Qui de DrBERT, Wikipédia ou Flan-T5 s'y connaît le plus en questions médicales ?

Clément Besnard, Mohamed Ettaleb, Christian Raymond, Nathalie Camelin

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

Qui de DrBERT, Wikipédia ou Flan-T5 s'y connaît-il le plus en questions médicales ?

Clément Besnard[1] Mohamed Ettaleb[1] Christian Raymond[2] Nathalie Camelin[1]
(1) LIUM, Le Mans Université, France
(2) INSA Rennes, IRISA, France
[prénom].[nom]@univ-lemans.fr, [prénom].[nom]@irisa.fr

RÉSUMÉ

Cet article décrit la participation de l'équipe LIUM-IRISA à la campagne d'évaluation DEFT 2023. Notre équipe a participé à la tâche principale. Cette année, celle-ci porte sur la résolution automatique de questions à choix multiples dans le domaine médical. Nous avons mis en place plusieurs systèmes : un premier qui exploite une base de connaissances, un second interroge un modèle génératif en lui demandant de répondre directement aux questions et le dernier système combine un ensemble de descripteurs.

ABSTRACT

This paper describes the participation of the LIUM-IRISA team in the DEFT 2023 evaluation campaign. Our team participated in the main task, which this year consists of developing approaches for automatically answering medical multiple-choice questions. We have implemented several systems, the first use a knowledge base, a second use generative model-based system, and a final system combining a set of descriptors.

MOTS-CLÉS : base de connaissances, modèles neuronaux pré-entrainés, TF-IDF, corpus spécifique.

KEYWORDS: knowledge base, pre-trained neural models, TF-IDF, specific corpus.

1 Introduction

L'édition 2023 du Défi Fouille de Textes (DEFT) porte sur l'exploration d'un corpus de questions fermées en français. Elles proviennent d'annales d'examens de pharmacie et sont réunies dans le corpus *FrenchMedMCQA* (Labrak *et al.*, 2022). Les questions offrent la possibilité de choisir une ou plusieurs réponses parmi celles proposées. Le challenge consiste en la mise en place d'approches afin de répondre automatiquement à ces questions à choix multiples. Il s'agit donc d'associer, de choisir, un ensemble fini de réponses à une question énoncée en langage naturel dans le domaine spécifique du médical.

Une des difficultés de cette tâche provient en partie du fait que le vocabulaire à traiter est très spécifique. Potentiellement, les mots les plus caractéristiques du sens de la question et des réponses n'ont que très peu d'occurrences dans le corpus. Ainsi, appliquer les méthodes classiques de représentation vectorielles des mots par des modèles neuronaux pré-entraînés peut s'avérer plus complexe que pour des mots usuels présentant de nombreuses occurrences et de nombreux contextes différents

d'apparition.

Par ailleurs, cette tâche peut s'apparenter à un cas particulier de la tâche classique question/réponse pour laquelle il existe plusieurs systèmes à l'état de l'art. Ces systèmes combinent généralement un modèle qui recherche un *contexte* permettant de répondre à la question avec un modèle *Lecteur/Générateur* qui à partir du contexte et de la question extrait la réponse.

Comme décrit dans l'article de Weng (Weng, 2020), plusieurs méthodes existent afin de trouver un contexte, mais cela reste limité aux questions ouvertes sur un sujet précis. Il est plus difficile de trouver un contexte dans notre application aux questions fermées. Cependant, des systèmes comme T5 (Raffel *et al.*, 2020) ou GPT-3 (Brown *et al.*, 2020) peuvent s'affranchir de contextes grâce à leur nombre élevé de paramètres. Le nombre de paramètres (plusieurs milliards) que peuvent contenir les deux modèles génératifs précédents permet de stocker de l'information et des connaissances. Cela permet d'atteindre des performances similaires à plusieurs modèles de type BERT (Devlin *et al.*, 2019) associés à un système de recherche de contexte.

Pour cette édition, deux tâches sont proposées. La tâche principale consiste à identifier automatiquement et exactement quelles sont les réponses correctes. La tâche annexe se limite à identifier le nombre de réponses correctes sans indiquer précisément lesquelles sont correctes. Notre équipe a porté son attention sur la tâche principale, pour laquelle nous avons élaboré plusieurs systèmes de classification. Le premier système repose sur de la fouille dans une base de connaissances, le deuxième utilise un modèle génératif tandis que le dernier combine un ensemble de descripteurs.

