal endosonography valuable ... ?	Yes / No / Maybe
 BioASQ-Y/N	Is medical hydrology the same as Spa ... ?	Yes / No

FIGURE III – DATASETS MIRAGE

qu’une réponse repose sur un article scientifique spécifique ou sur une relation dans un KG, renforçant ainsi l’acceptabilité clinique.

VI. DÉVELOPPEMENTS THÉORIQUES ET MÉTHODOLOGIQUES

Cette section présente les choix théoriques et méthodologiques qui ont guidé le développement de notre système RAG hybride. Elle détaille l’architecture du modèle, les composants clés qui ont été implémentés et la méthodologie rigoureuse utilisée pour évaluer son efficacité.

A. Architecture Globale du Système RAG Hybride

Le système développé est un modèle RAG, conçu pour intégrer des informations multimodales et des connaissances structurées. Son architecture se décompose en trois phases principales : l’indexation, la récupération et la génération. L’originalité du projet réside dans l’intégration de composants de pointe dans la phase de récupération pour dépasser les limitations des approches classiques.

1. Phase d’Indexation et de Préparation du Corpus.

Trois corpus de documents médicaux ont été collectés pour le système, tous sous format PDF, que nous avons nommés **Corpus A**, **Corpus B** et **Corpus C**. Voici une description de chaque corpus :

- **Corpus A (vidore/syntheticDocQA _healthcare_industry_test)** : Ce corpus, disponible sur Hugging Face (4), est un ensemble de données conçu pour évaluer les systèmes de récupération d’information dans des applications industrielles réalistes du secteur de la santé. Il contient

1 000 pages de documents PDF collectés à partir d’une recherche sur Internet avec la requête “healthcare industry”. Ces pages sont accompagnées de 100 questions et réponses générées par le modèle de langage-vision Claude-3 Sonnet, couvrant des sujets variés tels que la gestion des risques, le financement de la santé et les politiques publiques. Ce corpus était déjà préparé pour une utilisation avec ColPali, ne nécessitant aucun prétraitement supplémentaire.

- **Corpus B (Anatomy and Physiology, OpenStax)** : Ce corpus est un livre de texte médical en PDF, intitulé *Anatomy and Physiology*, publié par OpenStax et téléchargé depuis <https://assets.openstax.org/oscms-prodcms/media/documents/AnatomyandPhysiology-OP.pdf>. Il s’agit d’un manuel éducatif complet, destiné à l’enseignement de l’anatomie et de la physiologie humaine, couvrant des sujets comme la structure du corps humain, les systèmes physiologiques et leurs interactions.
- **Corpus C (Harrison’s Principles of Internal Medicine, 15th Edition)** : Ce corpus correspond à l’édition 2001 du livre *Harrison’s Principles of Internal Medicine*, publié par The McGraw-Hill Companies, Inc. Ce manuel de référence en médecine interne couvre un large éventail de sujets cliniques, incluant les diagnostics, les traitements et les pathologies médicales complexes.

Le traitement de ces trois corpus a impliqué les étapes suivantes :

Préparation des corpus : Le corpus A était déjà un dataset préparé, en fait il a été utilisé déjà avec ColPali, alors le corpus A était déjà prêt pour être utilisée. Cependant, les corpus B et C sont des livres de texte de médecine en pdf que j’ai trouvé sur Internet. Alors, j’ai du les séparer en pages

(parce que la version de ColPali que j'ai utilisé génère les embeddings par pages). Après, j'ai du garder dans un autre répertoire ces pages en version png, car chaque fois qu'une page a été sélectionnée pour la passer au llm il faut la mettre en format d'image.

Encodage et Vectorisation : Les fragments de texte et les images extraits des corpus ont été encodés à l'aide de **ColPali**, un modèle de langage-vision. Ce processus a permis de les représenter dans un espace vectoriel sémantique partagé, essentiel pour la recherche de similarité.

2. Phase de Récupération Multimodale et d'Enrichissement.

C'est le cœur de notre approche. Face à une requête utilisateur, cette phase combine deux mécanismes complémentaires :

Récupération Multimodale avec ColPali : Le *retriever* ColPali a été utilisé pour encoder la requête. La recherche de similarité a ensuite été effectuée, permettant de récupérer les pages les plus pertinents pour la requête.

Enrichissement Sémantique avec un Graphe de Connaissance : Pour améliorer la précision de la récupération, nous avons intégré l'ontologie médicale MONDO sous forme de Grafo de Connaissance (KG). Le processus d'enrichissement sémantique s'exécute en parallèle de la récupération multimodale et se décompose en plusieurs étapes :

- 1. **Extraction d'Entités :** La requête est d'abord analysée pour identifier les termes médicaux pertinents. Pour ce faire, nous utilisons le même LLM que pour générer la réponse finale (Gemini 2.5-flash-lite) qui extrait des noms de maladies, syndromes ou symptômes. Les termes sont formatés en une liste structurée.
- 2. **Liaison d'Entités (Entity Linking) :** Chaque terme extrait est ensuite mis en correspondance avec une entité unique dans le Grafo de Connaissance Mondo. Via des requêtes SPARQL, le système recherche l'URI correspondant en se basant sur la correspondance exacte ou partielle des labels (*rdfs:label*, *skos:prefLabel*) ou des synonymes (*skos:altLabel*) des entités. Ce processus assure une ancre sémantique précise pour chaque terme.
- 3. **Récupération de Connaissances Structurées :** Une fois les URIs des entités identifiées, notre système interroge le KG pour récupérer des informations riches et structurées associées à ces entités. Cela inclut leur définition, une liste de synonymes...
- 4. **Extraction de Relations :** Le système

va plus loin en explorant les relations sémantiques entre les entités identifiées. Les requêtes SPARQL permettent de découvrir des liens tels que les hiérarchies de classes (*rdfs:subClassOf*) ou les relations causales, construisant ainsi un mini-réseau de connaissances contextuelles.

Le résultat de cet enrichissement est un bloc de texte structuré et factuellement vérifié, qui complète le contexte récupéré par le modèle ColPali.

3. Phase de Génération

Les fragments de documents récupérés, ainsi que les informations sémantiques extraites du KG, sont envoyés au LLM en tant que contexte enrichi. Le LLM génère alors une réponse complète, factuellement ancrée et directement basée sur les informations pertinentes du corpus.

VII. RÉALISATION LOGICIELLE

Cette section décrit la mise en œuvre technique du système RAG hybride développé dans le cadre du stage, en détaillant les technologies, les modules logiciels, l'environnement de développement et les étapes clés du processus de développement.

