Actes de JEP-TALN-RECITAL 2024
Toulouse, France, 8 au 12 juillet 2024

Participation du CRIM à DEFT 2024 : Utilisation de petits modèles de Langue pour des QCMs dans le domaine médical

Ahmed Moubtahij[1] Charles-William Cummings[1] Azur Handan[1] Edith Galy[1] Eric Charton[1]
(1) CRIM, 101 – 405, avenue Ogilvy, H3N 1M3, Montréal (Québec), Canada
ahmed.moubtahij@crim.ca, charles-william.cummings@crim.ca, azur.handan@crim.ca, edith.galy@crim.ca, eric.charton@crim.ca

RÉSUMÉ

Ce papier décrit le travail de l'équipe du CRIM (Centre de recherche en Informatique de Montréal) dans le cadre du Défi Fouille de textes 2024. Nous présentons les expériences que nous avons menées dans le cadre de la tâche principale consistant à identifier automatiquement, pour une question donnée issue d'annales d'examens de pharmacie, l'ensemble des réponses correctes parmi les cinq proposées. La contrainte est d'utiliser un système de moins de 3 milliards de paramètres dont les données d'entraînement sont connues. Pour ce faire, nous avons testé des approches impliquant du *few-shot prompting*, du RAG, de l'affinage et de la génération contrainte en dernier recours.

ABSTRACT

CRIM's Participation in DEFT 2024 : Using Small Language Models in Medical MCQA

This paper describes the work of the CRIM team (Centre de recherche en Informatique de Montréal) as part of DEFT 2024. We present the experiments we conducted for the main task, which involves automatically identifying, for a given question from pharmacy exam archives, the set of correct answers among the five proposed. The constraint is to use a system with less than 3 billion parameters and known training data. To achieve this, we tested approaches involving few-shot prompting, RAG, fine-tuning and constrained generation fallback.

MOTS-CLÉS : LLM, few-shot prompting, RAG, génération contrainte en dernier recours.

KEYWORDS: LLM, few-shot prompting, RAG, Constrained Generation Fallback.

1 Introduction

La campagne d'évaluation DEFT 2024 porte sur la mise en place d'approches à base de grands modèles de langue (GML/LLM) permettant de répondre automatiquement à des questionnaires à choix multiples (QCM) issus d'annales d'examens de pharmacie. Ces questionnaires composent le corpus FrenchMedMCQA (Labrak *et al.*, 2022) qui était fourni aux participants pour leur permettre de tester et/ou entraîner leur système. FrenchMedMCQA comprend 3 105 QCMs, dont 1080 questions avec une réponse unique parmi les cinq possibles et 2025 avec de multiples réponses. La tâche principale consiste à répondre à une question posée en sélectionnant une ou plusieurs réponses parmi les 5 proposées. La contrainte est d'utiliser un système de moins de trois milliards de paramètres dont les sources d'entraînement sont connues (ce qui exclut ChatGPT ou Mistral par exemple).

Le recours à l'utilisation de ressources externes est limité à deux corpus supplémentaires : le corpus NACHOS (Labrak *et al.*, 2023) et Wikipedia. L'objectif des organisateurs étant de permettre aux participants d'explorer des techniques telles que le Retrieval-Augmented Generation (RAG) (Lewis *et al.*, 2021) .

Les métriques utilisées pour évaluer la tâche sont l'*Exact Match Ratio* (taux de réponses parfaitement justes) et le *Hamming Score* (taux de réponses justes parmi l'ensemble des réponses et la référence). L' EMR (*Exact Match Ratio*) a été utilisée comme métrique officielle pour classer les participants.

Dans cet article nous présentons les résultats de nos expériences sur la tâche principale qui s'est déroulée en deux temps : une première phase d'exploration avec le corpus d'évaluation issu de FrenchMedMCQA, puis une soumission avec 3 systèmes sur un corpus d'évaluation.

Au travers de ces expériences, nous avons été amenés à nous poser différentes questions :

— Quelles techniques (entre le RAG, l'affinage, le *few-shot prompting*, etc.) peuvent pallier la contrainte de taille des modèles ?
— Quelle est la performance des contraintes de grammaire à la génération dans une tâche QCM ?
— Quelles sont les performances d'un LLM (*large langage model*) entraîné uniquement sur de l'anglais comparé à un LLM multilingue sur cette tâche en français ?

