Pourquoi se limiter à une recherche quand on peut l'étendre ? Amélioration de l'architecture RAG par des stratégies d'expansion de requêtes et d'agrégation de documents

Louis Jourdain
ChapsVision
Paris, France
ljourdain@chapsvision.com

Skander Hellal
ChapsVision
Paris, France
shellal@chapsvision.com

Tony Marini
ChapsVision
Paris, France
tmarini@chapsvision.com

***Abstract*—Le *Retrieval Augmented Generation* (RAG) améliore les réponses générées par des grands Modèles de Langage (LLMs) en récupérant des informations externes. Cependant, la performance de ces systèmes dépend fortement de la qualité des documents récupérés. Dans cet article, nous proposons d'employer plusieurs stratégies d'expansion de requêtes, telles que la reformulation, la décomposition de requêtes et la génération de documents hypothétiques (HyDE) afin d'améliorer cette phase de récupération. Nous intégrons ces techniques dans une architecture RAG optimisée, qui comprend un routeur sémantique pour sélectionner la meilleure stratégie et un module d'agrégation pour fusionner les résultats des requêtes étendues. Nos expériences sur un corpus de 80 000 documents montrent une amélioration notable du rappel et de la qualité des réponses pour les questions complexes, tout en limitant les hallucinations. Cette nouvelle architecture présente un potentiel prometteur pour améliorer les systèmes RAG dans des domaines spécialisés.**

Mots-clés : Retrieval-Augmented Generation, expansion de requêtes, grands modèles de langage, agrégation de documents, recherche d'information.

I. Introduction

Si les grands Modèles de Langage (ou LLM pour *Large Language Models*) sont capables de générer automatiquement du texte d'une qualité qui tient la comparaison avec les productions humaines [1], le fait que leurs connaissances paramétriques soient limitées aux données suffisamment saillantes dans leur corpus d'entraînement les restreint dans de nombreux cas d'usage. Il n'est, par exemple, pas viable d'utiliser directement un LLM comme un système de *Question Answering* dans un domaine spécifique. Pour pallier cette limitation, une nouvelle architecture, nommée RAG (*Retrieval Augmented Generation*), a été proposée [2]. Elle consiste à combiner une étape de recherche documentaire (*retrieve*) dans un large corpus constitué en amont, afin de sélectionner les documents les plus pertinents par rapport à la requête de l'utilisateur (*query*) et d'inclure ces documents dans l'instruction (*prompt*) fournie au LLM, dans le but de lui fournir des connaissances non paramétriques utiles à la réalisation de la tâche de génération.

Bien que l'architecture RAG apparaisse comme une alternative prometteuse et frugale par rapport à d'autres méthodes, telles que le *fine-tuning*, pour adapter les LLMs à des domaines de spécialité [3], son succès dépend de la qualité de la recherche documentaire effectuée. En effet, l'ajout de contextes non pertinents dans l'instruction augmente non seulement le temps et le coût de réponse, mais dégrade également la qualité [4].

Améliorer la phase de récupération des documents est donc crucial pour optimiser les performances d'un système de *Question Answering* basé sur une architecture RAG, ce qui nous amène à aborder des problématiques d'optimisation des moteurs de recherche documentaire. La plupart des *retrievers* utilisés dans les architectures RAG reposent soit sur la fréquence des mots, comme le modèle BM25 [5], soit sur la similarité vectorielle entre le plongement lexical (*embedding*) de la requête et celui des documents, en utilisant des algorithmes tels que *K-Nearest Neighbours* (KNN) ou *Approximate Nearest Neighbours* (ANN) [6]. Avec cette dernière méthode, de loin la plus répandue actuellement, la qualité de la récupération dépend directement du modèle d'*embedding* utilisé. En effet, le modèle employé va mettre l'accent sur certains aspects de la question mais en ignorer d'autres.

De manière plus générale, la qualité de la récupération dépend fortement de la formulation précise de la question ; même une simple erreur orthographique peut réduire significativement la pertinence des documents sélectionnés. Les systèmes de recherche d'information (RI) sont connus pour être sensibles à la variation de leur entrée et moins efficaces lorsqu'ils sont confrontés à des requêtes ambiguës ou très courtes. Ainsi, les documents réellement pertinents sont souvent écartés au profit de ceux qui sont plus proches de la formulation de la question. Dès lors, il sera intéressant de s'inspirer des différents travaux en recherche d'information pour éviter ce phénomène.

Un moteur de recherche commence généralement par une phase de *query understanding*, consistant à extraire de la question l'information pertinente. Cette phase comporte le plus souvent une étape d'élimination des erreurs d'orthographe, de lemmatisation, mais aussi de reformulation (*query rewriting*) en une requête capturant mieux l'intention de la recherche. Optionnellement, une expansion de la requête (*query expansion*) [7] est réalisée pour améliorer la performance de l'extraction, est réalisée. Les techniques classiques consistaient à établir des dictionnaires de synonymes et de termes proches sémantiquement, ou à utiliser des ressources annotées. Cependant, on s'attendait à ce que le passage par des *embeddings* assure déjà l'utilisation de synonymes.

L'intersection de la recherche d'information (RI) et des LLMs a déjà été explorée sous deux angles. Certains auteurs, focalisés sur l'amélioration du RAG, présentent l'intégration de l'expansion de requêtes comme une piste intéressante à creuser [8]. Ainsi, Ma et al. [9] soulignent que, bien que plusieurs équipes de chercheurs aient essayé d'améliorer les modèles d'extraction de documents en les alignant par fine-tuning sur le modèle de génération lors d'une phase d'entraînement commune, peu de travaux ont été réalisés en amont pour tenter de transformer la question de l'utilisateur. Ils suggèrent donc d'investiguer une nouvelle architecture en trois étapes : *rewrite*, *retrieve*, *read*. La transformation qu'ils proposent est d'ordre vectoriel, modifiant le modèle d'*embedding* de la question de l'utilisateur *via* un apprentissage par renforcement. Or, modifier un modèle de cette manière nécessite de disposer d'un grand corpus avec des questions, les documents pertinents et des propositions de réponse, ce qui rend le processus non reproductible dans tout domaine de spécialité sans un lourd travail d'annotation.

Avant que l'architecture RAG ne se démocratise, des chercheurs comme Claveau [10] ont tenté, avec des résultats prometteurs, d'utiliser des modèles génératifs pour effectuer une expansion de requêtes dans le but d'améliorer des systèmes de recherche documentaire [11]–[13]. Cette approche, plus facilement généralisable, a donné lieu à de nombreux travaux établissant le lien entre le paradigme de *query expansion* et le *prompt engineering* [14], ainsi qu'à la proposition de différentes stratégies d'expansion.

