Actes de CORIA-TALN 2023
Paris, France, 5 au 9 juin 2023

LIS@DEFT'23 : les LLMs peuvent-ils répondre à des QCM ? (a) oui; (b) non; (c) je ne sais pas.

Benoit Favre[1]
(1) Aix Marseille Université, CNRS, LIS, Marseille, France. benoit.favre@lis-lab.fr

RÉSUMÉ

Cet article présente un ensemble d'expériences sur la tâche de réponse à des questions à choix multiples de DEFT 2023. Des grands modèles de langage sont amorcés avec les questions afin de collecter les réponses générées. Les résultats montrent que les modèles ouverts sans affinage obtiennent des performances similaires à celles d'un système supervisé fondé sur BERT, et que l'affinage sur les données de la tâche apporte des améliorations.

ABSTRACT

LIS@DEFT'23 : can LLMs answer MCQs ? (a) yes ; (b) no ; (c) I don't know.

This paper presents a set of experiments on the multiple-choice question answering task of DEFT 2023. Large language models are prompted with the questions and responses are collected from generated completions. Results show that open models without refinement achieve similar performance to that of a supervised system based on BERT, and that refinement on the task data brings improvements.

MOTS-CLÉS : Questions à choix multiples, DEFT, grands modèles de langage, Amorce, GPT, Affinage LoRA.

KEYWORDS: Multiple choice questions, DEFT, large language models, Prompt, GPT, LoRA finetuning.

1 Introduction

Les récents progrès en traitement automatique des langues ont montré que les grands modèles de langage (LLM, pour large language models) ont des propriétés de généralisation qui leur permettent d'adresser un grand nombre de tâches, y compris sans y avoir été soumis en entraînement, et ont la capacité de puiser dans les connaissances accumulées dans leurs données d'entraînement pour construire des raisonnements simples. Dans ce travail nous nous intéressons aux capacités de ces modèles pour répondre à des questions à choix multiples dans le domaine médical. Cet article présente en particulier le système proposé par le LIS [1] pour la tâche partagée DEFT 2023.

Nous tentons de répondre aux questions suivantes :

1. Quelles sont les différences de performances entre modèles ouverts sur la tâche de QCM ?
2. Quel est le lien entre taille des LLM et performances attendues ?
3. L'affinage des modèles à bas coût est-il bénéfique pour traiter la tâche ?

1. Code source sur https://gitlab.lis-lab.fr/benoit.favre/deft2023-llm

```
{
  "id": "e3e35ba581919533a9d7e75fa6437c201837f4cc6698c5bb2e7c8fd2580366f8",
  "question": "Parmi les propositions suivantes concernant le métabolisme
    du calcium, indiquer celles qui sont exactes.",
  "answers":{
    "a": "La majorité du calcium de l'organisme se trouve dans le plasma",
    "b": "L'hormone parathyroïdienne favorise la réabsorption tubulaire
      du calcium",
    "c": "La sécrétion de calcitonine est régulée par le calcium ionisé",
    "d": "La vitamine D favorise l'absorption intestinale du calcium",
    "e": "L'hormone parathyroïdienne inhibe l'action de la 1-alpha
      hydroxylase rénale"
  },
  "correct_answers": ["b","c","d"],
  "subject_name": "pharmacie",
  "type": "multiple",
  "nbr_correct_answers": 3
}
```


FIGURE 1 – Exemple d'instance du corpus DEFT 2023, contenant la question, les réponses possibles et les réponses correctes au format JSON.

L'article présente d'abord la tâche (section 2), il détaille ensuite l'approche et la méthode employée (section 3), puis présente les expériences et résultats associés (section 4), avant de conclure.

2 Réponse à des QCM

2.1 Travaux reliés

La tâche de question-réponse est une tâche emblématique du TAL et a été traitée principalement sous trois formes : la génération de réponses à des questions ouvertes (*question answering*), souvent des questions de connaissances générales, la localisation de réponses dans un texte (*machine reading comprehension*), et la sélection de réponses à des questions à choix multiples. De nombreuses revues de littérature présentent les avancées dans le domaine (Soares & Parreiras, 2020; Allam & Haggag, 2012; Hao *et al.*, 2022; Bouziane *et al.*, 2015).

