

Récap des avancées du 16 au 24 avril 2026

Les nouvelles informations sont en bleu dans le document.

Administratif et organisationnel

- Audition blanche en visio pour Thèse à Lannion prévue le 22/05 de 9h30 à 10h30 -> J'aurai besoin d'une salle de réunion
- [Rendez-vous de mercredi 29/04 14h-16h](#) : J'ai redécouvert un RDV dans mon agenda à 16h30 donc on ne pourra pas dépasser 16h10

Bibliographie pour le projet

Dataset

J'ai identifié plusieurs datasets en plus des benchmarks repérés. Voici ce que j'ai retenu:

- [MIRACL \[Zhang et al., 2022\]](#) : Benchmark multilingue avec une partie en français à extraire.
- [BERGEN](#) se base sur deux datasets (pour ce qui est de l'aspect multilingue) :
 - [MKQA \(Github, HuggingFace, \[Longpre et al., 2021\]\)](#) qui propose des questions en plusieurs langues dont le français MAIS ne met pas de documents à dispositions.
 - [XOR-TyDi QA \[Asai et al., 2021\]](#) qui propose un ensemble de documents et de questions dans plusieurs langues dont les réponses peuvent dépendre de documents d'une autre langue. *Cela pose problème pour l'extraction des entrées en français.*
- [TyDi QA \[Clark et al., 2020\]](#) est la version plus simple de XOR-TyDi QA. Les documents permettant de répondre aux questions sont dans la langue de la question. On peut extraire les questions en français sans grande inquiétude.

Je ne pense pas que les données en français proposées par BERGEN soient intéressantes. Par contre, **MIRACL** et **TyDi-QA** sont des ressources de données pour la tâche de QA utiles.

[Après relecture, il n'y a pas de données en français dans TyDi-QA. On ne pourra pas l'exploiter.](#)

Modèles de RAG

J'ai exploré deux modèles parmi ceux que nous avons trouvés et ceux que j'ai récupéré des proceedings :

- [\[Bellart & Deleruyelle 2025\]](#) *TalanSeeker : Application de l'architecture RAG pour la sélection de profils en conseil* La tâche visée est l'extraction de CV correspondant aux besoins de recrutement exprimés via un chatbot. Le [github](#) fourni presque tout ce qu'il faut pour faire fonctionner l'application qui contient le modèle mais je n'ai pas trouvé sa définition dans le code.
- [\[Tran et al., 2025\]](#) *Génération augmentée de récupération pour les journaux historiques* Le code est aussi disponible sur [github](#) et me paraît plus accessible. Le modèle est fourni avec la librairie `langchain` qui permet de construire toute la pipeline sans se battre pour faire communiquer les différents composants. Cela n'empêche pas de manipuler chaque partie du modèle (extracteur, générateur, reranker,...) individuellement à l'aide de la librairie `huggingface` et donc d'appliquer MechIR dessus. J'ai choisi de continuer sur cette architecture.

[\[Tran et al., 2025\]](#) fait du Question Answering sur des journaux historiques en français en finnois et en allemand. Ils proposent aussi une évaluation quantitative sur les parties françaises et finnoises de MIRACL. Les données sont exploitées en deux temps. On extrait des documents sur la base des titres (et résumés). Si on ne trouve pas de documents (ie. si aucun documents n'est jugé suffisamment proche) alors on bascule sur le module de recherche web. Sinon, on effectue une seconde extraction mais cette fois en exploitant le corps des documents puis on les reclasse. Si on ne trouve pas de document satisfaisant, on bascule aussi sur le module de recherche web. Une fois les documents obtenus (par recherche ou par extraction puis reranking), on les passe avec la requête au générateur à l'aide d'un prompt à trou complété. L'architecture proposée est la suivante:

- Les deux extractions se font avec des bases de données d'embeddings. Ils sont construits avec la version multilingue de E5 [\[Wang et al., 2024b, Wang et al. 2024a\]](#) et la base est gérée avec le moteur [Chroma](#) en utilisant la similarité par cosinus et la Maximal Marginal Relevance [\[Carbonell & Goldstein 1998\]](#).
- Le reranking des résultats de la seconde extraction (en exploitant le corps des documents) est effectué avec la somme pondérée des scores de [Cohere Multilingual Reranker](#) et de NER (Named Entity Reranker). NER extrait les entités nommées du document (à l'aide de `huggingface`) et les utilise comme métadonnées (*ma lecture est à affiner ici*) pour calculer les vecteurs TF-IDF qui sont confrontés aux vecteurs denses avec cosinus.
- Le générateur utilisé est LLaMA3.