Néanmoins, si l'expansion de requêtes est une technique prometteuse pour augmenter le rappel de la phase de récupération, s'appuyer sur des LLMs, susceptibles aux hallucinations [15] risque de dégrader les performances du système, notamment sa précision. Dès lors, comment intégrer l'expansion à l'architecture RAG de manière à limiter les risques ponctuels de dégradation des résultats ?

Cet article se propose :

- d'opérer une présentation synthétique des différentes stratégies d'expansion de requêtes par LLMs et de comparer leurs apports et défauts.
- de proposer une nouvelle architecture RAG plus complexe intégrant l'expansion de requêtes au moyen d'un routeur sémantique et procédant à une agrégation des documents en cas de requêtage multiple.
- de comparer cette nouvelle architecture à celle du RAG classique en les testant sur un corpus spécialisé.
- de proposer un début de réflexion sur l'évaluation non triviale des nouveaux composants de ce système.

II. Stratégies d'expansion de requêtes

Un certain nombre de travaux propose des techniques de reformulation de la question de l'utilisateur pour améliorer les résultats d'un RAG. Si certaines de ces idées semblent prometteuses, la rigueur de leur présentation et de l'évaluation est très variable. Ces méthodes ont rapidement été implémentées dans des frameworks permettant de construire une architecture RAG en quelques lignes de code, mais selon une implémentation opaque pour l'utilisateur. À notre connaissance, aucune étude à ce jour n'a regroupé de manière exhaustive ou comparé ces techniques. Il convient donc de déterminer leurs mérites respectifs.

A. Le Step Back

Cette technique est directement inspirée d'un des premiers articles sur le *prompt engineering*, qui suggérait qu'encourager un LLM à décomposer sa réponse en lui demandant « *take a deep breath and think* » améliorait ses performances sur des tâches de

raisonnement [16]. Le principe du *Step Back* [17] est d'utiliser un LLM avec un prompt demandant de simplifier la question de l'utilisateur en prenant du recul. On obtient alors une question plus générale, ce qui risque d'apporter des documents fournissant plus d'informations contextuelles et d'éviter que la formulation de la question initiale n'influence trop la récupération. Cette technique peut être considérée comme l'équivalent de l'emploi d'hyperonymes dans la phase de *query rewriting* d'un moteur de recherche. Cependant, effectuer un *Step Back* n'est pas nécessairement bénéfique, surtout si la question posée demandait une information précise, ce qui pourrait entraîner une baisse de précision.

B. La reformulation de question (query rewriting)

Cette technique consiste à demander à un LLM de reformuler la question de base de plusieurs manières différentes. En combinant les résultats de recherches effectuées à partir de formulations différentes, on espère récupérer tous les documents intéressants, indépendamment de la lettre exacte de la question de départ. Cette stratégie peut être considérée comme l'équivalent de l'emploi de synonymes dans la phase de *query rewriting* d'un moteur de recherche.

C. La décomposition de requête (query decomposition)

Cette méthode consiste à demander à un LLM de décomposer la question en sous-questions dont les réponses sont nécessaires pour pouvoir répondre à la question globale. Cette décomposition tire parti des capacités de raisonnement et de planification des LLMs et peut faciliter l'extraction d'informations précises, qui, combinées lors de la génération finale, permettent de trouver la bonne réponse (par exemple, « Quelle ville entre Paris et Londres est la plus peuplée ? » se décompose en [« Combien d'habitants y a-t-il à Londres ? », « Combien d'habitants y a-t-il à Paris ? »]). Sur des questions plus abstraites, cette technique peut encourager l'exploration de différentes pistes concrètes étayées par des récupérations plus précises. On utilise donc les capacités de raisonnement des LLMs pour améliorer le résultat de la collecte d'information. L'idée de décomposer une requête en sous-requêtes thématiques est déjà explorée en RI depuis bientôt 20 ans (*Topical Query Decomposition*) [18].

D. Document fictif / HyDE

Lors de la récupération, on compare par similarité vectorielle une question (modalité interrogative, phrase généralement assez brève) à un extrait de document (modalité déclarative, texte plus long, factuel) ce qui peut entraîner une dissonance sémantique qui peut expliquer l'imprécision des résultats. Pour pallier ce problème, il faudrait comparer des objets similaires, ce qui implique soit de créer pour chaque document une liste de questions auxquelles il répond et effectuer la recherche de similarité entre les questions, soit de convertir la question en document fictif. Associer des questions à chaque document entraîne une indexation coûteuse et inefficace si la question diffère trop des questions indexées. La piste du document fictif a été développée dans l'article [19] qui présente la technique HyDE *Hypothetical Document Embeddings*. Si cette proposition est stimulante, utiliser un LLM de la sorte risque d'introduire du bruit car il pourrait halluciner lors de la génération du document fictif, et le contenu de ces hallucinations serait utilisé pour calculer les similarités. L'idée de partir d'un document pour étendre une requête rappelle la technique PRF (*Pseudo-Relevance Feedback*) [20] employée en RI depuis les années 80 et qui consiste à améliorer les résultats en enrichissant automatiquement la requête initiale avec des termes extraits des premiers documents jugés pertinents.

E. Autres stratégies

D'autres stratégies comme par exemple *Least-to-Most prompting* [21] ont été proposées notamment pour traiter les questions à rebonds multiples (*multi hop questions* comme « Quel est le nom du mari de la fille du maire de Héron-ville? »). Ces méthodes nécessitent une architecture itérative différente de celle considérée dans le présent article et ne seront donc pas approfondies.

F. Bilan

Chaque stratégie présente des intérêts particuliers et aide à répondre à certains types de questions, au risque de parfois dégrader la réponse de questions simples et factuelles pour lesquelles la récupération avec la *query* initiale donnait une réponse correcte.

Technique d'expansion	**Types de question**
Step Back	Question comportant des détails trop précis
Reformulation	Question ambiguë ou à la formulation trop spécifique
Décomposition	Question abstraite ou générale comportant plusieurs dimensions, Question dont la réponse nécessite un raisonnement
HyDE / document fictif	Question où la réponse apparaît dans un type de document spécifique

TABLE 1: Usage respectif des stratégies d'expansion selon le type de question posée

Transformer la requête pourrait dégrader les résultats de la recherche compte tenu de la propension des LLMs à commettre des hallucinations. Pour limiter

ce risque, il serait prudent de ne pas utiliser uniquement la requête transformée pour la récupération mais d'amalgamer ces résultats avec ceux de la question originale. De plus, certaines stratégies proposent plusieurs reformulations de la question. Il convient donc de complexifier l'architecture RAG classique pour prévoir le choix d'une méthode d'expansion et possiblement la gestion de plusieurs récupérations en parallèle à partir de différentes requêtes étendues.