Les QCM se présentent sous la forme d'une question suivie d'un certain nombre de réponses possibles, dont un sous-ensemble peut être correct. Les méthodes développées par la communauté pour répondre automatiquement à des QCM ont suivi les avancées en TAL. Par exemple, la famille de réseaux de neurones pré-entraînés BERT a été exploitée en mettant en compétition plusieurs encodeurs qui prennent en entrée la question et un choix possible et génèrent une réponse binaire, des entrées parfois complétées par un contexte trouvé par recherche d'information dans un corpus de connaissances du domaine cible (Pal *et al.*, 2022; Roy *et al.*, 2021; Labrak *et al.*, 2022; Le Berre & Langlais, 2020). La génération précédente d'approches exploitait les réseaux de neurones récurrents comme les LSTM dans des conditions similaires (Guo *et al.*, 2017).

Les grands modèles de langage, après affinage sur des instructions correspondant à des tâches variées, sont devenus une méthode versatile pour traiter bon nombre de tâches de TAL dont la réponse à des

QCM. Ils offrent des performances prometteuses dans le domaine médical (Nori *et al.*, 2023), mais les performances ne sont pas toujours exceptionnelles, comme par exemple dans le cas de questions sur du code source, un type de contenu peu présent dans les données de pré-entraînement (Savelka *et al.*, 2023).

2.2 La tâche DEFT 2023

La campagne d'évaluation DEFT 2023 consiste en la génération automatique de réponses à des QCM provenant d'annales d'examens de pharmacie. Elle tire parti du corpus FrenchMedMCQA (Labrak *et al.*, 2022), calqué sur le corpus MedMCQA en anglais (Pal *et al.*, 2022). Ce corpus contient 3105 questions accompagnées de 5 choix possibles, dont au moins un est vrai. Un exemple est donné dans la figure 1. Ce corpus est divisé en entraînement (2171 instances), validation (312) et test (622). Deux tâches sont proposées : répondre aux questions (principale) , et déterminer le nombre de réponses correctes (annexe). Nous avons participé à la première tâche et dérivé des sorties pour la seconde à partir de celles de la première. Pour la tâche principale, les performances des systèmes sont évaluées avec deux métriques : la correspondance exacte entre les réponses produites (*exact match ratio*, EMR), et la distance de hamming entre les réponses produites et les réponses de référence (*Hamming score*). Nous rapportons dans cet article principalement la métrique EMR qui nous apparaît canonique pour la tâche. La tâche annexe est évaluée selon le taux de bonne classification sur le nombre de réponses.

3 Approche : amorçage et affinage de grands modèles de langage

3.1 Modèles et affinage

Les modèles de langage autoregressifs sont fondés sur l'idée de calculer la probabilité d'un texte étant donnée la langue, ou à défaut étant donné un corpus d'apprentissage. Cette probabilité peut être marginalisée sur les mots selon l'ordre classique de factorisation du début à la fin du texte, puis la probabilité d'un mot étant donné son historique est en général approximée par un contexte de taille fixe. Depuis l'avènement des n-grammes jusqu'aux transformers, les modèles de langage sont donc entraînés à prédire le mot suivant étant donné un contexte. Afin de limiter le nombre d'unités lexicales, les modèles les plus récents travaillent sur des tokens (mots ou facteurs de mots) plutôt que les mots eux-mêmes.

Depuis GPT (Brown *et al.*, 2020), l'architecture la plus courante pour les modèles de langage est le transformer. Il s'agit d'un empilement de couches comprenant un mécanisme d'attention multi-tête permettant de capturer les interactions entre les paires de mots (paires d'interactions, paires de paires, etc. quand on monte dans les couches). Ces modèles peuvent être implémentés de manière relativement efficace sur GPU et sont entraînés par rétro-propagation. Ils peuvent être exploités par amorçage (donner le début d'un texte), puis génération token par token d'une suite, les amorces étant choisies pour que la génération corresponde à une tâche de TAL.

Lorsque l'on augmente le nombre de paramètres des modèles de langage, leur entraînement fait émerger des propriétés remarquables, comme la spécialisation de têtes d'attention à des phénomènes linguistiques connus. À partir de plusieurs milliards de paramètres, les grands modèles de langage (LLM, *large language models*) peuvent faire des tâches sans avoir été spécialisés dessus comme

répondre à des questions, traduire ou résumer un texte, et leur comportement peut être calibré à cet effet en les affinant sur de nombreux exemples d'instructions et réponse attendue. Les plus grands modèles (BLOOM, GPT-3, OPT, PaLM...), avec plusieurs centaines de milliards de paramètres (175 pour GPT-3, 540 pour PaLM), nécessitent une infrastructure spécifique en entraînement et en inférence, avec de nombreux noeuds de calcul équipés chacun de plusieurs GPU de dernière génération.