2 Méthodologie

Les contraintes sur la taille du modèle et la source des données d'entraînement ont considérablement réduit le spectre des modèles utilisables.

2.1 Description des modèles

Nous avons porté notre choix sur trois modèles à tester : Bloomz-3B, stablelm-3b-4e1t et Flan-T5-xl.

- `Bloomz-3B` (Muennighoff *et al.*, 2022) [1] est un LLM auto-régressif basé sur une architecture de transformeur (décodeur) de 3 milliards de paramètres. Ses données d'entraînement sont multilingues (45 langues) et connues puisqu'il a été développé dans le cadre de Big Science, un projet collaboratif international. Bloomz-3B a été affiné pour répondre à des tâches de compréhension d'instructions sans exemple de la réponse attendue dans le prompt (*zero-shot*). Dans leur article, Muennighoff *et al.* (2022) montrent que le modèle est capable de généraliser les tâches à des langues qu'il n'a pas vu intentionnellement pendant son entraînement et qu'il performe sans surprise encore mieux sur des langues qui y étaient présentes. Le français est présent à 13,5% dans le corpus d'entraînement (contre 31,3 % pour l'anglais).

- `stablelm-3b-4e1t` (Tow *et al.*, 2023) [2], développé par Stable AI, est aussi un LLM auto-régressif basé sur une architecture de transformeur (décodeur) de 3 milliards de paramètres. Il a été entraîné sur 1 trillion de jetons, en anglais. S'inspirant des travaux de Muennighoff *et al.* (2023), ce modèle a été développé avec l'intention d'obtenir des performances comparables à ceux entraînés sur plus de données, notamment, en entraînant plus longtemps sur 4 epochs.

1. https://huggingface.co/bigscience/bloomz-3b
2. https://huggingface.co/stabilityai/stablelm-3b-4e1t

- `Flan-T5-xl` (Chung *et al.*, 2022)[3] est un LLM basé sur l'architecture T5 (Text-to-Text Transfer Transformer) qui utilise une structure encodeur-décodeur. Il a été affiné avec des instructions supplémentaires pour couvrir de nombreuses tâches, notamment, des tâches de questions/réponses. Le modèle compte environ $2,85$ milliards de paramètres et supporte de nombreuses langues.

2.2 Hypothèses

Les expériences préliminaires ont consisté à utilisé du *few-shot prompting* avec ou sans RAG pour ces trois modèles, avec un essai d'affinage pour `stablelm-3b-4e1t`.

Nous avons fait l'hypothèse que malgré l'unilinguisme anglais du modèle `stablelm-3b-4e1t`, le *few-shot*, le RAG ou l'affinage permettront de transférer les capacités du modèle pour répondre aux questions en français telles que décrites dans l'article de Schäfer *et al.* (2024).

`Bloomz-3B` et `Flan-T5-xl` ont tous deux été entraînés pour répondre à des tâches dans le cadre du *zero-shot prompting* dans un contexte multilingue, en plus de contenir du français dans leur corpus d'entraînement. On peut donc espérer qu'ils performent correctement pour cette tâche.

Nous avons aussi testé l'affinage de `stablelm-3b-4e1t` avec le corpus FrenchMedMCQA, supposant que l'ajout de français et du domaine spécifique lui permettrait de gagner en performance. Nous voulions aussi comparer cette approche à celle du *few-shot prompting* sans affinage.