III. Adaptation de l'architecture RAG à l'expansion de requêtes

A. Nouvelle architecture proposée

L'intégration de l'expansion de requêtes à de potentielles récupérations multiples soulève plusieurs questions :

- Comment choisir la stratégie d'expansion?
- Comment optimiser la transformation de la requête ?
- Comment amalgamer les résultats produits par différents processus de récupération ?

Chacun de ces trois rôles sera pris en compte par un nouveau composant de l'architecture. En effet l'architecture du RAG classique ne suffit plus à englober un tel système et doit être élargie de façon modulaire [22]. Un module de redirection (ou routeur sémantique) est chargé de déterminer si la question nécessite une expansion, et le cas échéant quelle technique serait la plus bénéfique. Un module d'expansion est chargé de procéder à la transformation de la requête. Après les phases de récupération parallèles, un module d'agrégation amalgame les résultats (voir annexe 1 pour plus de détails).

La phase de récupération peut inclure un *reranking* pour affiner les résultats de similarité vectorielle. Des modèles spécialisés, comme les *cross encoders* sont utilisés pour évaluer la pertinence d'un document par rapport à une requête et produire un classement plus précis [23].

B. Le routeur : rôle et fonctionnement

L'étude de la littérature a permis de sélectionner quatre stratégies d'expansion prometteuses. Néanmoins, chaque article n'évalue que la méthode qu'il introduit sans la comparer à d'autres méthodes d'expansion. De plus, pour les questions simples (« Quelle est la capitale de la France ? »), aucune expansion n'est *a priori* nécessaire. Pour éviter qu'un emploi systématique de l'expansion ne dégrade la précision du système sur les questions simples, il faudrait pouvoir sélectionner la bonne action à effectuer sur une requête (expansion ou non, et le cas échéant quelle stratégie). Pour ce faire, il est possible de s'inspirer de la notion de *routing*, également issue des travaux sur les moteurs de recherche.

Un LLM peut être employé comme agent responsable de la redirection comme démontré par certains travaux [24]. Il s'agit alors d'écrire un prompt décrivant la tâche et les outils disponibles, avec éventuellement des exemples via *few-shot prompting*. La réponse du LLM est analysée pour déterminer la technique d'expansion choisie. Cependant, utiliser un LLM n'offre pas toujours une explication claire du choix de la méthode. Par exemple, l'ordre de présentation des stratégies d'expansion peut influencer les résultats (*order bias*) [25] et sélectionner un mode d'expansion n'est pas trivial.

Notre implémentation du routeur a été testée sur un benchmark de 150 questions réparties équitablement en 5 catégories de difficulté progressive.

- Catégorie 1: questions factuelles admettant une réponse unique.
- Catégorie 2: questions factuelles admettant plusieurs réponses.
- Catégorie 3: questions nécessitant la présentation structurée d'un déroulé chronologique ou d'une chaîne de causalité.
- Catégorie 4: questions abstraites.
- Catégorie 5: questions volontairement ambiguës ou contenant des erreurs.

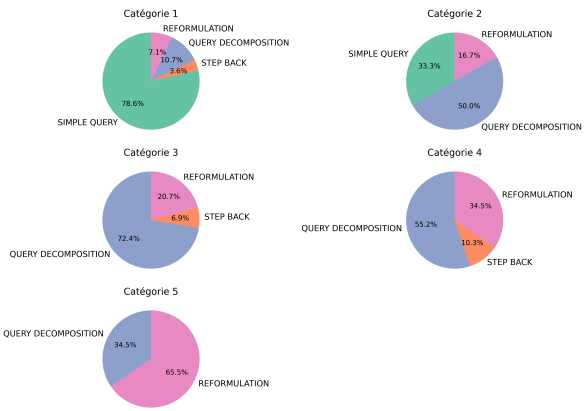

Fig. 1: Choix du routeur de la stratégie d'expansion selon la catégorie de difficulté de la question

On a ensuite analysé les choix du routeur en fonction de la catégorie de difficulté des questions. Pour la plupart des questions simples (deux premières catégories) le routeur choisit de ne pas réaliser d'expansion. Quand la question est plus complexe et nécessite un raisonnement (catégories 3 et 4), le routeur choisit

le plus souvent d'effectuer une décomposition. Enfin lorsque la question est ambiguë ou contient des informations fausses, c'est le plus souvent une reformulation qui est appliquée. Ce comportement correspond *a priori* aux attentes qu'on avait d'un routeur qui n'utilise l'expansion que lorsque nécessaire. Pour s'assurer du bon fonctionnement du LLM en tant que routeur, les trois auteurs ont associé à chaque question d'un corpus de difficulté variable la méthode d'expansion qu'ils considéraient comme la plus appropriée et ont obtenu des distributions de stratégies similaires à celle du LLM du moins en ce qui concerne le choix de réaliser ou non une expansion. L'hésitation est plus grande sur la méthode d'expansion à appliquer.

À noter que la technique du document fictif n'est jamais choisie. Cela peut s'expliquer par le fait que le bénéfice de cette méthode est moins clair que celui d'une reformulation par exemple pour le LLM chargé d'effectuer le *routing*. La description de l'utilité de l'outil qui devait rester succincte dans le prompt n'encourage pas le LLM à la choisir. La méthode de *Step Back* est également très peu utilisée peut-être parce que ses objectifs recoupent en partie ceux plus larges de la reformulation.

C. Algorithmes d'agrégation

Pour limiter les risques qu'une mauvaise expansion ne dégrade les résultats du système, nous n'utilisons pas uniquement la requête modifiée, comme c'est souvent le cas dans d'autres travaux, mais effectuons plusieurs récupérations en parallèle à partir de la requête initiale et de ses expansions. Cela est particulièrement nécessaire lorsque la méthode choisie génère plusieurs reformulations.

Dans l'architecture RAG classique, chaque processus de récupération est limité à j documents. La nouvelle architecture peut potentiellement extraire j documents par récupération soit jusqu'à $(q+1) \times j$ documents, où q correspond au nombre de requêtes modifiées issues de l'expansion, et 1 représente la requête initiale, à condition qu'aucun doublon ne soit présent.

Néanmoins, la fenêtre de contexte des LLMs ayant une taille limitée, il ne sera pas possible d'inclure tous ces documents dans le prompt final. Il convient de plus de les ordonner du plus au moins pertinent pour optimiser la génération finale. Ce problème rejoint un problème plus général en algorithmique qui est celui de la fusion de classements. Le problème n'est pas symétrique car on souhaite accorder plus d'importance aux documents issus de la question initiale qu'aux requêtes modifiées. Cette agrégation des résultats peut s'effectuer selon différentes stratégies :

1) Reranking par rapport à la query principale: Une possibilité serait d'amalgamer tous les documents, d'éliminer les doublons puis d'utiliser un modèle de *reranking* pour reclasser tous les documents en fonction de leur pertinence par rapport à la question initiale. Les documents sont classés une première fois par rapport à la requête qui avait servi à les extraire. On effectue ensuite un second classement en fonction de la question initiale sur tous les documents en même temps. L'avantage de cette méthode est qu'elle limite le bruit possiblement apporté par l'expansion de requêtes au risque de pénaliser des documents intéressants mais plus éloignés sémantiquement de la question initiale. Cette méthode d'agrégation favorise la précision au dépend du rappel.