La structure de l'article est la suivante : après une brève description du corpus et de la tâche dans la section 2, la section 3 présente les différents systèmes proposés. Les résultats des expériences menées sont exposés dans la section 4, suivis d'une synthèse des conclusions dans la section 5.

2 Analyse du corpus

Le corpus *FrenchMedMCQA* est une collection de 3 105 questions fermées en français provenant d'annales d'examens de pharmacie. Ce corpus est divisé en trois sous-ensembles, à savoir l'entraînement, le développement et le test. Les questions sont réparties de la manière suivante entre ces trois ensembles : 70% des questions pour l'entraînement, 10% pour le développement et 20% pour le test. Nous avons utilisé l'entrainement pour apprendre nos modèles et le corpus de test n'a été fourni que pendant la phase d'évaluation.

Chaque question est représentée par un identifiant et contient l'énoncé en langage naturel de la question, les cinq énoncés en langage naturel des options de réponse et l'ensemble des réponses correctes à la question. La question contient également le nombre de réponses correctes ainsi que le type de question : *simple* pour une seule réponse et *multiple* pour plusieurs réponses possibles.

On note ainsi une première différence entre les questions : soit on recherche *une seule* réponse, soit *plusieurs* réponses sont correctes.

On note également une deuxième différence, d'un point de vue de la sémantique :

— Soit la réponse recherchée est positivement liée à la question, comme par exemple dans l'Énoncé 1 (Table 1).

— Soit la réponse ne doit pas être vraie vis à vis de la question posée, comme par exemple dans

l'Énoncé 2 (Table 1).

Énoncé 1	*"Parmi les propositions suivantes, une seule est exacte. Laquelle ? La sérotonine est le (la) :"*
Énoncé 2	*"Parmi les affirmations suivantes, une seule est fausse, indiquer laquelle : les particules alpha"*
Énoncé 3	*"Parmi les propositions suivantes, laquelle (lesquelles) est (sont) exacte(s) ?"*

TABLE 1 – Quelques exemples d'énoncés de questions

Pour finir, nous avons noté une dernière différence entre les questions. Certaines contiennent des informations sémantiquement pertinentes, comme dans l'Énoncé 1 (Table 1) où l'on comprend que la question traite de la *sérotonine*. D'autres en revanche, ne présentent pas de sujet précis dans leur intitulé, comme dans l'Énoncé 3 (Table 1).

Partant de ce constat, nous avons choisi d'appliquer un premier système permettant de déterminer le type de la question avant d'appliquer ensuite un de nos systèmes entraînés pour détecter les réponses à associer aux questions. L'ensemble de ces systèmes est présenté dans la section suivante.

3 Systèmes

Nous avons mis en place plusieurs systèmes et un méta-système par apprentissage qui tente de fusionner un ensemble de descripteurs. Ces systèmes ont pour objectif d'estimer la pertinence qu'une réponse candidate soit bonne.

Comme les énoncés demandent de trouver soit la/les bonnes réponse(s) soit le(s) intru(s), nous avons développé un détecteur de type d'énoncé qui va conditionner la manière dont nous allons utiliser le score de pertinence calculé par les systèmes pour associer les réponses recherchées à l'énoncé de la question. Tout ceci est détaillé dans la section suivante.

3.1 Détection du type de question et stratégie de réponse

Une première information à connaître afin de répondre à une question est d'identifier si l'association question/réponse est positive ou négative (rechercher la bonne ou la mauvaise réponse). Après avoir analysé les énoncés des questions, nous avons remarqué que cette information pouvait être obtenue à l'aide d'une simple expression régulière qui détecte la présence de certains mots clés. Ainsi, nous avons recherché les mots "sauf", "fausse", "fausses", "ne" "inexacte", "inexactes", "n'", qui sont apparus spécifiques aux questions qui demandent la/les mauvaises réponses.

Une deuxième information utile pour nos systèmes est d'identifier si l'on veut une seule ou bien plusieurs réponses. Cela peut difficilement être réalisé par une expression régulière, car rechercher la présence de certains mots n'est pas suffisant. Un modèle de type CamemBERT (Martin *et al.*, 2020a) avec l'ajout d'une couche de classification permet de réaliser cette tâche. Celui-ci va automatiquement extraire les spécificités de chaque type de question afin de prédire une des deux classes ('simple' ou 'multiple'). Nous avons obtenu un F1-score de 96,90 sur le corpus de développement pour cette tâche.