3 Système proposé

Pour mener à bien nos expérimentations, nous avons conçu un système composé d'un ensemble de tâches exécutées successivement. Les modules qui exécutent ces tâches ont les fonctions suivantes :

1. En premier lieu, un pré-traitement est appliqué sur les caractères encodés. Cette étape - classique dans des évaluations telles que celle proposée - est nécessaire pour limiter les erreurs et augmenter la fiabilité générale du système.
2. En second lieu, notre système injecte des exemples de paires de questions et réponses dans le prompt pour conditionner le modèle et mieux orienter son fonctionnement.
3. Puis, notre système utilise l'outil de recherche d'information Lucene pour identifier, dans les corpus autorisés, un extrait utilisable dans le prompt. Ce module est une application de la méthodologie *RAG (Retrieval Augmented Generation)*. Ce module est optionnel.
4. Les prompts formés à ce stade sont envoyés au système génératif pour qu'il produisent une réponse à la question qui lui est soumise.
5. Un ensemble d'expressions régulières permet d'identifier la réponse à la question posée dans le texte retourné par le système. Si aucune structure attendue n'est identifiée, la réponse est regénérée avec une contrainte de grammaire.
6. Dans la dernière étape, un dispositif de rétroaction permet de prédire si la réponse est considérée comme bonne ou doit être à nouveau recherchée.

Notre système est présenté dans la figure 1 et nous le décrivons maintenant en détails.

3. https://huggingface.co/google/flan-t5-xl

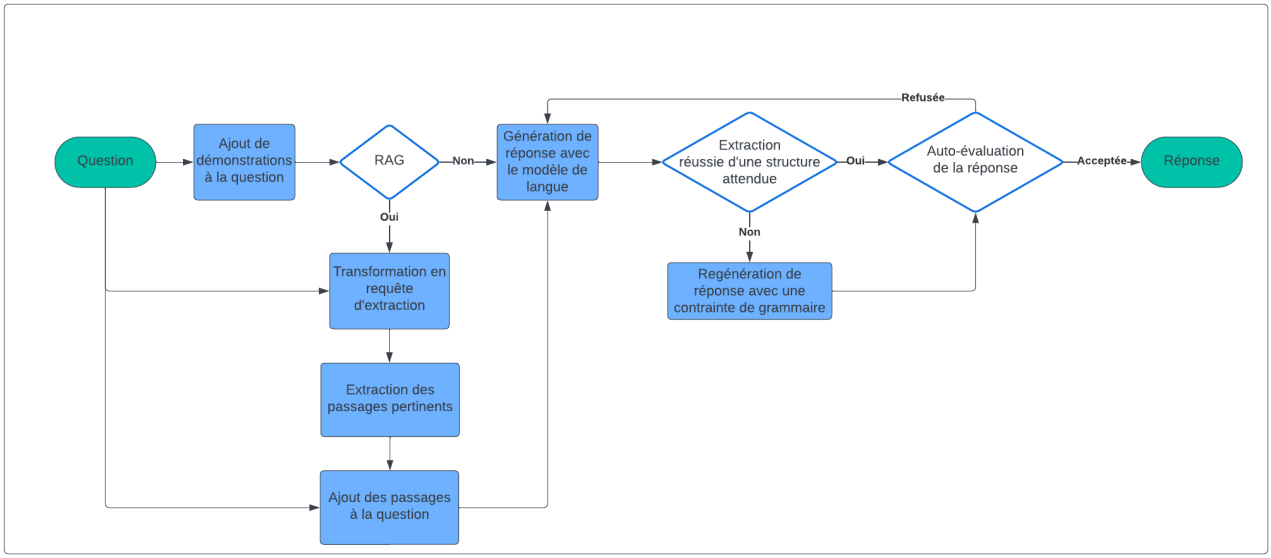

FIGURE 1 – Description de notre système. Il prends la forme d'un ensemble successif de modules qui en fin de traitement produisent une proposition de réponse. L'utilisation du module de RAG est optionnelle. Notre système est en mesure de s'auto-évaluer et de re-générer des réponses tant qu'un niveau de confiance dans les réponses proposées acceptable n'est pas atteint.

3.1 Pré-traitement

Les questions et les choix de réponses ont été encodées et décodées en unicode pour traiter les caractères accentués(ex : *s\u00e9cr\u00e9tion* →*sécrétion*). Il est courant, dans les campagnes d'évaluations telles que DEFT de prendre soin de la normalisation des contenus. Cette normalisation a pour but d'augmenter la robustesse et de réduire les erreurs de traitement qui pourraient être provoquées par l'apparition de données non comprises par le système. Sans surprise, notre module de pré-traitement, appliqué au fur à diverses étapes de préparation des prompts, a permis d'augmenter la performance sur la métrique EMR.