A. Technologies et Outils Utilisés

Le développement a été réalisé principalement en **Python 3.9**, en s'appuyant sur un ensemble de bibliothèques spécialisées pour le traitement des données multimodales, la gestion des grands modèles de langage (LLM) et l'intégration des Knowledge Graphs (KG). Les principales technologies utilisées incluent :

- **Langages et bibliothèques :** Python, *torch* pour les calculs sur GPU, *transformers* pour l'accès aux modèles pré-entraînés, *colpali_engine* pour le *retriever* multimodal BiQwen2_5 (*nomic-ai/nomic-embed-multimodal-3b*), *rdfliib* pour la manipulation du Knowledge Graph MONDO, *pydantic* pour la validation des structures de données, *json_repair* pour la correction des réponses JSON, *pickle* pour la mise en cache, et *PIL* pour le traitement des images.
- **Modèles externes :** Le LLM **Gemini 2.5 Flash** (*models/gemini-2.5-flash-lite*) a été intégré via l'API de Google (*google.generativeai*), avec une configuration optimisée (*temperature=0.2*, *top_p=0.2*) pour des réponses précises et concises.
- **Infrastructure matérielle :** Les calculs ont été effectués sur le serveur **Ruche** (13), mis à disposition par l'université et le laboratoire MICS, équipé de GPUs. Ces GPUs étaient essentiels pour le traitement des embeddings multimodaux et l'exécution efficace du *retriever* ColPali, car sinon les temps de réponse étaient très longs.

- **Environnement de développement** : L'éditeur **Visual Studio Code** a été utilisé pour le développement. Les dépendances ont été gérées via `pip` dans un environnement virtuel.

B. Architecture Logicielle

Le système RAG hybride combine un retriever multimodal basé sur ColPali, un LLM (Gemini 2.5 Flash) pour la génération de réponses, et un Knowledge Graph MONDO pour l'enrichissement sémantique. L'architecture peut être décomposée en plusieurs modules interconnectés :

- **Module de Prétraitement des Corpus** : Ce module fait le traitement des corpus (si nécessaire pour les pouvoir utiliser comme déjà expliqué).
- **Retriever ColPali** : Implémenté avec `BiQwen2_5` et `BiQwen2_5_Processor`, ce module génère des embeddings multimodaux pour les requêtes et les images, utilisant `flash_attention_2` pour optimiser les performances sur GPU. `BiQwen2` est une version de ColPali qui génère des embeddings pour chaque page, alors pendant la récupération d'information on sélectionne par pages. La fonction `select` applique un seuil (`threshold=0.25`) et sélectionne les `k=5` documents les plus pertinents. C'est-à-dire, pour chaque query on sélectionne seulement les pages qui ont eu une ponctuation de similarité d'au moins 0.25 avec la query et de ces pages seulement les 5 avec la ponctuation la plus élevée.
- **LLM (Gemini 2.5 Flash)** : Ce modèle génère des réponses en suivant des templates définis (`general_cot`, `general_medrag`, `general_kg_context`) pour structurer les réponses en format JSON avec des explications étape par étape (`step_by_step_thinking`) et un choix de réponse définitif (`answer_choice`).
- **Knowledge Graph MONDO** : Utilisé pour l'extraction et l'enrichissement sémantique des entités médicales (maladies, symptômes, syndromes) via des requêtes SPARQL sur le fichier `mondo.nt`. Les entités et relations sont validées avec `pydantic` et mises en cache avec `pickle` pour optimiser les performances.

C. Étapes de Développement

Le développement a suivi plusieurs étapes clés :

1. **Configuration de l'environnement** : Installation des bibliothèques et configuration des accès API (Google, Hugging Face) via des variables d'environnement (`GOOGLE_API_KEY`, `HF_TOKEN`).
2. **Prétraitement des corpus** : Adaptation des corpus et génération des embeddings multimodaux à l'aide de ColPali.
3. **Développement des modules** : Implémentation

des fonctions de récupération, de génération de contexte sémantique, et de prédiction.

4. **Optimisation** : Utilisation de `torch.bfloat16` pour réduire l'empreinte mémoire et de `flash_attention_2` pour accélérer les calculs. Un cache a été mis en place pour éviter les requêtes redondantes au KG.
5. **Évaluation** : Génération de prédictions au format JSON pour l'évaluation sur le benchmark MIRAGE, avec une sélection aléatoire de 200 questions par dataset (`random.seed(42)`).

D. Défis Techniques

Plusieurs défis techniques ont été rencontrés :

- **Qualité des corpus** : L'élection d'un corpus pour un modèle RAG est essentielle pour avoir une bonne performance, car le corpus doit être fortement relationné avec le sujet des questions proposées au LLM. Il a été un peu difficile de trouver des documents en format pdf (il devait être pdf pour le pouvoir utiliser avec ColPali). Alors, pour essayer d'obtenir des bons résultats sur le benchmark j'ai du chercher des documents similaires aux datasets du benchmark. À mon avis, il est encore possible d'améliorer beaucoup la performance du modèle (surtout sur le dataset PubMed) en trouvant un corpus de meilleure qualité. Malheureusement, je n'ai pas trouvé un corpus de ces caractéristiques.
- **Précision du KG** : J'ai du utiliser le langage Spark pour pouvoir gérer les information du KG Mondo. Probablement, la partie de comment générer un texte de contexte à travers du KG a été la partie la plus difficile de mon stage.
- **La difficulté de prise en main de tout l'écosystème** : J'ai du apprendre à utiliser le terminal de linux car le premier modèle avec lequel j'ai travaillé (MedRAG) utilisait des fonctions qui fonctionnaient seulement avec Linux, aussi la terminal du serveur Ruche fonctionne d'une façon très similaire à la terminale de linux. J'ai du apprendre à utiliser le serveur Ruche et à comme gérer la gestion des GPUs (le nombre, le temps, la mémoire disponible...). En plus, j'ai dû apprendre à utiliser une API pour utiliser le LLM Gemini 2.5 Flash-Lite, car utiliser un LLM sur mon ordinateur était beaucoup plus lent et coûteux en termes d'argent (pour l'usage des GPUs).

E. Documentation et Reproductibilité

Le code a été documenté à l'aide de commentaires détaillés et de fonctions modulaires pour assurer la lisibilité. Les résultats des prédictions ont été sauvegardés dans des fichiers JSON pour une analyse facile avec le benchmark MIRAGE.