2) Agrégation par ratio: L'agrégation la plus simple consiste à sélectionner un ratio fixe de documents provenant de la requête principale. Si l'on souhaite obtenir au total k documents, et que l'on fixe un ratio α pour les documents issus de la requête principale, on sélectionne alors les $\alpha \times k$ meilleurs documents récupérés à partir de la requête initiale. Après élimination des doublons, les $(1-\alpha) \times k$ documents restants sont sélectionnés parmi les récupérations des différentes propositions d'expansion. Cette méthode favorise la diversité dans les documents récupérés, mais reste sensible à la qualité de l'expansion effectuée par le LLM et peut entraîner une dégradation de la précision et une augmentation du bruit dans la génération finale.

3) Reciprocal Rank Fusion: L'algorithme de *Reciprocal Rank Fusion*, développé dans le domaine de la recherche d'information [26], calcule un score pour chaque document en fonction de son rang dans les différents classements fusionnés. Les documents ayant les meilleurs scores sont ensuite sélectionnés. La formule pour calculer le score d'un document d est la suivante :

$$\text{RRFScore}(d \in D) = \sum_{r \in R} \frac{1}{s + r(d)}$$

où $r(d)$ est le rang du document d dans chaque classement, et s est un coefficient de lissage (*smoothing coefficient*) ajouté au dénominateur pour réduire l'impact des rangs élevés. Ce coefficient permet d'éviter de survaloriser les documents en tête de classement tout en favorisant ceux qui apparaissent dans plusieurs classements. Cette méthode valorise donc la diversité tout en limitant l'impact des documents issus de requêtes mal formulées.

4) Algorithme hybride: Afin de donner un statut particulier à la récupération par rapport à la requête initiale, nous avons proposé une implémentation hybride. Celle-ci permet de réserver un certain ratio α aux documents récupérés via la question initiale, puis, après dédoublonnage, d'appliquer l'algorithme de *Reciprocal Rank Fusion* (RRF) aux documents issus des autres récupérations (fusion des méthodes 2 et 3).

Pour obtenir un total de k documents, on sélectionne d'abord $\alpha \times k$ documents provenant de la requête principale, puis $(1 - \alpha) \times k$ documents issus de l'application de RRF sur les autres récupérations.

Ces algorithmes de classement permettent ainsi de combiner les résultats de plusieurs recherches et d'obtenir un nouveau classement trié du plus pertinent au moins pertinent.

5) Lost in the middle: Une étude récente [27] a montré que les informations situées au centre du contexte d'un LLM risquent d'être ignorées lorsque celui-ci est trop long, un phénomène appelé *« lost in the middle »*. Bien que ce problème ait été corrigé dans certains modèles récents comme Gemini de Google [28], il persiste dans d'autres, tels que Llama2 [29].

Pour éviter cet oubli, il est recommandé de réordonner les documents : placer les plus importants au début et à la fin, et les moins pertinents au centre (par exemple, pour 6 documents : [1 3 5 6 4 2]), ce qui est trivial à implémenter.

IV. Résultats et Évaluation

Il s'agira de déterminer si la nouvelle architecture RAG proposée obtient de meilleurs résultats que l'architecture classique en comparant les deux systèmes sur un benchmark de questions posées sur un corpus spécialisé. On réfléchira ensuite à la difficile évaluation d'un système de RAG et des nouveaux composants introduits.

A. Jeu de données

On illustrera principalement le présent travail à partir d'une base de données d'un peu plus de 80 000 documents, issus de sources ouvertes sur le web, traitant le conflit au Mali. Cette base de données est tout à fait adéquate pour tester l'efficacité du RAG car elle présente certaines difficultés absentes des benchmarks traditionnels. En effet, la base de données est en français et contient un nombre très important de documents. De plus, les documents font référence à un domaine spécifique. Sur ce jeu de données, un corpus de 150 questions de difficultés progressives a été conçu sur la thématique de la base [cf III.3].

B. Détails sur l'implémentation utilisée

Dans les deux architectures évaluées : Le *retriever* utilisé est un modèle hybride combinant un module de recherche par mots clés et un module de similarité sémantique (*embeddings* du modèle paraphrase-multilingual-mpnet-base-v2[1]) [30] reposant sur l'algorithme HNSW [31] tel qu'implémenté nativement dans le moteur de recherche ElasticSearch. Le *reranker* utilisé est un *cross-encoder* bert-multilingual-passage-reranking-msmarco[2] [32] fine tuné à partir de BERT. Pour les étapes de *routing*, la transformation des requêtes ainsi que la génération finale, le modèle Llama 2 13B Chat[3] de Meta a été employé. Ces modèles ont été installés sur deux cartes graphiques A6000. Le mode d'agrégation choisi pour les résultats présentés est la méthode hybride.

C. Méthodes d'évaluations du RAG

Évaluer l'architecture RAG est complexe car elle intègre divers blocs et modèles pré-entraînés ce qui la rend très sensible au phénomène de propagation de l'erreur. Il est de plus important d'évaluer chaque composant individuellement, ce qui peut se faire de façon intrinsèque (comparaison de sa sortie avec des références) ou extrinsèque (impact de la modification du composant sur la qualité globale du système). Les défis incluent la difficulté d'évaluer les réponses finales, qu'elles soient factuelles ou abstraites, et le besoin de méthodes d'évaluation fiables sans corpus annotés coûteux.

De nombreux laboratoires et entreprises ont exploré l'évaluation du RAG sans aboutir à une méthode unifiée. Un *survey* récent [33], bien qu'ayant le mérite de rassembler des réflexions sur les métriques possibles, ne propose pas de protocole d'évaluation universel. De plus, il se concentre uniquement sur l'architecture RAG classique, sans envisager l'ajout de nouveaux composants. Parmi les frameworks d'évaluation du RAG, on peut citer ARES [34], RAGAS [35] et RAGET[4].