3.2 Conditionnement par ajout d'exemples

Nous avons injecté des exemples de questions-réponses dans le prompt pour conditionner le modèle avant de lui poser la question finale. Ces exemples sont sélectionnés dans le corpus d'entraînement de FrenchMedMCQA par le module `LabeledFewShot` de la librairie DSPy [4] (Khattab *et al.* (2023)). Ce module échantillonne des sous-ensembles de K exemples et sélectionne le sous-ensemble qui a le mieux performé selon la métrique donnée, en l'occurrence, EMR. L'hypothèse étant que si les données du corpus d'entraînement sont suffisamment représentatives, les meilleurs exemples de ce jeu de données permettront d'obtenir les meilleurs résultats dans le corpus de test.

4. https://github.com/stanfordnlp/dspy

3.3 Utilisation du RAG

Pour l'étape d'extraction de contexte, nous utilisons BM25, tel qu'implémenté par la librairie Lucene [5]. Étant donné que c'est un algorithme de recherche lexicale, nous adaptons la requête pour ne maintenir que les mots-clés en éliminant les mots vides. BM25 est choisi pour son efficacité accrue dans les domaines à terminologie spécialisée, en plus de ses moindres coûts en mémoire - contrairement à la recherche vectorielle - et en temps d'exécution.

Le nombre de jetons des passages extraits, additionné à celui de la question et des démonstrations doit être inférieur à la longueur de contexte d'entrée gérée par le LLM. Tant que ce n'est pas le cas, le dernier passage - en ordre descendant de pertinence à la question - extrait est écarté.

L'extraction cible les corpus NACHOS et Wikipedia, transformés tels que :

— NACHOS : segmenté en paires uniques de phrases composées d'au moins deux mots. Les phrases contenant uniquement des patrons de dates sont rejetées, étant donné qu'elles sont inutiles dans le domaine de questions en pharmacologie.

— Wikipedia : le sous-ensemble français de Wikipedia, en date du dépôt du 1er mars 2022, est chargé de la librairie *Datasets* [6] de HuggingFace. Les syntagmes nominaux des réponses dans la partition de validation du corpus FrenchMedMCQA sont utilisés pour ancrer les sujets des pages Wikipedia récupérées. On considère ces syntagmes (ex : "acide acétique") comme des titres potentiels de pages Wikipedia et on retient les pages ainsi titrées du corpus Wikipedia de HuggingFace. Le contenu de ces pages est ensuite segmenté par une fenêtre glissante de 1000 jetons avec 500 jetons de chevauchement. Les sections "Notes et références" des pages Wikipedia sont exclues.

3.4 Génération de la réponse avec le modèle de langue

Les prompts formés précédemment sont envoyés au système pour générer une réponse. Les figures 2 et 3 en annexe sont deux exemples de prompt possibles selon l'utilisation ou non du RAG.

Concernant les hyperparamètres, nous avons utilisé ceux fournis par le module *Text Generation Inference* [7] (TGI) de la librairie HuggingFace tel que suit :

— `temperature` : fixée à 0 étant donné la nature déterministe de la tâche QCM.

— `best_of` : Pour chaque prompt, on retient parmi deux prédictions celle qui totalise la plus haute probabilité de jetons générés. Ceci a pour effet de mitiger les prédictions aberrantes.

— `grammar` : On peut contraindre les jetons générés à adhérer à une expression régulière. Celle-ci exprime une liste de choix de réponse entre `a` et `e`.

3.5 Analyse de la structure attendue de la réponse

Nous avons observé que les réponses générées peuvent se présenter sous diverses formes telles que `[1, 3]`, `a, c` ou encore `a) Acide acétique, c) Des cercaires` au lieu de la structure attendue : `["a", "c"]`. Cette structure est ciblée pour s'aligner avec celle des démonstrations

5. https://github.com/apache/lucene
6. https://huggingface.co/datasets/wikimedia/wikipedia
7. https://huggingface.co/docs/huggingface_hub/main/en/package_reference/inference_client#huggingface_hub.InferenceClient.text_generation

échantillonnées du corpus FrenchMedMCQA.