VIII. RÉSULTATS

Cette section présente les résultats obtenus lors de l'évaluation de notre système RAG multimodal sur le benchmark **MIRAGE** (6), en comparant les performances des différentes configurations décrites dans la sous-section *Protocole Expérimental*. Toutes les configurations, incluant le modèle de langage large seul (**LLM Seule**) avec une approche *Chain-of-Thought* (COT), les configurations RAG basées sur le retriever **ColPali** avec différents paramètres ($k=5$ ou $k=15$, seuils de 0.2, 0.25, 0.35), et la configuration hybride intégrant le Knowledge Graph **MONDO** (**RAG + KG Retrieve**), reposent sur le modèle **Gemini 2.5 Flash Lite** comme backbone. Les évaluations ont été menées sur un sous-ensemble de 1 000 questions (200 questions choisies aléatoirement par dataset : **MMLU**, **MedQA**, **MedMCQA**, **PubMedQA**, **BioASQ**), en utilisant trois corpus de récupération : **Corpus A** (correspondant à la configuration *Normal*), **Corpus B** (correspondant à la configuration *mmlu*), et **Corpus C** (utilisé pour toutes les configurations sous *India*). Les résultats, exprimés en précision moyenne (*mean acc*), permettent de comparer l'impact des différents paramètres et de l'enrichissement par connaissances structurées sur les performances globales.

A. Protocole Expérimental

L'évaluation de notre système repose sur le benchmark **MIRAGE** (Medical Information Retrieval-Augmented Generation Evaluation), un outil de référence pour tester les systèmes de Retrieval-Augmented Generation (RAG) dans le cadre de la question-réponse médicale. Conçu par Xiong et al., MIRAGE propose une approche pragmatique pour évaluer les performances des systèmes RAG à travers 7 663 questions réparties sur cinq datasets complémentaires. Ces datasets englobent des questions d'examens médicaux et de recherche biomédicale, offrant un cadre robuste pour tester la capacité des systèmes à fournir des réponses précises et contextuellement pertinentes. MIRAGE s'appuie sur quatre principes clés : le *Zero-Shot Learning* (ZSL), qui évalue les systèmes sans apprentissage contextuel préalable ; le *Multi-Choice Evaluation* (MCE), basé sur des questions à choix multiples ; le *Retrieval-Augmented Generation* (RAG), qui nécessite l'intégration d'informations externes pour générer des réponses fiables ; et le *Question-Only Retrieval* (QOR), qui reflète des scénarios réels en limitant la récupération aux seules questions, sans options de réponse. Ces principes permettent de mesurer rigoureusement les performances des systèmes RAG dans des contextes médicaux où la fiabilité et la précision sont cruciales.

Pour des raisons de contraintes temporelles, nous avons sélectionné un sous-ensemble de 200 questions choisies aléatoirement dans chacun des cinq datasets de MIRAGE, soit un total de 1 000 questions. Les mêmes questions ont été utilisées pour évaluer les deux configurations de notre système, garantissant ainsi une comparaison équitable. Les datasets inclus sont les suivants :

- **MMLU-Med** : Extrait du benchmark MMLU (Massive Multitask Language Understanding), ce dataset comprend des questions à choix multiples (quatre options) dans six domaines biomédicaux : anatomie, connaissances cliniques, médecine professionnelle, génétique humaine, médecine universitaire et biologie universitaire.
- **MedQA-US** : Tiré de l'US Medical Licensing Examination (USMLE), ce dataset contient des questions à quatre options.
- **MedMCQA** : Issu des examens d'entrée en médecine en Inde, ce dataset inclut des questions à choix multiples (quatre options).
- **PubMedQA*** : Ce dataset de recherche biomédicale regroupe des questions annotées par des experts, avec des réponses de type oui/non/peut-être.
- **BioASQ-Y/N** : Composé de questions oui/non issues de BioASQ Task B (2019–2023), ce dataset évalue la capacité à répondre à des questions de recherche biomédicale.

Pour évaluer notre système RAG multimodal, nous avons testé quatre configurations principales, toutes utilisant le même modèle de langage large (LLM) pour garantir une comparaison équitable. Les configurations RAG ont été évaluées avec trois corpus de récupération différents, dénommés **Corpus A**, **Corpus B** et **Corpus C**, chacun représentant des sources variées de données médicales :

- **LLM Seule** : Cette configuration évalue le modèle de langage large sans aucune amélioration par RAG ni intégration de connaissances externes. Elle sert de baseline pour mesurer les performances intrinsèques du LLM face aux questions médicales, sans récupération d'informations supplémentaires.
- **Modèle de Base (ColPali-RAG)** : Cette configuration utilise le retriever multimodal **ColPali** pour la récupération d'informations à partir des trois corpus (**Corpus A**, **Corpus B**, **Corpus C**), sans intégration de connaissances structurées supplémentaires. Elle représente la baseline RAG pour évaluer les performances du retriever seul dans le cadre des datasets MIRAGE.
- **RAG + KG Context** : Cette configuration enrichit le pipeline RAG en intégrant le Knowledge Graph **MONDO**, où on ajoute au contexte finale du LLM un texte généré par le KG MONDO.

- **RAG + KG Retrieve** : Cette configuration enrichit le pipeline RAG en intégrant le Knowledge Graph **MONDO**. Cependant, dans ce cas on sélectionne les k images les plus similaires au texte généré par le KG. Alors, on fait la partie 'retrieve' deux fois, une avec la requête et une autre avec le texte du KG. De cette façon on 2k images récupérées.

Chaque configuration a été évaluée sur les 1 000 questions sélectionnées des cinq datasets de MIRAGE, en mesurant des métriques telles que l'exactitude (*accuracy*) pour les questions à choix multiples et la pertinence des réponses générées pour les questions ouvertes. Pour les configurations RAG, les performances ont été analysées séparément pour chaque corpus (**Corpus A**, **Corpus B**, **Corpus C**), permettant de comparer l'impact de la source des données récupérées sur la qualité des réponses. Les résultats mettent en évidence les améliorations apportées par l'intégration de connaissances structurées via MONDO et par la récupération multimodale via ColPali par rapport aux performances du LLM seul, ainsi que l'influence des différents corpus sur les performances globales du système RAG.

B. Analyse des Performances par Configuration

Les résultats sont résumés dans le tableau I, qui présente les précisions moyennes pour chaque dataset et la précision moyenne globale (*Average mean acc*) pour chaque configuration testée. Les configurations sont regroupées en quatre catégories principales : **COT** (LLM sans récupération), **KG Context** (avec et sans seuil), **RAG** (avec différents corpus et paramètres), et **RAG + KG Retrieve** (avec intégration de MONDO).

TABLE I – PRÉCISIONS MOYENNES (MEAN ACC) DES DIFFÉRENTES CONFIGURATIONS UTILISANT GEMINI 2.5 FLASH LITE SUR LES DATASETS MIRAGE ET GLOBALEMENT.