La série de modèles LLaMa (Touvron *et al.*, 2023) a été rendue publique sous une licence permettant la recherche. Ces modèles, fondés sur l'architecture transformers avec quelques innovations proposées dans GPT-3 et PaLM comme les rotary embeddings (Su *et al.*, 2021), sont disponibles en plusieurs tailles : 7 milliards (7B), 13 milliards (13B), 30 milliards (30B) et 65 milliards (65B) de paramètres. Ils sont entraînés sur un jeu de données constitué de Common Crawl, C4 (variante de pré-traitement de Common Crawl), Wikipédia, Github, ArXiv, Books et StackExchange. Les corpus anglophones sont filtrés pour conserver en majorité cette langue, alors que Wikipédia contient explicitement 20 langues dont le français. Les plus petits modèles sont entraînés sur 10^{12} tokens, alors que les plus gros le sont sur 1.4×10^{12} tokens, suivant les observations de passage à l'échelle des LLM (Kaplan *et al.*, 2020). Tous ces modèles prennent en compte un historique maximum de 2048 tokens. Leurs paramètres exacts peuvent être trouvés dans l'article original.

L'affinage des LLM demandant beaucoup de mémoire GPU pour conserver l'état de l'optimiseur ainsi que les activations intermédiaires des modèles, plusieurs techniques d'adaptation à moindre coût ont été proposées. L'adaptation à faible rang (LoRA, *Low Rank Adaptation*) repose sur l'idée intuitive que l'ensemble des paramètres n'a pas besoin d'être modifié lors d'un affinage avec relativement peu d'exemples (Hu *et al.*, 2021). Pour cela, on considère le modèle original comme gelé, puis on ajoute à chaque matrice de paramètres une matrice de faible rang calculée comme le produit de deux petites matrices. Le nombre de paramètres ajoutés au modèle est faible et la mémoire requise est raisonnable. Conjuguée à un stockage des paramètres du modèle d'origine sur 8 bits ou moins, cette technique permet d'affiner un modèle 65B sur un seul GPU A100 à 80G de mémoire.

3.2 Amorce et extraction de réponse

Dans les expériences présentées dans la suite, nous exploitons une amorce simple dans laquelle sont introduites la question et les réponses possibles, telle que présentée dans la figure 2. Cette amorce est constituée d'une description de la tâche donnant le contexte et une contrainte sur le format de sortie, puis de la question et de la liste des cinq réponses possibles, préfixées par des lettres entre parenthèses, et enfin suivies d'une sollicitation de la réponse, contenant selon les modèles une parenthèse ouvrante pour les forcer à présenter les réponses sous formes de lettres. Cette amorce a été sélectionnée parmi plusieurs candidates prometteuses mentionnant par exemple un "corrigé des épreuves de pharmacie" comme contexte, ou prenant la forme d'une mise en scène de dialogue entre deux interlocuteurs, l'un spécialiste du domaine et l'autre posant des questions, ou encore avec une description en anglais de la tâche comme c'est le cas dans les instructions de BLOOMz (Muennighoff *et al.*, 2022). Ces variantes n'ont pas donné de meilleurs résultats que l'amorce présentée ici lors de tests préliminaires, elle est donc exploitée dans l'ensemble des expériences.

Malgré les contraintes, la réponse d'un modèle de langage à ce type d'amorce peut être variée en termes de formattage et de contenu (copier ou non les réponses choisies, utiliser ou non les parenthèses autour des lettres, utiliser ou non la numérotation des réponses, développer ou non des explications ou un raisonnement avant de donner la réponse). Afin d'extraire la ou les réponses à une question,

Ceci est une question de QCM de l'examen de pharmacie. Réponds avec la ou les lettres correspondant à la bonne réponse.