Nous extrayons donc, à l'aide d'expressions régulières, les structures acceptables issues de la réponse générée et les corrigeons pour produire la réponse finale. Si aucune structure acceptable n'est identifiée dans la prédiction, le modèle est invité à la générer à nouveau en ajoutant cette fois une contrainte de grammaire lors de la génération. Ce garde-fou structurel assure la fiabilité de notre système pour la phase d'évaluation. Il n'est utilisé qu'en dernier recours afin de permettre la génération de jetons précurseurs d'assimilation et de décomposition du problème. Similairement au mécanisme du *Chain-of-Thought* (Wei *et al.*, 2023), il a été observé que de tels jetons améliorent souvent la qualité de la réponse. Nous désignons ce mécanisme *Constrained Generation Fallback* ou CGF.

3.6 Auto-évaluation de la réponse

Le modèle est optionnellement invité à évaluer sa propre réponse par rapport à la question et aux choix de réponses possibles. Si la prédiction est négativement évaluée, le modèle doit reproduire une réponse autre que la prédiction précédente. Cette boucle de rétroaction est réitérée jusqu'à ce que le système évolue positivement sa réponse.

Cette logique est implémentée avec le mécanisme d'assertion [8] de la librairie DSPy. En outre, la génération de l'évaluation est contrainte par l'option `grammar` de TGI à répondre `Yes` ou `No`. Un exemple de prompt intégrant cette fonctionnalité est fournie en annexe à la figure 4 .

4 Résultats préliminaires

Le tableau suivant résume nos expériences sur le corpus de validation fourni par les organisateurs en phase préliminaire. Pour le RAG, nous indiquons quel corpus a été utilisé (wikipedia ou NACHOS). La stratégie indique s'il y a eu un affinage sur le jeu de données d'entraînement de FrenchMedMCQA, le nombre d'exemples injectés dans le *few-shot prompting* et les techniques annexes décrites à la section 3.4.

Modèle	RAG(k=3)	Stratégie	EMR	Hamming
Flan-T5-XL	_	6-shot	12.86	40.69
Flan-T5-XL	wiki	6-shot	12.54	39.6
bloomz-3B	_	0-shot	8.52	19.72
bloomz-3B	NACHOS	0-shot	10.61	20.89
bloomz-3B	NACHOS	6-shot	13.34	23.06
stablelm-3b-4e1t	_	zero-shot	1.93	75.52
stablelm-3b-4e1t	wiki	zero-shot	1.13	99.00
stablelm-3b-4e1t	wiki	16-shot	11.25	35.76
stablelm-3b-4e1t	_	affinage-16shot-unicode-cgf	14.95	51.88
stablelm-3b-4e1t	_	16shot-unicode-cgf	15.76	50.04
stablelm-3b-4e1t	_	11shot-bestof2-autojudge-unicode-cgf	17.85	51.63

TABLE 1 – Résultats préliminaires sur le corpus de test FrenchMedQCA

8. https://dspy-docs.vercel.app/api/assertions

Les résultats obtenus sur le corpus de test du corpus FrenchMedMCQA sont parfois contre-intuitifs. L'expérience à base de *few-shot prompting* sur la base du modèle `stablelm-3b-4e1t` qui n'est entraîné qu'avec des données en langue anglaise, obtient de meilleurs résultats que les expériences avec de l'affinage en français, sur la tâche ciblée, avec le même modèle. Notre hypothèse est que du sur-apprentissage ou de l'oubli catastrophique (Aleixo *et al.*, 2023) puissent être les causes de ces résultats surprenants. Des contraintes de temps n'ont pas permis d'approfondir l'analyse.

Concernant l'utilisation du RAG, les résultats sont inégaux. `Bloomz-3B` a bénéficié du RAG sur du *zero-shot* avec le corpus NACHOS et encore plus de l'augmentation du nombre d'exemples (quand on passe à du *few-shot* avec 6 exemples). Mais ce n'est pas le cas de `Flan-T5-XL` pour qui l'ajout du RAG (avec Wikipedia), sur du *few-shot*, a dégradé légèrement les résultats. Quant à `stablelm-3b-4e1t`, on voit clairement qu'en *zero-shot* seul, les performances sont très mauvaises, et que le RAG les dégrade légèrement plus (comme pour Flan).