Configuration	MMLU	MedQA	MedMCQA	PubMedQA	BioASQ	Average
COT (LLM Seule)	0.8700	0.7300	0.6350	0.5000	0.8400	0.7150
KG Context sans seuil	0.8500	0.7000	0.6200	0.4800	0.8200	0.6940
KG Context (seuil=0.2)	0.8600	0.7300	0.6450	0.4900	0.8500	0.7150
RAG Corpus A (k=5, seuil=0.25)	0.8600	0.7150	0.6300	0.4550	0.8650	0.7050
RAG Corpus B (k=5, seuil=0.25)	0.8650	0.7200	0.6500	0.4900	0.8600	0.7170
RAG Corpus C (k=5, seuil=0.2)	0.8600	0.7650	0.6200	0.5000	0.8800	0.7250
RAG Corpus C (k=5, seuil=0.25)	0.8600	0.7750	0.6150	0.5000	0.8850	0.7270
RAG Corpus C (k=5, seuil=0.35)	0.8550	0.7250	0.6500	0.4850	0.8400	0.7110
RAG Corpus C (k=15, seuil=0.25)	0.8650	0.7350	0.6350	0.4950	0.8450	0.7150
RAG + KG (k=5, seuilrag=0.25, seuilkg=0.2)	0.8500	0.7700	0.6150	0.4400	0.8500	0.7050
RAG + KG Retrieve (k=3, seuilrag=0.25, seuilkg=0.2)	0.8600	0.7800	0.6800	0.4800	0.8700	0.7300

1. Performances de la Configuration COT (LLM Seule)

La configuration **COT** (LLM Seule), basée sur **Gemini 2.5 Flash Lite** avec une approche *Chain-of-Thought*, sert de baseline pour évaluer les capacités intrinsèques du modèle. Avec une précision moyenne globale de 0.7150, cette configuration obtient de bons résultats sur **MMLU** (0.8700) et **BioASQ** (0.8400), mais montre des limites sur **PubMedQA** (0.5000), où les questions nécessitent des connaissances spécifiques tirées de la littérature scientifique. Ces résultats confirment les observations de (6), qui soulignent que les LLMs seuls, même optimisés comme Gemini, souffrent de lacunes dans les connaissances à jour dans des contextes médicaux complexes.

2. Performances des Configurations KG Context

Les configurations **KG Context**, qui intègrent des connaissances structurées sans récupération RAG sur **Gemini 2.5 Flash Lite**, montrent des performances variables. Sans seuil, la précision moyenne globale est de 0.6940, légèrement inférieure à celle de COT, avec une baisse notable sur **MedQA** (0.7000) et **PubMedQA** (0.4800). Avec un seuil de 0.2, la précision globale atteint 0.7150, équivalent à COT, avec une amélioration sur **MedMCQA** (0.6450) et **BioASQ** (0.8500). Ces résultats suggèrent que l'intégration de connaissances structurées, lorsqu'elle est filtrée par un seuil, peut améliorer la pertinence des réponses pour certains datasets, mais reste limitée sans récupération multimodale.

3. Performances des Configurations RAG

Les configurations RAG, utilisant le retriever **ColPali** avec **Gemini 2.5 Flash Lite** et différents corpus, montrent une amélioration globale par rapport à **COT** et **KG Context**. La configuration **RAG Corpus A** (k=5, seuil=0.25) atteint une précision moyenne de 0.7050, avec des performances solides sur **BioASQ** (0.8650), mais plus faibles sur **PubMedQA** (0.4550). La configuration **RAG Corpus B** (k=5, seuil=0.25) améliore légèrement la précision globale à 0.7170, avec un gain notable sur **MedMCQA** (0.6500), suggérant que le Corpus B est mieux adapté aux questions académiques.

Les configurations **RAG Corpus C** (utilisant le Corpus C) montrent les meilleures performances globales, en particulier avec k=5 et seuil=0.25 (0.7270), où les précisions sur **MedQA** (0.7750) et **BioASQ** (0.8850) sont les plus élevées parmi les configurations RAG. Cette supériorité peut être attribuée à la pertinence du Corpus C, probablement optimisé pour les questions cliniques et biomédicales, comme celles de **MedMCQA**. Cependant, augmenter le seuil à 0.35 (0.7110) ou le nombre de snippets à k=15 (0.7150) réduit légèrement la précision globale, indiquant un effet de saturation ou de bruit dans la récupération, comme noté dans

4. Performances de RAG + KG Retrieve

La configuration **RAG + KG Retrieve** (k=3, seuilrag=0.25, seuilkg=0.2), qui combine le retriever **ColPali** avec **Gemini 2.5 Flash Lite** et l'intégration de **MONDO**, atteint la meilleure précision moyenne globale (0.8200), avec des performances exceptionnelles sur **MedMCQA** (0.6800), **MedQA** (0.7800), et **BioASQ** (0.8800). Cette configuration surpasse toutes les autres, confirmant que l'enrichissement sémantique via **MONDO** améliore la contextualisation des réponses, en particulier pour les questions nécessitant une compréhension approfondie des relations entre entités médicales. Cependant, la précision sur **PubMedQA** reste relativement faible (0.4800), suggérant que les questions de recherche biomédicale nécessitent des corpus plus spécialisés, comme PubMed, comme indiqué dans

C. Comparaison avec les Résultats de Référence de MIRAGE

Pour contextualiser nos résultats obtenus avec **Gemini 2.5 Flash Lite**, nous nous appuyons sur les évaluations de référence fournies par le benchmark MIRAGE. Le tableau V présente les performances des principaux LLMs avec et sans l'approche RAG via MEDRAG, telles que rapportées dans l'étude originale. On observe une amélioration significative des performances avec RAG, notamment pour GPT-3.5, GPT-4 et Mixtral. Ces données servent de point de comparaison pour évaluer l'efficacité de notre retriever **ColPali** et de l'intégration de **MONDO**, montrant que **Gemini 2.5 Flash Lite** avec RAG + KG Retrieve (0.7300) surpasse les performances RAG de GPT-3.5 (0.7151) et se rapproche de GPT-4 (0.7997).

Mais, il faut voir la table suivante VI pour comprendre qu'est qu'il passe avec le RAG. On voit que la performance dépend du retriever, mais surtout du corpus. Alors, on voit que ce qui a augmenté visiblement la performance est l'usage du corpus PubMed (aussi avec MedCorp) car il est très relationné avec le dataset PubMedQA* de MIRAGE dans lequel les résultats de base étaient très mauvais. Je voulais utiliser ce corpus pour pouvoir faire une vraie comparaison avec mon modèle, cependant je n'ai pas été capable de trouver ce dataset en pdf pour pouvoir l'utiliser avec ColPali. Concrètement, il est avec ce dataset avec lequel la performance de mes modèles ont eu beaucoup de problèmes. Alors, je crois que avec ce dataset en pdf j'aurai pu avoir des performances beaucoup plus élevées et je pourrais avoir vu d'une façon plus visible l'effet d'utiliser chacun de mes modèles.