Ces *packages* implémentent des métriques basées principalement sur des comparaisons entre étapes intermédiaires de la réponse, permettant d'évaluer le comportement d'un composant spécifique. Si certains frameworks utilisent des modèles de type BERT pour évaluer certains critères, la plupart adoptent l'approche

[1] https://huggingface.co/sentence-transformers/paraphrase-multilingual-mpnet-base-v2

[2] https://huggingface.co/amberoad/bert-multilingual-passage-reranking-msmarco

[3] https://huggingface.co/meta-llama/Llama-2-13b-chat

[4] https://docs.giskard.ai/en/stable/open_source/testset_generation/rag_evaluation/index.html

« LLM as a judge », où un LLM est utilisé pour juger la sortie d'un autre LLM [36]. Bien que des doutes subsistent quant à la validité des scores numériques et aux biais potentiels [37], [38], cette méthode s'est largement imposée dans l'industrie ces derniers mois. Elle présente l'avantage de ne pas nécessiter d'annotation manuelle et s'adapte facilement à différents corpus.

Devant le manque de consensus, nous avons décidé d'implémenter plusieurs métriques générales pour évaluer le comportement de certains composants du pipeline, notamment la phase de récupération (*retrieve*). Ces métriques consistent à comparer deux à deux les principaux états du système, à savoir la requête de l'utilisateur, les documents sélectionnés et la réponse finale.

Les métriques implémentées sont les suivantes :

1) **Context Relevance** : Les passages sélectionnés sont-ils pertinents pour répondre à la question de l'utilisateur ? (évaluation de la phase de récupération)
2) **Context Adherence/Factuality** : La génération finale est-elle fidèle aux extraits fournis dans la fenêtre de contexte, ou contient-elle des passages hallucinés ? (évaluation de la génération finale et de sa fidélité aux informations fournies)
3) **Overall Quality** : La réponse finale répond-elle de façon satisfaisante et correcte à la question de l'utilisateur ?

Ces métriques peuvent être calculées en établissant des ratios entre les entités nommées présentes dans les états comparés, ou en utilisant l'approche « LLM as a judge » basée sur des prompts inspirés des méthodes RAGAS et ARES.

D. Protocole d'évaluation

Les deux architectures ont été implémentées et connectées à la base de données. Les métriques ont ensuite été calculées sur les réponses des systèmes au benchmark III-B.

Mesurer l'influence de l'expansion de requêtes n'est pas évident car elle n'est pas déclenchée de façon systématique (rôle du routeur). Son apport est conditionné par le comportement d'autres composants (*retriever*, *reranker*, taille et contenu de la base de données). Pour vérifier si l'expansion de requêtes fonctionne correctement, il convient donc de déterminer :

- Si elle dégrade les réponses par rapport à un RAG classique.
- Si la diversité et le nombre de documents rapportés est plus importants.

1) Evaluation individuelle des composants:
Chaque composant a été évalué de façon intrinsèque:

- Le routeur a été testé sur un benchmark hétérogène pour s'assurer que son comportement correspondait aux attentes. Ses productions ont été comparées avec le comportement d'annotateurs.
- Les méthodes d'expansion ont été évaluées sur ce même benchmark et contrôlées par un évaluateur humain à défaut d'avoir un corpus de référence auquel comparer les résultats (phase d'optimisation des prompts).
- Les algorithmes d'agrégation étant déterministes, c'est le choix de l'algorithme à utiliser et éventuellement de ses paramètres à ajuster qui pourra être affiné par une évaluation extrinsèque.

E. Mesure quantitative des apports de l'expansion de requêtes

Il est possible de faire une évaluation quantitative en étudiant le nombre de nouveaux documents apportés par l'expansion. Avec une agrégation hybride, on a ainsi constaté environ 15 % de nouveaux documents.

F. Mesure qualitative des apports de l'expansion de requêtes

Les résultats des différentes métriques pour les deux architectures sont présentés en annexe 4.

On constate que si la performance du système sur les questions simples n'a pas évolué (catégories 1 et 2), ce qui est attendu car le routeur a dû ne pas sélectionner d'expansion (voir III.2), elle a progressé sur les questions plus abstraites (catégories 4 et 5). Ainsi la qualité moyenne des réponses augmente de 4%. La note de *context adherence* donnée par le LLM s'améliore avec l'expansion, ce qui indique que le LLM utilisé comme juge constate un meilleur usage des contextes fournis et moins de segments de réponse hallucinés.

Néanmoins, on constate une légère baisse (environ 2 %) de la pertinence du contexte (*context relevance*), ce qui n'est pas surprenant, car les contextes apportés par l'expansion peuvent être plus éloignés de la question originale. Cependant, cette présence de documents exotiques ne semble pas avoir nui à la qualité générale des réponses, qui a tout de même progressé.

G. Illustration des bénéfices de l'expansion

L'expansion de requêtes est particulièrement fructueuse pour les questions pour lesquelles peu de documents sont présents dans la base de données. Même si le corpus constitué sur le Mali traite

principalement de questions politiques, quelques documents évoquent la situation des femmes et sont plus difficiles à retrouver dans la base de données. L'annexe 5 fournit un exemple de résultat avec et sans expansion de requête.

Les sous-questions proposées explorent divers aspects du problème principal, ou bien ses liens avec différents acteurs, élargissant ainsi le champ de la recherche. D'un point de vue quantitatif, la recherche avec expansion de requêtes retourne un plus grand nombre de documents.

Sans expansion, seuls cinq documents sont retrouvés, parmi lesquels quatre sont réellement pertinents. Le cinquième document est hors sujet, puisqu'il mélange la politique de François Hollande sur les femmes en France et sa politique étrangère au Mali. En revanche, avec l'expansion, sept documents sont retournés, incluant les cinq de départ.

La sous-question 4, portant sur l'éducation, a permis d'identifier un nouveau document parlant d'un projet visant à aider les jeunes filles maliennes à accéder à l'école. Par exemple :

> « Au Mali, le projet SWEDD distribue des bicyclettes permettant aux jeunes filles d'aller à l'école, et vient en aide aux sage-femmes afin qu'elles puissent assurer des services de santé prénatale, natale et postnatale dans les zones pauvres, réduisant ainsi la mortalité maternelle et infantile. »

Ce document s'est révélé particulièrement pertinent pour approfondir la réponse à la question, et il a même été classé premier par le *reranker*. De manière générale, une amélioration notable des résultats a également été constatée pour des questions plus abstraites nécessitant d'explorer différents aspects d'un même problème.

V. Critiques et conclusion

Cet article dresse un bilan des différentes techniques d'expansion de requêtes et propose une nouvelle architecture RAG permettant leur intégration. Bien qu'on démontre qu'elle surpasse l'architecture RAG classique, notamment en termes de rappel, il reste encore à évaluer le comportement global du système. L'évaluation complète du RAG et de chacun de ses composants nécessite des études supplémentaires.

Pour évaluer plus finement les bénéfices de l'expansion de requêtes, on pourrait comparer les réponses avec et sans expansion en demandant à un LLM de juger (selon l'approche dite de *pairwise comparison*) ou en sollicitant des annotateurs humains.