La diminution d'une unité pH correspond à une concentration en H+ :
(a) 2 fois plus forte.
(b) 10 fois plus faible.
(c) 10 fois plus forte.
(d) 100 fois plus forte.
(e) 100 fois plus faible.
Réponse(s) : (

FIGURE 2 – Exemple d'amorce pour une question de QCM du corpus de développement de DEFT 2023. L'amorce est constituée d'une description de la tâche (bleu), de la question, d'une liste de réponses possibles (magenta), et d'un incitateur (marron) suivi éventuellement une parenthèse ouvrante forçant le début de la réponse (rouge).

nous avons développé une stratégie simple : (1) supprimer tout le texte correspondant à une répétition du début de l'amorce car nombre de modèles continuent de générer une liste de questions/réponses après avoir donné leur réponse, (2) détecter tous les caractères uniques (a-e) entre parenthèses, et (3) si (2) ne renvoie aucune réponse, détecter tous les caractères uniques (a-e) sans parenthèses. Pour les modèles sans affinage, nous exploitons les 32 premiers tokens générés après l'amorce.

Les modèles avec affinage reposent sur le même motif d'amorce et la même stratégie d'extraction de réponses, mais ils sont supervisés avec des réponses qui mentionnent explicitement la lettre et le texte des choix corrects, afin que les modèles associent explicitement les réponses aux choix possibles, et ne se contentent pas de capturer les régularités dans les lettres associés aux choix. La figure 3 montre le format d'une instance pour laquelle le modèle a généré plusieurs choix de réponses. Les modèles affinés sont limités à la génération de 128 tokens.

4 Expériences et résultats

Une première série d'expériences teste les compétences sur la tâche d'une série de modèles disponibles pour la recherche sur le hub huggingface [2]. Tous les modèles testés sont des modèles "instruits", qui ont été préalablement affinés sur des jeux de triplets (instruction, entrée et réponse attendue) à réaliser les tâches décrites en amorce plutôt que générer du texte ressemblant à leur corpus d'entraînement. Malgré le fait que le format d'instruction recommandé pour chaque modèle puisse être différent, nous avons conservé l'amorce décrite ci-avant. La plupart des modèles sont entraînés sur des corpus majoritairement en anglais, mais contenant du français, et affinés avec des instructions en anglais. Le tableau 1 présente les résultats en termes d'EMR sur l'ensemble de développement de DEFT 2023 pour les modèles : BLOOMz (Muennighoff *et al.*, 2022), issu du projet BigScience et affiné sur environ 80 millions d'instructions des corpus xP3/xP3-mt synthétisées à partir de corpus de TAL existants, Flan-T5 (Chung *et al.*, 2022) et Flan-UL2 (Tay, 2023), deux modèles affinés sur environ

2. huggingface.co/models/

Ceci est une question de QCM de l'examen de pharmacie. Réponds avec la ou les lettres correspondant à la bonne réponse.

Les complications d'une hépatite virale aiguë peuvent être à plus ou moins long terme :
(a) Une lithiase vésiculaire.
(b) Une hépatite chronique.
(c) Un cancer du foie.
(d) Une cirrhose.
(e) Une pancréatite aiguë.
Réponse(s) : (b) Une hépatite chronique ; (c) Un cancer du foie ; (d) Une cirrhose.

FIGURE 3 – Exemple de sortie sur le test de DEFT 2023 pour le système LLaMa-65B affiné sur les données d'entraînement. La partie générée par le système est explicitée en bleu : les réponses sont prefixées de leur lettre entre parenthèses, séparées par un point virgule, et terminées par une fin de ligne.

15 millions d'instructions multilingues et variées, Tk-Instruct (Wang *et al.*, 2022) un modèle T5 affiné avec des instructions naturelles collectées auprès d'humains, Pythia affiné sur les instructions naturelles d'OpenAssistant (Biderman *et al.*, 2023) ciblant le cas d'utilisation d'un assistant, OPT-IML (Iyer *et al.*, 2022) un modèle affiné sur 2000 tâches de TAL, Galactica (Taylor *et al.*, 2022) un modèle entraîné majoritairement sur des publications scientifiques, MPT (Team, 2023) un modèle ouvert reproduisant LLaMa et instruit sur de petits jeux d'instructions naturelles, PMC-LLaMa (Wu *et al.*, 2023) une version de LLaMa affinée sur des articles médicaux. Ces modèles ont été sélectionnés pour leur pertinence potentielle à la tâche, selon plusieurs tailles représentatives, et avec la contrainte de fonctionner sur un seul GPU. On observe les tendances suivantes dans ces résultats : dans une famille de modèles, la taille est corrélée avec les performances ; certains modèles ont des performances bien supérieures à taille égale, possiblement à cause d'une meilleure représentation du français ; les modèles entraînés spécifiquement sur des données médicales ou scientifiques n'offrent pas d'avantage par rapport aux modèles génériques. On observe aussi que les meilleures performances obtenues par tk-instruct-11b dans un contexte zero-shot sont proches de celles rapportées par Labrak et al. sur le corpus de test de DEFT'23 avec des systèmes supervisés fondés sur BERT/BART dont les instances sont augmentées par un contexte provenant d'articles scientifiques (Labrak *et al.*, 2022).