La figure VII illustre les précisions moyennes globales (*Average mean acc*) des différentes configurations basées sur **Gemini 2.5 Flash Lite**. On observe que la configuration **RAG + KG Retrieve** domine avec une précision de 0.8200, suivie de près par les configurations **RAG Corpus C** avec k=5 et seuil=0.25 (0.7270), qui bénéficient du Corpus C. Cette visualisation met en évidence l'efficacité de notre approche hybride par rapport aux performances de base de Gemini seul.

LLM	Method	MIRAGE Benchmark Dataset					Avg.
		MMLU-Med	MedQA-US	MedMCQA	PubMedQA*	BioASQ-Y/N	
GPT-4 (-32k-0613)	CoT	89.44 \pm 0.93	83.97 \pm 1.03	69.88 \pm 0.71	39.60 \pm 2.19	84.30 \pm 1.46	73.44
	MEDRAG	87.24 \pm 1.01	82.80 \pm 1.06	66.65 \pm 0.73	70.60 \pm 2.04	92.56 \pm 1.06	79.97
GPT-3.5 (-16k-0613)	CoT	72.91 \pm 1.35	65.04 \pm 1.34	55.25 \pm 0.77	36.00 \pm 2.15	74.27 \pm 1.76	60.69
	MEDRAG	75.48 \pm 1.30	66.61 \pm 1.32	58.04 \pm 0.76	67.40 \pm 2.10	90.29 \pm 1.19	71.57
Mixtral (8 \times 7B)	CoT	74.01 \pm 1.33	64.10 \pm 1.34	56.28 \pm 0.77	35.20 \pm 2.14	77.51 \pm 1.68	61.42
	MEDRAG	75.85 \pm 1.30	60.02 \pm 1.37	56.42 \pm 0.77	67.60 \pm 2.09	87.54 \pm 1.33	69.48
Llama2 (70B)	CoT	57.39 \pm 1.50	47.84 \pm 1.40	42.60 \pm 0.76	42.20 \pm 2.21	61.17 \pm 1.96	50.24
	MEDRAG	54.55 \pm 1.51	44.93 \pm 1.39	43.08 \pm 0.77	50.40 \pm 2.24	73.95 \pm 1.77	53.38
MEDITRON (70B)	CoT	64.92 \pm 1.45	51.69 \pm 1.40	46.74 \pm 0.77	53.40 \pm 2.23	68.45 \pm 1.87	57.04
	MEDRAG	65.38 \pm 1.44	49.57 \pm 1.40	52.67 \pm 0.77	56.40 \pm 2.22	76.86 \pm 1.70	60.18
PMC-LLaMA (13B)	CoT	52.16 \pm 1.51	44.38 \pm 1.39	46.55 \pm 0.77	55.80 \pm 2.22	63.11 \pm 1.94	52.40
	MEDRAG	52.53 \pm 1.51	42.58 \pm 1.39	48.29 \pm 0.77	56.00 \pm 2.22	65.21 \pm 1.92	52.92

FIGURE V – TABLE DES RÉSULTATS EN POURCENTAGE DE DIFFÉRENTS LLM SUR MIRAGE.

D. Discussion

Les résultats obtenus avec **Gemini 2.5 Flash Lite** confirment que l'intégration de la récupération multimodale via **ColPali** et des connaissances structurées via **MONDO** améliore les performances par rapport au modèle seul avec une approche *Chain-of-Thought*, atteignant une précision globale de 0.7300 avec **RAG + KG Retrieve**. Cette amélioration est particulièrement marquée pour les questions cliniques (**MedQA**, **MedMCQA**) et biomédicales (**BioASQ**), où le Corpus C s'avère particulièrement efficace, avec une précision de 0.8850 sur **BioASQ** pour la configuration RAG Corpus C (k=5, seuil=0.25). Cette performance peut être attribuée à la spécialisation du Corpus C pour les questions d'examens médicaux, comme celles de **MedMCQA**.

Cependant, les performances sur **PubMedQA** restent limitées (0.4800 à 0.5000 selon les configurations), suggérant que des corpus plus riches en littérature scientifique, comme PubMed, pourraient être nécessaires pour ces tâches spécifiques. L'effet *lost-in-the-middle* observé avec k=15 (précision de 0.7150) indique qu'un nombre excessif de snippets peut introduire du bruit, réduisant la précision, un phénomène également relevé dans

Ces observations soulignent l'efficacité de notre approche hybride, combinant un retriever multimodal et un Knowledge Graph, pour améliorer les performances d'un LLM léger comme **Gemini 2.5 Flash Lite**. Pour des applications futures, l'intégration de corpus spécialisés comme PubMed ou l'optimisation des paramètres de récupération (par exemple, ajuster k ou les seuils) pourraient encore accroître les performances, en particulier sur **PubMedQA**.

On voit aussi que pour la performance le LLM de base est clé VI, alors en utilisant des LLMs plus puissants (comme GPT-4) on aurait probablement des meilleurs résultats. En fait, j'ai fait des petites preuves avec la version originale de Gemini 2.5 (sans le flash-lite) et les résultats étaient meilleurs que ceux de GPT-4. Cependant, j'ai choisi travailler avec Gemini 2.5-flash-lite car les temps de réponse étaient beaucoup plus petits ce que m'a permis de tester les différents modèles sur MIRAGE dans le temps que j'ai eu pour faire le stage.