La présente étude a été menée sur un corpus spécialisé. Il conviendrait de vérifier la portabilité de l'architecture proposée à de nouveaux domaines de spécialité en la testant sur d'autres corpus avec des *benchmarks* de taille plus importante.

Des études d'ablation pourraient enfin permettre de déterminer le rôle de chaque composant dans l'amélioration des performances du système.

En absence de corpus de référence, on a évalué la pertinence de la récupération au moyen d'une métrique sur les entités nommées ou de l'approche « LLM as a judge », mais il serait pertinent d'utiliser les métriques classiques sur le *retrieval* (*Exact Match Ratio*, *Mean Average Precision*, *R-Precision*...) à condition d'avoir un corpus annoté.

L'objectif de l'article était de proposer une nouvelle architecture incorporant différentes stratégies d'expansion de requêtes, et pas de comparer le mérite respectif de chaque stratégie. Néanmoins une étude des résultats de l'application systématique de chacune des stratégies serait intéressante et permettrait notamment de vérifier le bien fondé de la technique du document fictif jamais sélectionnée par le routeur. De même, il conviendrait de tester les différentes techniques d'agrégation implémentées pour voir laquelle trouve le meilleur *trade-off* entre précision et rappel.

Bibliographie

[1] Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. 2021. All That's "Human" Is Not Gold: Evaluating Human Evaluation of Generated Text. arXiv:2107.00061.

[2] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. arXiv:2005.11401.

[3] Christophe Bouvard, Mathieu Ciancone, Antoine Gourru, and Marion Schaeffer. 2024. Derby LLM: Évaluation comparative des approches RAG et fine-tuning. In Catherine Roussey Ghislain Atemezing, editor, 10 ème Conférence Nationale sur les Applications Pratiques de l'Intelligence Artificielle, pages 38–47, La Rochelle, France. AFIA-Association Française pour l'Intelligence Artificielle. Backup Publisher: AFIA-Association Française pour l'Intelligence Artificielle.

[4] Florin Cuconasu, Giovanni Trappolini, Federico Siciliano, Simone Filice, Cesare Campagnano, Yoelle Maarek, Nicola Tonellotto, and Fabrizio Silvestri. 2024. The Power of Noise: Redefining Retrieval for RAG Systems. arXiv:2401.14887.

[5] M. E. Maron and J. L. Kuhns. 1960. On Relevance, Probabilistic Indexing and Information Retrieval. J. ACM, 7(3):216–244.

[6] Alexandr Andoni, Piotr Indyk, and Ilya Razenshteyn. 2018. Approximate Nearest Neighbor Search in High Dimensions. arXiv:1806.09823.

[7] Hiteshwar Kumar Azad and Akshay Deepak. 2019. Query expansion techniques for information retrieval: A survey. arXiv:1708.00247.

[8] Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024a. Retrieval-Augmented Generation for Large Language Models: A Survey. arXiv:2312.10997.

[9] Xinbei Ma, Yeyun Gong, Pengcheng He, Hai Zhao, and Nan Duan. 2023. Query Rewriting for Retrieval-Augmented Large Language Models. arXiv:2305.14283.

[10] Vincent Claveau. 2020. Query expansion with artificially generated texts. arXiv:2012.08787.

[11] Lukas Gienapp, Harrisen Scells, Niklas Deckers, Janek Bevendorff, Shuai Wang, Johannes Kiesel, Shahbaz Syed, Maik Fröbe, Guido Zuccon, Benno Stein, Matthias Hagen, and Martin Potthast. 2024. Evaluating Generative Ad Hoc Information Retrieval. In Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval, pages 1916–1929, Washington DC USA. ACM.

[12] Gautier Izacard and Edouard Grave. 2021. Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering. In Paola Merlo, Jorg Tiedemann, and Reut Tsarfaty, editors, Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, pages 874–880, Online. Association for Computational Linguistics.

[13] Rolf Jagerman, Honglei Zhuang, Zhen Qin, Xuanhui Wang, and Michael Bendersky. 2023. Query Expansion by Prompting Large Language Models. arXiv:2305.03653.

[14] Pranab Sahoo, Ayush Kumar Singh, Sriparna Saha, Vinija Jain, Samrat Mondal, and Aman Chadha. 2024. A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications. arXiv:2402.07927.

[15] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2023. A Survey on Hallucination in Large Language Models: Principles, Taxonomy, Challenges, and Open Questions. arXiv:2311.05232.

[16] Chengrun Yang, Xuezhi Wang, Yifeng Lu, Hanxiao Liu, Quoc V. Le, Denny Zhou, and Xinyun Chen. 2024. Large Language Models as Optimizers. arXiv:2309.03409.

[17] Huaixiu Steven Zheng, Swaroop Mishra, Xinyun Chen, Heng-Tze Cheng, Ed H. Chi, Quoc V. Le, and Denny Zhou. 2024. Take a Step Back: Evoking Reasoning via Abstraction in Large Language Models. arXiv:2310.06117.

[18] Francesco Bonchi, Carlos Castillo, Debora Donato, and Aristides Gionis. 2008. Topical query decomposition. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 52–60, New York, NY, USA. Association for Computing Machinery.

[19] Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2022. Precise Zero-Shot Dense Retrieval without Relevance Labels. arXiv:2212.10496.

[20] Hang Li, Ahmed Mourad, Shengyao Zhuang, Bevan Koopman, and Guido Zuccon. 2022. Pseudo Relevance Feedback with Deep Language Models and Dense Retrievers: Successes and Pitfalls. arXiv:2108.1104.

[21] Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. 2023. Least-to-Most Prompting Enables Complex Reasoning in Large Language Models. arXiv:2205.10625.

[22] Yunfan Gao, Yun Xiong, Meng Wang, and Haofen Wang. 2024b. Modular RAG: Transforming RAG Systems into LEGO-like Reconfigurable Frameworks. arXiv:2407.21059.

[23] Rodrigo Nogueira and Kyunghyun Cho. 2020. Passage Reranking with BERT arXiv:1901.04085.

[24] Yuheng Cheng, Ceyao Zhang, Zhengwen Zhang, Xiangrui Meng, Sirui Hong, Wenhao Li, Zihao Wang, Zekai Wang, Feng Yin, Junhua Zhao, and Xiuqiang He. 2024. Exploring Large Language Model based Intelligent Agents: Definitions, Methods, and Prospects. arXiv:2401.03428.

[25] Sheng-Lun Wei, Cheng-Kuang Wu, Hen-Hsen Huang, and Hsin-Hsi Chen. 2024. Unveiling Selection Biases: Exploring Order and Token Sensitivity in Large Language Models. arXiv:2406.03009.

[26] Gordon V. Cormack, Charles L A Clarke, and Stefan Buettcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, pages 758–759, New York, NY, USA. Association for Computing Machinery.