Une seconde série d'expériences présente des résultats à partir de la famille de modèles LLaMa (Touvron *et al.*, 2023) avec ou sans affinage. Dans ces expériences, les paramètres des modèles LLaMa sont convertis en 8 bits afin d'occuper moins de mémoire GPU, puis ils sont affinés pendant une époque avec la méthode LoRA décrite plus haut, et avec comme paramètres $R = 4$ (rang des matrices de paramètres), $\alpha = 16$, un dropout de 0.05, une taille de batch de 24, un taux d'apprentissage de 3×10^{-4}, une longueur maximale de 256 tokens pour le mécanisme d'attention, et une optimisation par la méthode Adam avec un warmup de 5% des batches. Seules les matrices de paramètres de projection de la clé et la valeur du mécanisme d'attention sont modifiées par LoRA (`q_proj` et `v_proj` dans le modèle). La taille de micro-batches est ajustée pour maximiser la vitesse d'entraînement ; le modèle 65B peut être entraîné sur un GPU A100 avec 80GB de RAM avec une taille de micro-batch de 1. Les modèles sont affinés selon quatre jeux d'entraînement : les 2171 instances d'entraînement de DEFT 2023 qui correspondent exactement à la tâche ciblée, le jeu de données

Modèle	Taille	EMR
bloomz-560m	0.5B	0.0737
bloomz-3b	3B	0.1442
bloomz-7b1	7.1B	0.1602
bloomz-7b1-mt	7.1B	0.1762
flan-t5-xxl-11b	11B	0.1794
flan-ul2-20b	20B	0.1570
tk-instruct-3b-def	3B	0.1346
tk-instruct-11b-def	11B	0.1826
oasst-sft-1-pythia-12b	12B	0.0705
opt-iml-1.3b	1.3B	0.0673
opt-iml-30b	30B	0.1442
mpt-instruct-7b	7B	0.0641
galactica-6.7b	6.7B	0.0352
pmc-llama-7b	7B	0.0224

TABLE 1 – Performances des modèles huggingface affinés sur des instructions non spécifiques à la tâche. Les performances sont données en termes de réponses exactes sur le jeu de développement.

Alpaca (Taori *et al.*, 2023) contenant 52k instructions génériques générées à partir de ChatGPT, le jeu de données Alpaca-fr, traduit par nos soins en français de manière automatique à l'aide du modèle nllb-200-distilled-1.3B [3], et le jeu de données Vicuna (Chiang *et al.*, 2023) contenant 70k dialogues sélectionnés par des utilisateurs de ChatGPT pour leur pertinence. La génération est effectuée sur un maximum de 128 tokens, avec une température de 0.1 et en utilisant 4 faisceaux, exploitant les 40 meilleurs candidats par token avec une masse de probabilité totale minimale de 0.75.

On observe dans les résultats du tableau 2 que : la représentation en (int8) diminue les performances par rapport à la représentation d'origine (fp16), les performances augmentent avec la taille des modèles, les modèles de langage génériques de grande taille offrent des performances similaires aux modèles affinés de petite taille, l'affinage sur des instructions est bénéfique par rapport à un modèle générique de même type et taille, et l'affinage sur les données de la tâche cible permettent des gains substantiels par rapport à un affinage générique. On remarque aussi que les performances après affinage sur Alpaca-fr ne sont pas différentes de celles d'un modèle affiné sur de l'anglais, suggérant que les LLM entraînés sur une petite partie d'une autre langue sont déjà capables de traitements multilingues. Nous n'avons pas beaucoup expérimenté avec cette capacité, et il serait souhaitable de mieux la cerner dans des travaux futurs.