En revanche, l'augmentation du nombre d'exemples, couplée à l'utilisation du RAG, améliore les performances de quasiment 10 points. Il semble que le module de RAG n'en soit pas la cause puisque les performances sont encore meilleures lorsque d'autres techniques couplées au *few-shot* sont mises en oeuvre, tout en n'utilisant pas le RAG. Une rapide inspection visuelle nous laisse croire que la performance du module de *Retrieval* n'est pas en cause, car celui-ci avait une bonne capacité à retrouver les bons contextes pour augmenter le prompt. Nous en concluons que le modèle `stablelm-3b-4e1t` n'a simplement pas pu tirer avantage du contexte fourni par le RAG.

Finalement, les meilleures performance de `stablelm-3b-4e1t` par rapport aux autres modèles sont probablement liées plutôt à la longueur de la fenêtre contextuelle, deux fois plus grande que `Bloomz-3B` et `Flan-T5-XL` (4096 jetons versus 2048 pour Bloomz) qui nous a permis d'injecter plus d'exemples de *few-shot*. Cela a sûrement permis d'améliorer l'apprentissage en contexte (Brown *et al.*, 2020).

Sur la base de ces expériences préliminaires, nous avons donc choisi de soumettre notre système avec le modèle `stablelm-3b-4e1t` avec seulement du *few-shot* et différentes techniques de sélection de la réponse finale : la contrainte de grammaire en dernier recours, l'auto-évaluation et le *best_of_2*.

5 Résultats finaux et discussions

Suite aux résultats obtenus sur le corpus de validation fourni, les trois soumissions sur le corpus de test ont été faites avec le modèle `stablelm-3b-4e1t`. Pour les trois systèmes, nous avons utilisé le *few-shot prompting*, sans affinage du modèle de base. Seuls les exemples dans le *few-shot prompting* sont en français. Le reste du prompt est en anglais pour s'accorder avec la langue de pré-entraînement du modèle.

Les systèmes soumis ont tous bénéficié de la contrainte de grammaire de dernier recours, dénotée par le suffixe `-cgf` pour *constrained generation fallback*. En outre, ils ont bénéficié de démonstrations échantillonnées par le module `dspy.LabeledFewShot` et d'un décodage en unicode des questions/réponses en français. Ces systèmes diffèrent uniquement au niveau des fonctionnalités *auto-judge* et *best_of_2* décrites à la section 3.4.

Les résultats que nous avons obtenus figurent dans le tableau 2.

Les performances de nos systèmes sur les données d'évaluation externes sont inférieures à celles

1	stablelm-3b-4e1t	16shot-unicode-cgf	2.31	29.68
2	stablelm-3b-4e1t	11shot_auto-judge-unicode-cgf	**10.27**	**43.05**
3	stablelm-3b-4e1t	11shot_bestof2-unicode-cgf	9.22	40.28

TABLE 2 – Soumission à Deft 2024

obtenues sur la partition de test fournie pour le développement. En particulier pour la soumission 1 dont l'EMR est de 2.31, là où pour la même stratégie avec les corpus de test, nous avions obtenus un score EMR de 15.76.

Finalement, un peu moins d'exemples dans le *few-shot* et des techniques de sélection telles que *auto judge* ou *best of 2* permettent d'obtenir des scores d'EMR moins bonnes de 5 à 7 points par rapport à celles obtenues avec les mêmes techniques sur le corpus de validation de la phase exploratoires.

Ces résultats pointent vers un changement distributionnel significatif entre la partition de test fournie pour le corpus FrenchMedMCQA et celle utilisée par la campagne d'évaluation. Nous en concluons à ce stade - sous réserve d'investigations ultérieures - que les systèmes développés avec le modèle `stablelm-3b-4e1t` sont donc limités dans leur capacité de généralisation.