[27] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2023. Lost in the Middle: How Language Models Use Long Contexts. arXiv:2307.03172.

[28] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M. Dai, Anja Hauth, Katie Millican, David Silver, Melvin Johnson, Ioannis Antonoglou, Julian Schrittwieser, Amelia Glaese, Jilin Chen, Emily Pitler, Timothy Lillicrap, Angeliki Lazaridou, et al. 2024. Gemini: A Family of Highly Capable Multimodal Models. arXiv:2312.1180.

[29] Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, et al. 2023. Llama 2: Open Foundation and Fine-Tuned Chat Models. arXiv:2307.09288.

[30] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. arXiv:1908.10084.

[31] Yu A. Malkov and D. A. Yashunin. 2018. Efficient and robust approximate nearest neighbor search using Hierarchical Navigable Small World graphs. arXiv:1603.09320.

[32] Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2018. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. arXiv:1611.09268. arXiv:1708.00247.

[33] Hao Yu, Aoran Gan, Kai Zhang, Shiwei Tong, Qi Liu, and Zhaofeng Liu. 2024. Evaluation of Retrieval-Augmented Generation: A Survey. arXiv:2405.07437.

[34] Jon Saad-Falcon, Omar Khattab, Christopher Potts, and Matei Zaharia. 2024. ARES: An Automated Evaluation Framework for Retrieval-Augmented Generation Systems. arXiv:2311.0947.

[35] Shahul Es, Jithin James, Luis Espinosa Anke, and Steven Schockaert. 2024. RAGAs: Automated Evaluation of Retrieval Augmented Generation. In Nikolaos Aletras and Orphee De Clercq, editors, Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations, pages 150–158, St. Julians, Malta. Association for Computational Linguistics.

[36] Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zi Lin, Zhuohan Li, Dacheng Li, Eric P. Xing, Hao Zhang, Joseph E. Gonzalez, and Ion Stoica. 2023. Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena. arXiv:2306.0568.

[37] Shreya Shankar, J. D. Zamfirescu-Pereira, Björn Hartmann, Aditya G. Parameswaran, and Ian Arawjo. 2024. Who Validates the Validators? Aligning LLM-Assisted Evaluation of LLM Outputs with Human Preferences. arXiv:2404.12272.

[38] Peiyi Wang, Lei Li, Liang Chen, Zefan Cai, Dawei Zhu, Binghuai Lin, Yunbo Cao, Qi Liu, Tianyu Liu, and Zhifang Sui. 2023. Large Language Models are not Fair Evaluators. arXiv:2305.17926.

ANNEXES

A. Annexe 1 : Nouvelle architecture proposée

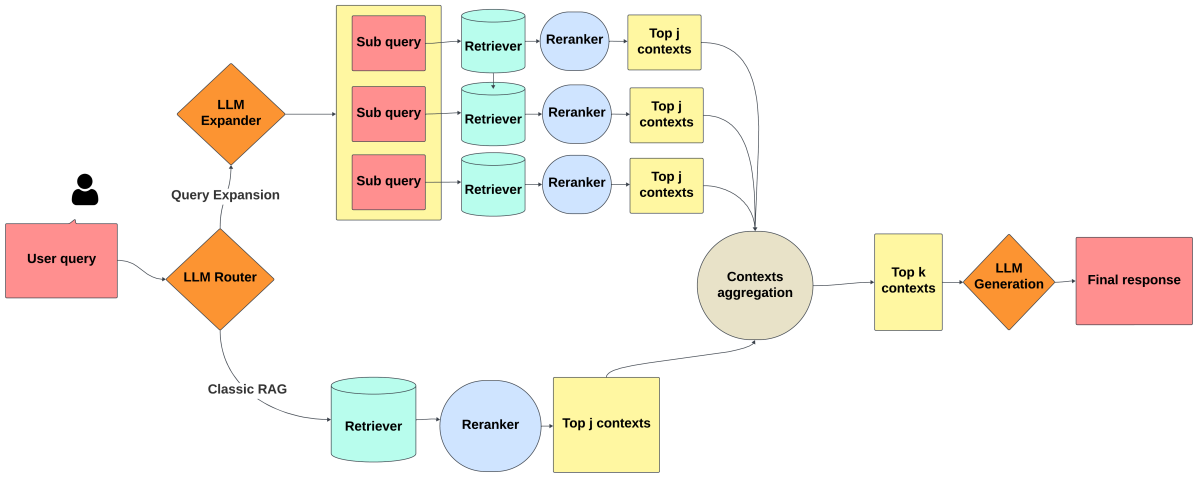

Fig. 2: Architecture RAG intégrant l'expansion de requêtes.

B. Annexe 2 : Exemples de questions du benchmark de difficulté progressive

Catégorie	Exemples de questions
Catégorie 1 : Questions factuelles demandant une réponse unique	Quel est le nom du fleuve qui traverse le Mali ? Quelle est la principale ressource naturelle exploitée au Mali ? Quelle est la date d'indépendance du Mali ?
Catégorie 2 : Questions factuelles attendant plusieurs éléments de réponse	Nommez trois organisations régionales africaines impliquées dans la résolution de la crise malienne. Donnez trois exemples de mesures prises par le gouvernement malien pour lutter contre le terrorisme. Quels sont les groupes armés signataires de l'Accord pour la paix issu du processus d'Alger ?
Catégorie 3 : Questions nécessitant la présentation structurée d'un déroulé chronologique ou d'une chaîne de causalité	Décrivez la séquence d'événements qui ont conduit à l'accord de paix d'Alger et expliquez son impact sur la situation politique au Mali. Reconstituez la chronologie des négociations et des accords visant à résoudre la crise malienne depuis le début du conflit. Analysez le parcours et l'impact d'Amadou Toumani Touré sur la scène politique du Mali.
Catégorie 4 : Questions abstraites	Comment le changement climatique peut-il aggraver les tensions et les conflits au Mali ? Comment le trafic illicite, y compris le trafic de drogue, est-il lié au conflit au Mali ? Quel rôle jouent les groupes ethniques dans la dynamique du conflit au Mali ?
Catégorie 5 : Questions volontairement ambiguës ou contenant des erreurs factuelles volontaires	Comment la récente réorganisation des forces armées maliennes a-t-elle amélioré l'efficacité opérationnelle ? Quelles mesures ont été prises pour protéger les droits des minorités ethniques au Mali ? Quelles initiatives ont été lancées pour résoudre la crise énergétique au Mali ?

TABLE 2: Exemples de questions classées par catégories

C. Annexe 3 : Prompts utilisés

1) Prompt utilisé pour le routeur:

As an expert, your task is to answer a user query. You have several strategies available:

- **SIMPLE QUERY**: the query is straightforward and can be answered directly.
- **QUERY DECOMPOSITION**: the query is complex and can be broken down into simpler sub-questions.
- **REFORMULATION**: the query is unclear, poorly written, or abstract.
- **STEP BACK**: the question is too precise; asking a more general question might yield relevant information.
- **FICTIVE DOCUMENT**: the expected answer may be found in a specific type of document (e.g., official statement, news article).