Une troisième série d'expériences explore cette fois les performances de modèles fermés mis à disposition par les industriels à travers des API. Nous testons, toujours avec le même schéma d'amorce, les modèles des entreprises Cohere [4], AI21 [5] et OpenAI [6]. Les détails techniques des modèles associés ne sont pas toujours disponibles, les poids des modèles ne sont pas accessibles, et certains modèles ne sont plus accessibles au public depuis que nous les avons utilisés. Nous avons testé

3. Traduction automatique de la structure json complète de chaque instance pour conserver le contexte, aboutissant à 48k instructions après filtrage des instances mal formées à l'issue de la traduction.
4. https://cohere.com/
5. https://www.ai21.com
6. https://openai.com/

Modèle	Taille	Affinage	EMR
llama (int8)	7B	-	0.0576
llama (int8)	7B	alpaca	0.1217
llama (int8)	7B	alpaca-fr	0.1185
llama (int8)	7B	deft	0.1378
llama (int8)	13B	-	0.0769
llama (int8)	13B	alpaca	0.1474
llama (int8)	13B	vicuna	0.1538
llama (int8)	13B	deft	0.1730
llama (int8)	30B	-	0.1442
llama (fp16)	30B	-	0.1891
llama (int8)	30B	alpaca	0.1923
llama (int8)	30B	deft	0.2467
llama (int8)	65B	-	0.1730
llama (fp16)	65B	-	0.2179
llama (int8)	65B	deft	0.3044

TABLE 2 – Performances des variantes du modèles LLaMa en termes de réponses exactes, sur le jeu de développement.

le modèle command-xlarge-beta [7] de Cohere, le modèle j1-jumbo de A2AI, et code-cushman-001[6], code-davinci-002[6], text-curie-001, text-davinci-003, gpt-3.5-turbo-0301 (ChatGPT), et gpt-4-0314 (GPT-4) de OpenAI. Dans ces expériences, aucun effort n'a été fait pour adapter les amorces ou l'analyse de la réponse aux modèles. Le coût associé à l'utilisation de ces API est non nul, mais reste raisonnable, avec par exemple un coût de 2 USD pour faire tourner l'inférence de GPT-4 sur le test de DEFT2023, étant donné un tarif entre 0.03 et 0.06 USD pour 1000 tokens selon qu'ils appartiennent à l'amorce ou à la partie générée.

Les résultats sur les API fermées donnés dans le tableau 3 montrent que les performances en termes d'EMR varient beaucoup, avec des modèles aux performances similaires à celles des modèles publics, et d'autres modèles aux performances bien supérieures. Il est intéressant de constater par exemple

7. Modèle qui n'est plus disponible à l'heure actuelle

Fournisseur	Modèle	EMR
cohere	command-xlarge-beta	0.1057
ai21	j1-jumbo	0.0833
openai	code-cushman-001	0.1121
openai	code-davinci-002	0.3108
openai	text-curie-001	0.1217
openai	text-davinci-003	0.2884
openai	gpt-3.5-turbo-0301	0.4551
openai	gpt-4-0314	0.7788

TABLE 3 – Performances des modèles fermés disponibles à travers une API, sur l'ensemble de développement et en termes de réponses exactes. Certains modèles ne sont plus ouverts au public.

que code-davinci-002 obtient des performances similaires à LLaMa 65B affiné alors qu'il a été entraîné principalement sur du code source. Les résultats obtenus par GPT-4 sont similaires à ceux obtenus sur le jeu de données MedMCQA en anglais (Nori *et al.*, 2023) par le même modèle. Il n'est malheureusement pas possible de tirer de conclusions scientifiques de ces expériences car les détails des modèles, leurs données d'entraînement, etc. ne sont pas publiques.

		Tâche principale		Tâche annexe	
Nom du système	**Repro.**	**Hamming**	**EMR**	**F1-Score**	**Accuracy**
LIS/llama-65b-lora	✓	52.94	33.76	42.42	68.65
LIS/llama-30b-lora	✓	47.43	27.81	35.26	65.92
LIS/llama-13b-lora	✓	35.93	17.85	34.52	65.11
LIS/gpt-3.5-turbo-0301_prompt0		64.75	46.95	47.51	68.17
LIS/gpt-4-0314_prompt0		85.17	72.83	71.57	79.58

TABLE 4 – Résultats officiels sur le test de DEFT 2023 pour les systèmes soumis dans les deux tâches. Les systèmes reproductibles sont dénotés par ✓.