Un système à base de règles est utilisé afin de prédire la ou les réponses correctes. En entrée du système, on a un score pour chaque réponse, l'information qui nous indique si l'on veut les réponses correctes ou non ainsi que le type de question ('simple' ou 'multiple').

À partir des sorties d'un système, les règles sont les suivantes :

1. Si l'on veut une seule réponse et la réponse correcte, on retourne la réponse avec le plus grand score.
2. Si l'on veut une seule réponse et la réponse fausse, on retourne la réponse avec le plus petit score.
3. Si l'on veut plusieurs réponses et les réponses correctes, on retourne les réponses qui ont un score supérieur au premier quartile des scores.
4. Si l'on veut plusieurs réponses et les réponses fausses, on retourne les réponses qui ont un score inférieur au troisième quartile des scores.

Si l'on cherche une seule réponse, et que l'ensemble des scores est nul, ou bien que plusieurs réponses ont un score égal, le choix de la lettre est réalisé en fonction de la fréquence d'apparition de chaque lettre dans le corpus d'apprentissage. L'ordre défini est différent si l'on cherche la bonne ou bien la mauvaise réponse : **Bonne réponse** : 'd', 'c', 'b', 'a', 'e' ; **Mauvaise réponse** : 'd', 'c', 'e', 'b', 'a'

Lorsque l'on cherche plusieurs réponses et que le score pour chaque réponse est nul, la réponse multiple la plus fréquente dans le corpus d'apprentissage est retournée : **Réponse multiple** : 'bcd'.

3.2 Fouille dans une base de connaissances

Notre première approche consiste à vérifier si chacune des réponses candidates peut être associée à sa question en exploitant une base de connaissances.

3.2.1 Système FBC-ngram-rule

Pré-traitements L'ensemble des pré-traitements suivants ont été appliqués à l'ensemble des textes (énoncés de questions, réponses, titre et contenu des articles) :

— Suppression de la ponctuaction.

— Suppression des stopwords : Nous avons utilisé la liste de mots de la bibliothèque Spacy que nous avons enrichi des mots avec forte occurrence et sans apport sémantique dans l'énoncé des questions : '%exact%', 'proposition%', 'indique%', 'réponse%', 'fausse%', 'affirmation%', 'propos', 'vraie%', 'coche%', 'donner','trouve'.

Les réponses et le contenu des articles ont également subi :

— Modification de la liste de stopword : 'moins', 'plus', 'peu' ainsi que les chiffres ont été retirés de la liste de stopwords. En effet, ces informations apportent des nuances et des indications utiles à la compréhension sémantique de la réponse.

— Remplacement des caractères grecs par leur forme latine

— Remplacement des chiffres romains dans leur notation arabe

— Normalisation des caractères selon la norme NFKD afin d'éviter les variations potentielles

— Suppression des sauts de lignes

— Lemmatisation avec le modèle *fr_core_news_md* de l'outil Spacy [1]

Construction de la base de connaissances Deux approches ont été testées afin d'obtenir une liste d'articles pertinents :

1. Modèle Vectoriel : l'énoncé de la question et les titres de chacun des articles sont représentés par leur vecteur de poids tf-idf puis une similarité cosinus est appliquée pour extraire les articles les plus pertinents selon cette mesure.
2. API Wikipedia [2] : l'API de recherche Wikipédia est utilisée avec comme requête l'énoncé de la question. Les articles les plus pertinents sont alors extraits directement par l'API (recherche par présence de mots clés).

Les n articles les plus pertinents vis à vis de l'énoncé de la question constituent alors la base de connaissance pour la question.

Nous avons ensuite choisi de compter le nombre d'occurrences des unigrammes et des bigrammes présents à la fois dans l'énoncé de la réponse et dans cette base de connaissances.

Association de la base de connaissance aux questions À partir du nombre d'unigrammes et de bigrammes de chaque réponse, un score est calculé selon la formule :

$$(2 * NbBigram + NbUnigram)/NbTokens \tag{1}$$

Nous avons choisi de donner un poids plus importants aux bigrammes car ceux-ci étaient beaucoup moins fréquents que les unigrammes. Par manque de temps, nous avons appliqué une formule simple avec un poids doublé. Une meilleure proposition consisterait à pondérer le nombre de bigrammes et d'unigrammes avec une formule tf-idf.

Selon cette formule, une réponse plus longue aura mécaniquement un score plus élevé, on normalise donc le score par le nombre de mots dans la réponse pré-traitée.