The language of the query is not relevant to pick a strategy. Output the name of the selected strategy in capital letters without any other comment. Then jump a line and briefly justify this choice. Do not be verbose or acknowledge instructions.

`{user_query}`

2) Prompt utilisé pour le Step Back:

As an expert, your mission is to step back from a question that is asked and rephrase it as a more general question. The new question must be easier to answer than the first one.

Examples:

- **Initial query**: "What studies did Assimi Goita pursue before becoming president?"
  Step Back query: "What is Assimi Goita's background?"
- **Initial query**: "Where did the most recent terrorist attack occur that resulted in over ten deaths in Mali?"
  Step Back query: "What were the most deadly terrorist attacks in Mali's recent history?"

Only output the new question.

`{user_query}`

3) Prompt utilisé pour la reformulation de requêtes:

As an expert, your mission is to rephrase the following question in three different ways to make it clearer. Provide three reformulations of the question and return only a numbered list, skipping lines between each question.

`{user_query}`

4) Prompt utilisé pour la décomposition de requêtes:

As an expert, your mission is to generate a list of specific sub-questions that need to be answered to address the user's general question. Here are some guidelines:

- Be precise.
- Each sub-question should be relevant.
- The answer should be findable in a database.
- Provide the results as a numbered list of questions (maximum 5 questions). Return only a numbered list, skipping lines between each question. Do not add any comments.

`{user_query}`

5) Prompt utilisé pour la création de documents hybrides:

Generate a document that answers the given question. The document could be, for instance, extracted from a newspaper article or an official document or be any document you seem fit to contain the answer to the query. Provide only the document.

`{user_query}`

6) Prompt utilisé pour la génération finale:

You're a reliable system that specializes in answering questions. Your mission is to generate an answer based on the question posed and the context provided. You always respond using the contextual information given to you in the context and without using any other information.

Here are a few rules to follow:

1) Answer in `{lang}`.
2) Only consider context that is relevant to answer the question.
3) Do not refer directly to the contextual information provided.
4) Avoid formulations such as "According to the information provided."
5) If the given contexts do not allow you to answer the question, answer "I don't know."

`{user_query}`

D. Annexe 4 : Comparaison des deux architectures RAG

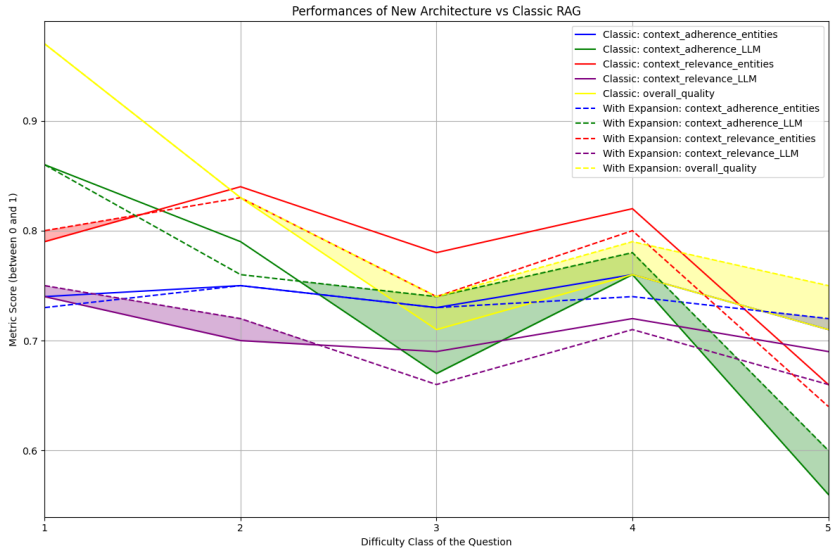

Fig. 3: Résultat de l'évaluation comparative des deux systèmes

E. Annexe 5 : Exemple de documents avec et sans expansion

Question : Détaillez les initiatives prises pour promouvoir l'égalité des sexes et les droits des femmes au Mali.

Réponse du RAG classique	**Réponse avec Expansion de Requête**
Au Mali, plusieurs initiatives ont été prises pour promouvoir l'égalité des sexes et les droits des femmes. Parmi ces initiatives, on trouve : • **Code du mariage et de la tutelle** : Législation visant à garantir les droits des femmes. • **Conférence des femmes africaines de 1962** : Participation à un événement important pour l'unité des femmes en Afrique. • **Unions de femmes** : Soutien au niveau local pour promouvoir les droits des femmes. • **Participation à l'agenda 2030 de l'ONU** : Contribution aux objectifs de développement liés à l'égalité des sexes. • **Action de l'Union européenne** : Soutien à la gouvernance et à l'égalité des sexes dans le cadre d'actions contre l'insécurité.	Au Mali, plusieurs initiatives ont été prises pour promouvoir l'égalité des sexes et les droits des femmes. Parmi ces initiatives, on trouve : • **Projet SWEDD** : Une initiative de grande envergure soutenue par l'ONU pour promouvoir l'égalité des sexes, l'éducation des filles, et la réduction des mariages d'enfants. • **Action de l'Union européenne** : Continuation des efforts pour soutenir la gouvernance et les droits des femmes, avec davantage de détails sur l'implication dans les régions affectées. • **Code du mariage et de la tutelle** : Mention plus précise du rôle des femmes maliennes dans l'élaboration de ce code. • **Conférence des femmes africaines de 1962** : Informations contextualisées sur l'objectif de promotion de l'unité des femmes africaines. • **ONG comme CAFO** : Rôle des ONG nationales dans la lutte contre les obstacles sociaux, économiques et juridiques empêchant les femmes de jouir pleinement de leurs droits. • **Promotion de l'éducation des femmes** : Mention du soutien de la Banque mondiale à l'éducation des filles, visant à garantir les droits des femmes et à prévenir les violations pendant les périodes de conflit.
Sous-questions	**Documents Supplémentaires Apportés Par l'Expansion**
• Quelles organisations sont impliquées ? • Quelles sont les mesures législatives prises ? • Quelles initiatives de sensibilisation et de formation ont été mises en place ?	L'expansion de requête a permis d'apporter plusieurs documents supplémentaires pertinents : • **Document SWEDD** : Détaille les initiatives dans la région du Sahel pour promouvoir l'égalité des sexes, l'éducation, et la sécurité sociale. • **Document sur l'implication des ONG** : Montre le rôle des organisations locales dans la promotion des droits des femmes.

TABLE 3: Comparaison des résultats des deux architectures sur une question précise