Enfin, le tableau 4 liste les résultats officiels selon toutes les métriques sur le test pour les systèmes soumis dans la catégorie *reproductible* et dans la catégorie *sans restriction*. Les systèmes reproductibles sont basés sur le modèle LLaMa affiné sur les données d'entraînement DEFT comme indiqué plus haut. Seuls deux systèmes sans limites ont été lancés sur le test : ChatGPT et GPT-4. Les résultats en termes d'EMR sont proches de ceux obtenus sur l'ensemble de développement pour tous les systèmes. On peut remarquer que sur la tâche annexe, de grandes différences en EMR n'impliquent pas de grandes différences en précision ou F-score, sauf pour GPT-4 qui semble être plus précis dans sa compréhension du nombre de réponses attendues.

5 Conclusion et perspectives

Les expériences montrent que certaines familles de modèles ont des performances très différentes à taille similaire, ce qui pourrait être expliqué par les compétences en français des modèles, et donc par la quantité de données en français sur lesquelles ils ont été entraînés. Nous observons aussi que la taille des modèles, à travers une famille, est un bon prédicteur des performances. Ceci peut s'expliquer par une meilleure compréhension du texte et une meilleure fidélité des connaissances représentées dans les modèles. Toutefois, une autre explication pourrait provenir de la présence des questions et réponses du corpus dans les données d'apprentissage des modèles. Il serait intéressant de tester cette hypothèse avec la méthode MELD (Nori *et al.*, 2023) qui détermine si une question a été mémorisée par un modèle en générant la fin de la question à partir de son début avec une température de 0 et en la comparant à l'original. Une autre façon de faire serait de retrouver les questions dans les données d'apprentissage des modèles.

D'autres perspectives pour améliorer les systèmes seraient d'explorer l'espace des amorces afin de comprendre les caractéristiques d'une amorce associées à la tâche, de donner des exemples de réponses attendues (few-shot learning), de générer un contexte à partir de connaissances externes comme des ouvrages du domaine, de faire des ensembles à partir de modèles initialisés différemment. D'un point de vue des ressources nécessaires pour affiner les grands modèles, des méthodes alternatives à LoRA devraient être étudiées.

Références

ALLAM A. M. N. & HAGGAG M. H. (2012). The question answering systems : A survey. *International Journal of Research and Reviews in Information Sciences (IJRRIS)*, **2**(3).

BIDERMAN S., SCHOELKOPF H., ANTHONY Q., BRADLEY H., O'BRIEN K., HALLAHAN E., KHAN M. A., PUROHIT S., PRASHANTH U. S., RAFF E., SKOWRON A., SUTAWIKA L. & VAN DER WAL O. (2023). Pythia : A suite for analyzing large language models across training and scaling.

BOUZIANE A., BOUCHIHA D., DOUMI N. & MALKI M. (2015). Question answering systems : survey and trends. *Procedia Computer Science*, **73**, 366–375.

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.

CHIANG W.-L., LI Z., LIN Z., SHENG Y., WU Z., ZHANG H., ZHENG L., ZHUANG S., ZHUANG Y., GONZALEZ J. E., STOICA I. & XING E. P. (2023). Vicuna : An open-source chatbot impressing gpt-4 with 90%* chatgpt quality.

CHUNG H. W., HOU L., LONGPRE S., ZOPH B., TAY Y., FEDUS W., LI E., WANG X., DEHGHANI M., BRAHMA S., WEBSON A., GU S. S., DAI Z., SUZGUN M., CHEN X., CHOWDHERY A., 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. DOI : 10.48550/ARXIV.2210.11416.

GUO S., LIU K., HE S., LIU C., ZHAO J. & WEI Z. (2017). IJCNLP-2017 task 5 : Multi-choice question answering in examinations. In *Proceedings of the IJCNLP 2017, Shared Tasks*, p. 34–40, Taipei, Taiwan : Asian Federation of Natural Language Processing.

HAO T., LI X., HE Y., WANG F. L. & QU Y. (2022). Recent progress in leveraging deep learning methods for question answering. *Neural Computing and Applications*, p. 1–19.

HU E., SHEN Y., WALLIS P., ALLEN-ZHU Z., LI Y., WANG L. & CHEN W. (2021). Lora : Low-rank adaptation of large language models.

IYER S., LIN X. V., PASUNURU R., MIHAYLOV T., SIMIG D., YU P., SHUSTER K., WANG T., LIU Q., KOURA P. S. *et al.* (2022). Opt-iml : Scaling language model instruction meta learning through the lens of generalization.