Les scores calculés sont ensuite utilisés comme décrit dans la section 3.1.

3.3 Système Flan-T5

Cette approche utilise *Flan-T5* (Chung *et al.*, 2022), un modèle de langage affiné sur plus de 1 000 tâches (traduction, résumé, classification, question/réponse). Ce type de modèle génère la séquence de mots la plus probable sachant la séquence de mots précédents donnée en entrée du système. L'idée est d'interroger Flan-T5 afin de comparer les réponses candidates à celles qu'il propose.

Il est possible de le spécialiser sur de nouvelles taches grâce à des instructions. Deux instructions ont été définies :

— Une première qui permet d'indiquer au modèle qu'il doit fournir une seule réponse. Le format d'entrée est le suivant : *Choisis la bonne réponse : **{question}** (A) **{réponse A}** (B) **{réponse B}** (C) **{réponse C}** (D) **{réponse D}** (E) **{réponse E}** context : **{contexte}***

1. https://spacy.io/
2. https://fr.wikipedia.org/w/index.php?search=&title=Spécial:Recherche

— Une seconde qui permet de choisir plusieurs réponses : *Choisis les bonnes réponses : **{question}** (A) **{réponse A}** (B) **{réponse B}** (C) **{réponse C}** (D) **{réponse D}** (E) **{réponse E}** context : **{contexte}***

Le contexte ajouté à la fin de l'entrée a été obtenu à l'aide d'un modèle DPR (Dense Passage Retrieval) en français (Karpukhin *et al.*, 2020). Ce type de modèle a été développé dans le cadre de la tâche de questions ouvertes. Il permet de sélectionner pour une question les contextes contenant la réponse. Ce modèle est basé sur une architecture dense avec deux encodeurs. Les données utilisées pour l'entraînement sont pour chaque question les contextes positifs, négatifs et fortement négatifs. Les contextes positifs sont ceux contenant la réponse à la question posée. Le but étant de réduire la distance entre les représentations vectorielles des paires de questions et des contextes positifs.

Les articles Wikipédia [3] en français ont été découpés dans un ensemble de passage de 100 mots chacun avec un recouvrement de 10 mots entre chaque passage du même article. Le passage le plus proche de notre question obtenu par similarité cosinus est utilisé comme contexte. Nous avons utilisé deux modèles déjà entraînés à cette tâche. Un premier modèle (etalab-ia/dpr-question_encoder-fr_qa-camembert) encode la question avec ses réponses dans un vecteur de taille 768. Un second modèle (etalab-ia/dpr-ctx_encoder-fr_qa-camembert) encode les passages. Les deux modèles permettant de réaliser les plongements des questions et des passages ont été réalisés par la branche Intelligence artificielle de l'Etalab [4].

En sortie, le système a été entraîné pour donner les lettres des réponses correctes de A à E séparées par des espaces.

Lors de l'entraînement, la version *Flan-T5-large* (780 millions de paramètres) du modèle a été utilisée sur 10 époques avec un taux d'apprentissage de $5e^{-5}$, un batch de taille 4, un *weight decay* de 0,01. L'aléatoire a été contrôlé en utilisant la graine 0. Après chaque époque, une version du modèle est sauvegardée, celle qui obtient les meilleurs résultats en termes d'*Exact Match ratio* sur l'ensemble de développement a été sélectionnée.

3.4 Méta-système

Un dernier système a été implémenté en utilisant un algorithme de boosting appris sur un ensemble de descripteurs.

Dans un premier temps, le corpus de questions est *binarisé* : un ensemble de couples énoncé_question/énoncé_réponseX est créé à partir de chaque question multiple (autant de couples que de réponses candidates). Ensuite, il s'agit de vérifier la validité de l'association question/réponseX grâce au méta-système.

Les descripteurs suivants ont été extraits :

1. **FBC-ngram-rule** : le nombre d'apparitions des unigrammes et bigrammes pour chaque réponse avec les 20 articles, les 5 articles et l'article le plus proche selon l'API de recherche Wikipédia.
2. **Flan-T5** : les scores pour chaque réponse à partir de la distribution de probabilité qui permet au modèle Flan-T5 de générer la lettre de la première réponse.
3. **Descripteur biomédical** : Nous avons utilisé le modèle DrBERT (Labrak *et al.*, 2023) qui est pré-entraîné sur des corpus de données médicales en français. Celui-ci permet de produire

3. https://dumps.wikimedia.org/frwiki/latest/frwiki-latest-pages-articles.xml.bz2
4. Département de la direction interministérielle du numérique

des vecteurs représentatifs d'un énoncé médical (token '[CLS]'). DrBERT est interrogé pour représenter énoncés des questions et énoncés des réponses. Une similarité cosinus est ensuite calculée pour chaque couple question/réponseX.