Evaluation et Explication

J'ai retenu 4 publications que je compte reprendre en profondeur plus tard (2 sur l'évaluation et 2 sur l'explication par perturbation):

- [Zhou & Mulhem & Schwab 2026] TempPerturb-Eval: On the Joint Effects of Internal Temperature and External
- [Park & Lee 2025] Investigating Language Preference of Multilingual RAG Systems
- [Zhou & Mulhem & Schwab 2025] Explicabilité par Perturbations pour les Systèmes RAG
- [Vast et al., 2025] Comprendre la Nature des Signaux de Correspondance dans les Modèles Neuronaux pour la RI

Prise en main

Je me suis lancée dans la prise en main du code fourni par [Tran et al., 2025] sur ce [github](#). Pour cela, je me suis remise sur la prise en main des outils de CaSciModOT. J'ai passé la journée de mardi sur la résolution de conflit de dépendances liés aux problèmes de disponibilités des librairies sur conda-forge. Il semble que deux librairies (langchain-community et langchain-classic) aient besoin d'une version différente d'un même dépendance (langchain-text-splitters). Je continue à investiguer cela dans les prochains jours. Par conséquent, je n'ai toujours pas fait tourner le code fourni sur le github à l'heure actuelle.

Après plusieurs tentatives de résolution des problèmes de mise en place de l'environnement, j'ai choisi de reprendre le code pour utiliser les versions récentes des librairies langchain (dont la structure a été refondue depuis la publication du [github RAG4historical-newspapers](#)). J'ai donc

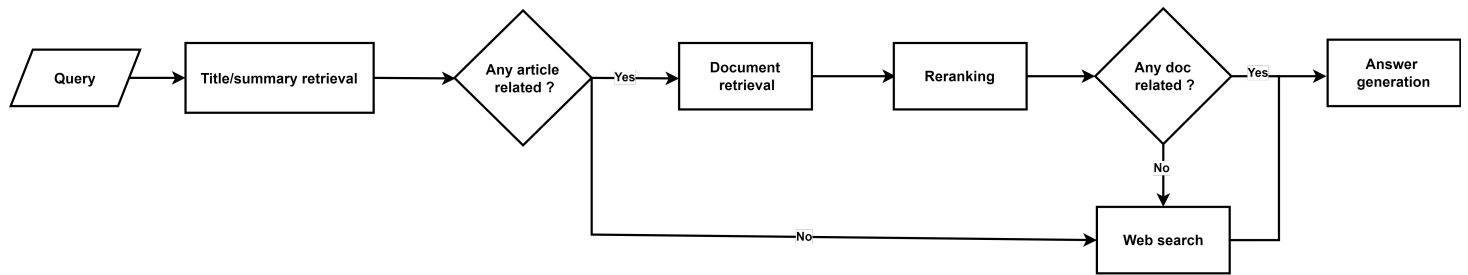
- mis de côté les imports non utilisés dans les notebooks et scripts python
- mis à jour la liste des librairies à importer
- déduit les librairies requises pour exécuter les documents
- repris la liste des librairies pour rester compatibles avec le catalogue forge-conda. Une partie des librairies est à installer via conda et une autre via pip. Les listes de librairies sont dans les fichiers `ragrights/ChallengeEvalLLM/req_rag4hn_conda.txt` et `ragrights/ChallengeEvalLLM/req_rag4hn_pip.txt`.
- installé le nouvel environnement sur la grappe de calcul

Je suis actuellement en train de vérifier que le code fonctionne toujours avec les modifications d'installation que j'ai effectuées. Cela permettra d'avoir une première quasi reproduction du modèle proposé par [Tran et al., 2025].

Je me suis permise d'écarter la partie websearch (voir image après) d'office car elle ne nous sert pas

et n'est pas commentée dans l'évaluation présentée dans la publication.

J'ai lancé une execution du script qui génère les données. Ce qui prendra de l'ordre de 4h d'après mes précédentes tentatives Je passerai les notebooks dessus dans la foulée.



En attendant, j'ai commencé à reprendre plus sérieusement les données ciblées. Je compte

- repérer la structure des données
- en extraire (au moins une partie) pour les mettre au format pour l'usage de MechIR

Ce sera fini lundi en début de journée.