KAPLAN J., MCCANDLISH S., HENIGHAN T., BROWN T. B., CHESS B., CHILD R., GRAY S., RADFORD A., WU J. & AMODEI D. (2020). Scaling laws for neural language models. *arXiv preprint arXiv :2001.08361*.

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 *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.

LE BERRE G. & LANGLAIS P. (2020). Attending knowledge facts with bert-like models in question-answering : Disappointing results and some explanations. In *Advances in Artificial Intelligence : 33rd Canadian Conference on Artificial Intelligence, Canadian AI 2020, Ottawa, ON, Canada, May 13–15, 2020, Proceedings 33*, p. 356–367 : Springer.

MUENNIGHOFF N., WANG T., SUTAWIKA L., ROBERTS A., BIDERMAN S., SCAO T. L., BARI M. S., SHEN S., YONG Z.-X., SCHOELKOPF H., TANG X., RADEV D., AJI A. F., ALMUBARAK K., ALBANIE S., ALYAFEAI Z., WEBSON A., RAFF E. & RAFFEL C. (2022). Crosslingual generalization through multitask finetuning.

NORI H., KING N., MCKINNEY S. M., CARIGNAN D. & HORVITZ E. (2023). Capabilities of gpt-4 on medical challenge problems. *arXiv preprint arXiv :2303.13375*.

PAL A., UMAPATHI L. K. & SANKARASUBBU M. (2022). Medmcqa : A large-scale multi-subject multi-choice dataset for medical domain question answering. In *Conference on Health, Inference, and Learning*, p. 248–260 : PMLR.

ROY S., EHTESHAM N., ISLAM M. S. *et al.* (2021). Augmenting bert with cnn for multiple choice question answering. In *2021 24th International Conference on Computer and Information Technology (ICCIT)*, p. 1–5 : IEEE.

SAVELKA J., AGARWAL A., BOGART C. & SAKR M. (2023). Large language models (gpt) struggle to answer multiple-choice questions about code. *arXiv preprint arXiv :2303.08033*.

SOARES M. A. C. & PARREIRAS F. S. (2020). A literature review on question answering techniques, paradigms and systems. *Journal of King Saud University-Computer and Information Sciences*, **32**(6), 635–646.

SU J., LU Y., PAN S., MURTADHA A., WEN B. & LIU Y. (2021). Roformer : Enhanced transformer with rotary position embedding. *arXiv preprint arXiv :2104.09864*.

TAORI R., GULRAJANI I., ZHANG T., DUBOIS Y., LI X., GUESTRIN C., LIANG P. & HASHIMOTO T. B. (2023). Stanford alpaca : An instruction-following llama model. https://github.com/tatsu-lab/stanford_alpaca.

TAY Y. (2023). A new open source flan 20b with ul2.

TAYLOR R., KARDAS M., CUCURULL G., SCIALOM T., HARTSHORN A., SARAVIA E., POULTON A., KERKEZ V. & STOJNIC R. (2022). Galactica : A large language model for science.

TEAM M. N. (2023). Introducing mpt-7b : A new standard for open-source, commercially usable llms.

TOUVRON H., LAVRIL T., IZACARD G., MARTINET X., LACHAUX M.-A., LACROIX T., ROZIÈRE B., GOYAL N., HAMBRO E., AZHAR F., RODRIGUEZ A., JOULIN A., GRAVE E. & LAMPLE G. (2023). Llama : Open and efficient foundation language models.

WANG Y., MISHRA S., ALIPOORMOLABASHI P., KORDI Y., MIRZAEI A., ARUNKUMAR A., ASHOK A., DHANASEKARAN A. S., NAIK A., STAP D., PATHAK E., KARAMANOLAKIS G., LAI H. G., PUROHIT I., MONDAL I., ANDERSON J., KUZNIA K., DOSHI K., PATEL M., PAL K. K., MORADSHAHI M., PARMAR M., PUROHIT M., VARSHNEY N., KAZA P. R., VERMA P., PURI R. S., KARIA R., SAMPAT S. K., DOSHI S., MISHRA S. D., REDDY S. C., PATRO S., DIXIT T., DONG SHEN X., BARAL C., CHOI Y., HAJISHIRZI H., SMITH N. A. & KHASHABI D. (2022). Benchmarking generalization via in-context instructions on 1,600+ language tasks.

WU C., ZHANG X., ZHANG Y., WANG Y. & XIE W. (2023). Pmc-llama : Further finetuning llama on medical papers.