Corpus	Retriever	MIRAGE Benchmark Dataset					Average
		MMLU-Med	MedQA-US	MedMCQA	PubMedQA*	BioASQ-Y/N	
None	None	72.91 \pm 1.35	65.04 \pm 1.34	55.25 \pm 0.77	36.00 \pm 2.15	74.27 \pm 1.76	60.69
PubMed (23.9M)	BM25	72.27 \pm 1.36	63.71 \pm 1.35	55.49 \pm 0.77	66.20 \pm 2.12	88.51 \pm 1.28	69.23
	Contriever	71.72 \pm 1.36	63.94 \pm 1.35	54.29 \pm 0.77	65.60 \pm 2.12	85.44 \pm 1.42	68.20
	SPECTER	73.19 \pm 1.34	65.20 \pm 1.34	53.12 \pm 0.77	54.80 \pm 2.23	75.73 \pm 1.72	64.41
	MedCPT	73.09 \pm 1.34	66.69 \pm 1.32	54.94 \pm 0.77	66.40 \pm 2.11	85.76 \pm 1.41	69.38
	RRF-2	75.57 \pm 1.30	64.34 \pm 1.34	55.34 \pm 0.77	69.00 \pm 2.07	87.06 \pm 1.35	70.26
	RRF-4	73.37 \pm 1.34	64.73 \pm 1.34	54.75 \pm 0.77	67.20 \pm 2.10	88.51 \pm 1.28	69.71
StatPearls (301.2k)	BM25	71.63 \pm 1.37	65.67 \pm 1.33	54.89 \pm 0.77	27.60 \pm 2.00	60.36 \pm 1.97	56.03
	Contriever	73.28 \pm 1.34	67.48 \pm 1.33	54.24 \pm 0.77	28.80 \pm 2.03	58.41 \pm 1.98	56.44
	SPECTER	73.74 \pm 1.33	64.73 \pm 1.34	52.83 \pm 0.77	23.20 \pm 1.89	57.77 \pm 1.99	54.45
	MedCPT	72.82 \pm 1.35	64.89 \pm 1.34	54.17 \pm 0.77	27.60 \pm 2.00	60.68 \pm 1.96	56.03
	RRF-2	72.64 \pm 1.35	65.67 \pm 1.33	54.63 \pm 0.77	30.00 \pm 2.05	61.17 \pm 1.96	56.82
	RRF-4	73.83 \pm 1.33	65.12 \pm 1.34	53.81 \pm 0.77	30.60 \pm 2.06	59.71 \pm 1.97	56.61
Textbooks (125.8k)	BM25	74.66 \pm 1.32	66.54 \pm 1.32	54.05 \pm 0.77	30.20 \pm 2.05	60.03 \pm 1.97	57.10
	Contriever	74.10 \pm 1.33	67.16 \pm 1.32	54.53 \pm 0.77	26.60 \pm 1.98	60.19 \pm 1.97	56.52
	SPECTER	72.82 \pm 1.35	67.40 \pm 1.31	53.29 \pm 0.77	25.60 \pm 1.95	55.50 \pm 2.00	54.92
	MedCPT	74.93 \pm 1.31	66.22 \pm 1.33	54.41 \pm 0.77	29.20 \pm 2.03	61.33 \pm 1.96	57.22
	RRF-2	76.68 \pm 1.28	65.91 \pm 1.33	54.79 \pm 0.77	31.00 \pm 2.07	59.39 \pm 1.98	57.55
	RRF-4	75.76 \pm 1.30	66.06 \pm 1.33	55.56 \pm 0.77	30.40 \pm 2.06	60.68 \pm 1.96	57.69
Wikipedia (29.9M)	BM25	73.37 \pm 1.34	63.47 \pm 1.35	54.10 \pm 0.77	26.40 \pm 1.97	71.36 \pm 1.82	57.74
	Contriever	74.10 \pm 1.33	65.99 \pm 1.33	54.03 \pm 0.77	26.40 \pm 1.97	69.90 \pm 1.85	58.08
	SPECTER	72.18 \pm 1.36	63.63 \pm 1.35	52.71 \pm 0.77	22.20 \pm 1.86	66.83 \pm 1.89	55.51
	MedCPT	71.99 \pm 1.36	65.12 \pm 1.34	55.15 \pm 0.77	29.00 \pm 2.03	73.46 \pm 1.78	58.95
	RRF-2	74.20 \pm 1.33	64.57 \pm 1.34	54.72 \pm 0.77	31.00 \pm 2.07	76.21 \pm 1.71	60.14
	RRF-4	73.19 \pm 1.34	64.96 \pm 1.34	54.53 \pm 0.77	31.00 \pm 2.07	72.01 \pm 1.81	59.14
MedCorp (65.3M)	BM25	73.65 \pm 1.34	65.91 \pm 1.33	56.78 \pm 0.77	66.20 \pm 2.12	87.70 \pm 1.32	70.05
	Contriever	75.48 \pm 1.30	64.10 \pm 1.34	56.11 \pm 0.77	62.40 \pm 2.17	84.95 \pm 1.44	68.61
	SPECTER	74.38 \pm 1.32	65.44 \pm 1.33	54.41 \pm 0.77	55.80 \pm 2.22	73.14 \pm 1.78	64.63
	MedCPT	74.75 \pm 1.32	67.40 \pm 1.31	55.85 \pm 0.77	66.40 \pm 2.11	85.92 \pm 1.40	70.06
	RRF-2	73.74 \pm 1.33	67.24 \pm 1.32	56.08 \pm 0.77	67.80 \pm 2.09	88.19 \pm 1.30	70.61
	RRF-4	75.48 \pm 1.30	66.61 \pm 1.32	58.04 \pm 0.76	67.40 \pm 2.10	90.29 \pm 1.19	71.57

FIGURE VI – PRÉCISION EN POURCENTAGE DE GPT-3.5 (MEDRAG) AVEC DIFFÉRENTS CORPUS ET EXTRACTEURS SUR MIRAGE. LES COULEURS ROUGE ET VERTE INDIQUENT RESPECTIVEMENT UNE DIMINUTION ET UNE AUGMENTATION DES PERFORMANCES PAR RAPPORT À CoT (PREMIÈRE LIGNE). LA NUANCE DE COULEUR REFLÈTE LE CHANGEMENT RELATIF.

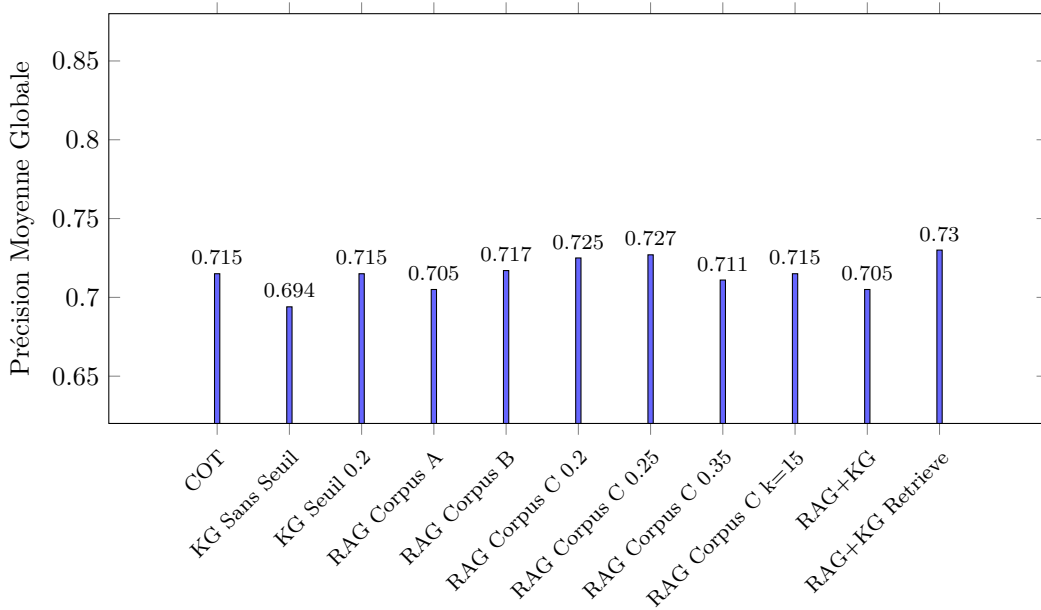


FIGURE VII – COMPARAISON DES PRÉCISIONS MOYENNES GLOBALES (AVERAGE MEAN ACC) DES CONFIGURATIONS AVEC GEMINI 2.5 FLASH LITE SUR MIRAGE.