6 Conclusion

Les résultats obtenus dans notre étude révèlent plusieurs points intéressants et parfois contre-intuitifs quant à l'utilisation de différentes techniques pour répondre à des questions à choix multiples en français, avec des modèles de langue de taille limitée. Nos expériences ont montré que le *few-shot prompting* est la technique qui a été la plus efficace pour pallier la taille limitée des modèles plutôt que l'utilisation du RAG ou de l'affinage de modèle. C'est particulièrement le cas lorsque le modèle dispose d'une longue fenêtre contextuelle comme `stablelm-3b-4e1t`.

Nos expériences ont montré que la capacité multilingue d'un modèle n'est pas nécessairement un facteur déterminant pour obtenir de bonnes performances sur une tâche en français. Le modèle `stablelm-3b-4e1t`, uniquement entraîné en anglais, a montré des performances compétitives grâce à des techniques de *few-shot prompting*. Ceci suggère que la richesse et la qualité des exemples fournis dans le prompt peuvent compenser l'absence de données d'entraînement en français.

Au delà des grandes techniques (*few-shot*, RAG et affinage), notre principale contribution repose sur l'utilisation de la contrainte de génération en dernier recours après l'extraction et correction d'expressions régulières tolérées, et ce, afin ne pas brimer la génération de jetons précurseurs d'assimilation et de décomposition du problème avant de produire la réponse attendue.

Références

ALEIXO E. L., COLONNA J. G., CRISTO M. & FERNANDES E. (2023). Catastrophic forgetting in deep learning : A comprehensive taxonomy.

BROWN T. B., MANN B., RYDER N., SUBBIAH M., KAPLAN J., DHARIWAL P., NEELAKANTAN A., SHYAM P., SASTRY G., ASKELL A., AGARWAL S., HERBERT-VOSS A., KRUEGER G., HENIGHAN T., CHILD R., RAMESH A., ZIEGLER D. M., WU J., WINTER C., HESSE C., CHEN M., SIGLER E., LITWIN M., GRAY S., CHESS B., CLARK J., BERNER C., MCCANDLISH S., RADFORD A., SUTSKEVER I. & AMODEI D. (2020). Language models are few-shot learners.

CHUNG H. W., HOU L., LONGPRE S., ZOPH B., TAY Y., FEDUS W., LI Y., WANG X., DEHGHANI M., BRAHMA S., WEBSON A., GU S. S., DAI Z., SUZGUN M., CHEN X., CHOWDHERY A., CASTRO-ROS A., PELLAT M., ROBINSON K., VALTER D., NARANG S., MISHRA G., YU A., ZHAO V., HUANG Y., DAI A., YU H., PETROV S., CHI E. H., DEAN J., DEVLIN J., ROBERTS A., ZHOU D., LE Q. V. & WEI J. (2022). Scaling instruction-finetuned language models.

DIAS G., Éd. (2015). *Actes de TALN 2015 (Traitement automatique des langues naturelles)*, Caen. ATALA, HULTECH.

KHATTAB O., SINGHVI A., MAHESHWARI P., ZHANG Z., SANTHANAM K., VARDHAMANAN S., HAQ S., SHARMA A., JOSHI T. T., MOAZAM H., MILLER H., ZAHARIA M. & POTTS C. (2023). Dspy : Compiling declarative language model calls into self-improving pipelines. *arXiv preprint arXiv :2310.03714*.

LABRAK Y., BAZOGE A., DUFOUR R., DAILLE B., GOURRAUD P.-A., MORIN E. & ROUVIER M. (2022). FrenchMedMCQA : A French multiple-choice question answering dataset for medical domain. In A. LAVELLI, E. HOLDERNESS, A. JIMENO YEPES, A.-L. MINARD, J. PUSTEJOVSKY & F. RINALDI, Éds., *Proceedings of the 13th International Workshop on Health Text Mining and Information Analysis (LOUHI)*, p. 41–46, Abu Dhabi, United Arab Emirates (Hybrid) : Association for Computational Linguistics. DOI : 10.18653/v1/2022.louhi-1.5.

LABRAK Y., BAZOGE A., DUFOUR R., ROUVIER M., MORIN E., DAILLE B. & GOURRAUD P.-A. (2023). Drbert : A robust pre-trained model in french for biomedical and clinical domains.

LEWIS P., PEREZ E., PIKTUS A., PETRONI F., KARPUKHIN V., GOYAL N., KÜTTLER H., LEWIS M., TAU YIH W., ROCKTÄSCHEL T., RIEDEL S. & KIELA D. (2021). Retrieval-augmented generation for knowledge-intensive nlp tasks.

MUENNIGHOFF N., RUSH A. M., BARAK B., SCAO T. L., PIKTUS A., TAZI N., PYYSALO S., WOLF T. & RAFFEL C. (2023). Scaling data-constrained language models.

MUENNIGHOFF N., WANG T., SUTAWIKA L., ROBERTS A., BIDERMAN S., SCAO T. L., BARI M. S., SHEN S., YONG Z.-X., SCHOELKOPF H. *et al.* (2022). Crosslingual generalization through multitask finetuning. *arXiv preprint arXiv :2211.01786*.

SCHÄFER A., RAVFOGEL S., HOFMANN T., PIMENTEL T. & SCHLAG I. (2024). Language imbalance can boost cross-lingual generalisation.

TOW J., BELLAGENTE M., MAHAN D. & RIQUELME C. (2023). Stablelm 3b 4e1t.

WEI J., WANG X., SCHUURMANS D., BOSMA M., ICHTER B., XIA F., CHI E., LE Q. & ZHOU D. (2023). Chain-of-thought prompting elicits reasoning in large language models.

7 Annexes

```
You are a Pharmacology Expert.
Your task is to answer a Multiple Choice Question from a French Pharmacy Exam.
The following are Question-Answer examples from said exam:
---

Follow the following format.

Question: Question asking which choices correctly answer it.
Choices: Mapping of choices to answers where the keys range from letters 'a' to 'e' and
the values are the corresponding Answers.
Answer: List of correct choices with no duplicates.
---
[N demonstrations]
---
Question: Parmi les composés suivants retrouvés dans le métabolisme de l'éthanol,
lequel est considéré comme étant le métabolite toxique?

Choices: {"a": "éthanol", "b": "Acétaldéhyde", "c": "Acide acétique", "d": "Ester éthylique",
"e": "Acide lactique"}

Answer:
```

FIGURE 2 – Prompt pour la génération de réponses avec un modèle de langue sans RAG.

You are an Expert in Pharmacology, Biomedicine and Chemistry.
First, you will be provided with Question-Answer examples from a French Pharmacy Exam.
Second, you will be provided with Context passages that may be relevant to Answer the final Question.
Your task is to answer the Multiple Choice Question at the end.

Follow the following format.

Question: Question asking which choices correctly answer it.
Choices: Mapping of choices to answers where the keys range from letters 'a' to 'e' and the values are the corresponding Answers.
Context: May or may not contain relevant facts to Answer the Question.
Answer: List of correct choices with no duplicates.

[N demonstrations]

Question: Parmi les composés suivants retrouvés dans le métabolisme de l'éthanol, lequel est considéré. comme étant le métabolite toxique?

Choices: {"a": "éthanol", "b": "Acétaldéhyde", "c": "Acide acétique", "d": "Ester éthylique", "e": "Acide lactique"}

Context:
Acide acétique; Dans le corps humain, l'acide acétique est normalement produit après la consommation d'alcool : l'éthanol est converti en acétaldéhyde qui est alors converti en acide acétique sous l'influence de l'enzyme acétaldéhyde déshydrogénase et ensuite en acetyl-coA par la ligase acétate-CoA.

Answer:

FIGURE 3 – Prompt pour la génération de réponses avec un modèle de langue avec RAG.

```
Judge if the answer is correct based on the Question and the Choices.
---

Follow the following format.

Question: Question Asking which Choices Correctly Answer it.
Choices: Mapping of choices to answers where the keys range from letters 'a' to 'e'
and the values are the corresponding Answers.
Answer: List of correct choices with no duplicates.
Verdict: Is the Answer correct based on the Choices?

---

Verdict: Correct[Yes/No]:
```

FIGURE 4 – Prompt pour la génération de réponses avec auto-évaluation de la réponse