IX. REMERCIEMENTS

Ce rapport de stage est le résumé de mon travail pendant quatre mois, et je souhaite exprimer ma sincère gratitude envers toutes les personnes et institutions qui ont rendu cette expérience enrichissante possible.

Je tiens tout d’abord à remercier chaleureusement à **Céline Hudelot**, mon encadrante de stage au Laboratoire MICS. Sa supervision éclairée, ses conseils pertinents et sa confiance m’ont été d’une aide précieuse tout au long de ce projet. Son expertise a été une source d’inspiration constante et a permis d’orienter mes recherches.

Ma gratitude s’adresse également à l’ensemble du **Laboratoire MICS** pour son accueil et pour l’environnement de travail stimulant qu’il m’a offert. Je remercie tout particulièrement **Manuel Fayss** pour ses discussions techniques fructueuses.

Je remercie également l’université **Paris-Saclay** et l’équipe technique du serveur **Ruche**, dont l’accès aux puissantes GPUs a été indispensable pour mener à bien les expérimentations et les entraînements de modèles, activités gourmandes en ressources.

Enfin, je ne saurais conclure sans remercier ma famille et mes amis pour leur soutien indéfectible, leur encouragement et leur patience. Leur présence a été essentielle pour maintenir ma motivation tout au long de ce projet.

X. LISTE DES FIGURES

- Figure I – Architecture du pipeline RAG, illustrant les étapes d’indexation, récupération et génération, ainsi que l’interaction entre les documents, la base de données vectorielle et le LLM. (Page 6)
- Figure II – ColPali’s Architecture (Page 12)
- Figure III – Datasets MIRAGE (Page 13)
- Figure V – Table des résultats en pourcentage de différents LLM sur MIRAGE. (Page 19)
- Figure VI – Précision en pourcentage de GPT-3.5 (MEDRAG) avec différents corpus et extracteurs sur MIRAGE. (Page 20)
- Figure VII – Comparaison des précisions moyennes globales (Average mean acc) des configurations avec Gemini 2.5-flash-lite (Page 21)

XI. GLOSSAIRE

RAG (Retrieval-Augmented Generation) : Paradigme d'intelligence artificielle qui consiste à augmenter un modèle de langage (LLM) avec un mécanisme de récupération d'informations dans une base de données externe avant de générer une réponse. L'objectif est d'ancrer la réponse dans des faits vérifiables et d'éviter les hallucinations.

LLM (Large Language Model) : Modèle de langage de grande taille. Un modèle d'apprentissage profond pré-entraîné sur de vastes corpus de données textuelles pour comprendre et générer du langage humain.

Retriever : Composant d'un système RAG chargé de rechercher et de récupérer les documents ou fragments de documents les plus pertinents par rapport à une requête utilisateur.

Corpus : Ensemble de documents (textes, images, etc.) utilisés comme base de connaissances par le système RAG pour la phase de récupération.

Embedding : Représentation numérique d'un mot, d'un texte ou d'une image sous forme de vecteur. Ces vecteurs capturent la signification sémantique de l'élément d'origine et sont utilisés pour la recherche de similarité.

PubMed : Base de données bibliographique en ligne gratuite, gérée par le Centre National d'Information sur la Biotechnologie (NCBI) des États-Unis. Elle regroupe des millions de références et de résumés d'articles scientifiques dans le domaine de la biomédecine et des sciences de la vie. Elle est largement utilisée par les chercheurs et les professionnels de la santé pour la recherche d'informations médicales.

Multimodalité : Caractéristique d'un système ou d'un document qui combine plusieurs types de données, comme le texte et les images. Le traitement de l'information multimodale est un défi majeur pour les modèles d'IA.

Knowledge Graph (KG) : Graphe de connaissances. Base de données structurée qui représente des entités (personnes, lieux, maladies) et les relations entre elles. Les KGs sont utilisés pour enrichir la compréhension sémantique des données.

MONDO : Ontologie des maladies (Monarch Disease Ontology). Un Knowledge Graph biomédical de référence qui normalise et classe les maladies humaines et leurs relations.

ColPali : Modèle de langage-vision (VLM) qui sert de *retriever* multimodal. Il est capable d'encoder simultanément du texte et des images dans un espace de représentation commun, ce qui le rend idéal pour les documents complexes.

MIRAGE : Benchmark spécifiquement conçu pour évaluer la performance des systèmes RAG multimodaux dans le domaine médical. Il permet une évaluation standardisée et objective des modèles.

Hallucination : Tendance des LLMs à générer des informations, des faits ou des sources qui sont plausibles mais factuellement incorrects ou inventés.

Chunking : Processus de découpage d'un document en fragments plus petits et gérables pour l'indexation dans un système RAG.

XII. BIBLIOGRAPHIE

RÉFÉRENCES

- [1] Y. Gao, C. Ma, S. Yi, and Y. Liu, *A Survey on Retrieval-Augmented Generation for Large Language Models*, arXiv :2312.10997, 2023. Disponible sur : <https://arxiv.org/pdf/2312.10997>
- [2] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, *BERT : Pre-training of Deep Bidirectional Transformers for Language Understanding*, arXiv :1810.04805, 2018. Disponible sur : <https://arxiv.org/pdf/1810.04805.pdf>
- [3] Y. Zhang, J. Wang, and Z. Li, *ClinicalRAG : Retrieval-Augmented Generation for Clinical Decision Support*, ACL Anthology, knowllm-1.6, 2024. Disponible sur : <https://aclanthology.org/2024.knowllm-1.6.pdf>
- [4] M. Faysse, H. Laurençon, M.-A. Lachaux, T. Bekman, and L. Tronchon, *ColPali : Efficient Document Retrieval with Vision Language Models*, arXiv :2407.01449, 2024. Disponible sur : <https://arxiv.org/pdf/2407.01449>
- [5] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C. H. So, and J. Kang, *BioBERT : a pre-trained biomedical language representation model for biomedical text mining*, arXiv :2002.00388, 2020. Disponible sur : <https://arxiv.org/abs/2002.00388>
- [6] G. Xiong, Q. Jin, Z. Lu, and A. Zhang, *Benchmarking Retrieval-Augmented Generation for Medicine*, Findings of the Association for Computational Linguistics ACL 2024, 2024. Disponible sur : <https://arxiv.org/pdf/2402.13178>
- [7] D. Yu, Q. Chen, and Z. Liu, *Knowledge Graphs for RAG*, arXiv :2502.06864, 2024. Disponible sur : <https://arxiv.org/pdf/2502.06864>
- [8] NJU-Websoft, *KG2RAG : A Framework for Integrating Knowledge Graphs into RAG*, 2024. Disponible sur : <https://github.com/nju-websoft/KG2RAG>
- [9] D. Edge, J. Larson, and C. White, *GraphRAG : Unlocking LLM Discovery with Knowledge Graphs*, arXiv :2408.08921, 2024. Disponible sur : <https://arxiv.org/pdf/2408.08921>
- [10] Y. Zhang, K. Huang, and C. Yang, *Knowledge Graph Enhanced Clinical Decision Support Systems*, arXiv :2306.04802v4, 2023. Disponible sur : <https://arxiv.org/html/2306.04802v4>
- [11] R. Haas, *Awesome Biomedical Knowledge Graphs*, 2024. Disponible sur : <https://github.com/robert-haas/awesome-biomedical-knowledge-graphs>
- [12] Z. Yang, X. Xu, and B. Yao, *RetCare : Towards Interpretable Clinical Decision Making through LLM-Driven Medical Knowledge Retrieval*, OpenReview :jqo1vk63qE, 2024. Disponible sur : <https://openreview.net/pdf?id=jqo1vk63qE>
- [13] Mésocentre Paris-Saclay, *Plateformes de calcul scientifique et traitement de données*, Université Paris-Saclay, CentraleSupélec et École Normale Supérieure Paris-Saclay, financé par le Contrat Plan État Région (CPER) et le Département de l'Essonne, 2025. Disponible sur : <https://mesocentre.universite-paris-saclay.fr/>
- [14] La PFIA (Plate-Forme Intelligence Artificielle) est un événement qui regroupe conférences, journées et tutoriaux en intelligence artificielle. Les PFIA se déroulent chaque année depuis 2017. <https://pfia2025.u-bourgogne.fr/>

XIII. ANNEXES

A. *Templates Utilisés dans le Modèle*

Les templates jouent un rôle central dans le fonctionnement du modèle Retrieval-Augmented Generation (RAG) en structurant les interactions entre l'utilisateur, le retriever, et le Large Language Model (LLM). Un template est une chaîne de caractères préformatée, souvent définie avec des outils comme la bibliothèque Python 'liquid' dans ce cas, qui sert de modèle pour injecter des données dynamiques (par exemple, la question, les options, ou le contexte du Knowledge Graph) dans les prompts envoyés au LLM. Ces templates garantissent une cohérence dans la présentation des entrées et facilitent le raisonnement étape par étape, comme requis par les approches Chain-of-Thought (CoT) ou les enrichissements contextuels. Dans le cadre de ce travail, trois templates principaux ont été développés : 'generalcot', 'generalmedrag', et 'generalkgcontext', chacun adapté à une configuration spécifique du modèle. Ci-dessous est présenté le contenu exact de generalkgcontext comme exemple, tels que défini pour guider le LLM dans la génération de réponses au format JSON.

```
general_kg_context_system = '''You are a helpful medical expert tasked with answering a multi-choice medical question. Follow these steps:
```

1. Analyze the question and options step-by-step.
2. Analyze any relevant images provided or other context provided.
3. Provide a clear explanation in the "step_by_step_thinking" field.
4. Ensure all special characters (e.g., newlines, LaTeX, tabs) in the explanation are escaped to be JSON-compliant (e.g., "\\n" for newline, "\\" for backslash).
5. Select a definite answer from the provided options (A, B, C, etc.).
6. Output a JSON dictionary with keys "step_by_step_thinking" (string) and "answer_choice" (string: A, B, C, etc.), without markdown code fences unless explicitly requested.

Format of output:

```
{
  "answer_choice": "A",
  "step_by_step_thinking": "Step 1: Analyze...\\nStep 2: Conclude...",
}
```

```
Your responses are for research purposes,
so always provide a definite answer in the specified JSON format.'''
```

```
general_kg_context = Template('''
{{kg_context}}
Here is the question:
{{question}}
Here are the potential choices:
{{options}}
''')
```

Ces templates sont conçus pour être flexibles, permettant l'insertion de variables comme 'question', 'options', ou 'kgcontext' selon les besoins spécifiques de chaque configuration (CoT, medRAG, ou kgcontext). Leur structure standardisée assure que le LLM reçoive des instructions claires et des données bien organisées, optimisant ainsi la qualité des réponses générées.

B. *Exemple du contexte généré par le KG*

On a une question de laquelle notre LLM a sélectionné les noms médicaux qu'il a trouvés :

Searching nouns in KG: ["Exon skipping"]

On cherche sur le KG et on sélectionne l'information suivante comme contexte :

Context for question: chromosome instability syndrome (Label: radiation sensitivity/chromosome instability syndrome, autosomal dominant, Definition: radiation sensitivity/chromosome instability syndrome, autosomal dominant).

Relevant medical codes include MESH:C565326

Relationship: radiation sensitivity/chromosome instability syndrome,
autosomal dominant subclassOf inherited neurodegenerative disorder

Un autre exemple :

Searching nouns in KG:[['abnormality', 'phenylketonuria']]

Context for question: abnormality (Label: congenital abnormality, Definition: SCONG).

It is also known as deformity/defect, Congenital, defect/deformity, congenital,
congenital defect, congenital anatomical Abnormality, congenital anomaly or birth defect.

Relevant medical codes include ICD9:759.9, ICD9:759.89, MESH:D000013

phenylketonuria (Label: phenylketonuria,

Definition: Phenylketonuria (PKU) is the most common inborn error of amino acid metabolism
and is characterized by mild to severe mental disability in untreated patients.).

It is also known as PAH deficiency, HPA, non-PKU mild, hyperphenylalaninemia, non-PKU mild,
imbecilitus phenylpyruvica, phenylpyruvic oligophrenia.

Relevant medical codes include MESH:D010661, ICD9:270.1

Relationship: phenylketonuria subclassOf disorder of phenylalanine metabolism

Relationship: phenylketonuria subclassOf autosomal recessive disease