4. **Enrichissement de l'énoncé de la question** : Un score de similarité (cosinus) est calculé entre chaque réponse et la question enrichie. Pour enrichir la question, nous procédons à deux étapes : 1) Uniquement les 5 mots ayant le plus haut score de tf-idf sont conservés dans l'énoncé de la question ; 2) les 10 noms (NOUN dans Spacy) les plus fréquents dans les 10 pages les plus pertinentes selon l'API Wikipedia sont ajoutés à cet énoncé réduit.
5. **Utilisation d'un système de question/réponse** : L'énoncé de la question est à nouveau réduit à ces 5 mots les plus pertinents et les 10 articles les plus pertinents sont à nouveau considéré. Le modèle francophone de question/réponse *Camembert-base-squadFR-fquad-piaf*[5] est utilisé pour extraire la réponse des articles pertinents en considérant l'énoncé réduit de la question comme requête. Un score de similarité cosinus est ensuite calculé entre la réponse du modèle et la réponse candidate considérée.
6. **Utilisation d'un seul article pertinent** : Le dernier descripteur est obtenu comme le précédent mais en ne considérant qu'un seul article pertinent au lieu de 10.

Tous ces descripteurs ont été utilisés en entrée d'un algorithme de Gradient Boosting afin d'évaluer la validité d'association de chacun de nos couples question/réponseX.

4 Expériences et résultats

Les deux métriques utilisées afin d'évaluer la performance des systèmes sont le *Hamming score* (taux de bonnes réponses parmi l'ensemble des hypothèses et références) et l'*Exact Match Ratio* (taux de réponses parfaitement justes).

La table 2 montre les résultats du système *FBC-ngram-rule* selon les 2 méthodes proposées et en considérant plusieurs valeurs n.

		1 article		5 articles		20 articles	
		Hamming	EMR	Hamming	EMR	Hamming	EMR
API Wikipédia	Dev	36,43	17,31	38,74	18,91	38,35	17,95
	Test	-	-	36,72	17,85	-	-
Modèle Vectoriel	Dev	35,60	17,31	34,78	16,35	-	-

TABLE 2 – Résultats système 1 : FBC-ngram-rule

L'API de Wikipédia est la meilleure méthode pour extraire les documents pertinents. De plus, on note qu'il est intéressant de considérer 5 articles. En revanche, en considérer beaucoup plus comme 20 n'apporte pas de gain significatif. En considérant ce système et le corpus de développement, nous avons observé ceci :

5. Il utilise comme base CamemBERT (Martin *et al.*, 2020b) fine-tuné sur la combinaison de trois jeux de données francophones de questions-réponses : PIAFv1.1 (Keraron *et al.*, 2020), FQuADv1.0 (d'Hoffschmidt *et al.*, 2020), SQuAD-FR (Kabbadj, 2018).

— Recherche d'une réponse *simple* : En considérant les 312 questions du corpus de développement, 6 d'entres elles obtiennent un descripteur nombre d'unigramme et de bigramme nul. 17 questions obtiennent des scores égaux pour plusieurs réponses (dont 11 lorsque l'on recherche une réponse incorrecte).

— Recherche d'une réponse *multiple* : Un ensemble de scores nuls apparait pour 3 questions.

Notons que les résultats sur le test avec ce système restent moins performants que ceux obtenus sur le corpus de développement. Cependant, l'utilisation de notre système assez léger permet d'atteindre des résultats similaires à des plus gros modèles de type BERT ou RoBERTa (Labrak *et al.*, 2022).

Les résultats du système Flan-T5 sont présentés dans la Table 3.

	Wiki passages DPR		Sans contexte	
	Hamming	EMR	Hamming	EMR
Dev	44,88	25,64	45,04	25,32
Test	43,24	22,19	-	-

TABLE 3 – Résultats Flan-T5

Les résultats obtenus sont bien meilleurs qu'avec le premier système. On remarque que l'ajout d'un contexte ne permet pas d'améliorer significativement les résultats. Il est difficile d'associer un contexte utile pour répondre aux questions avec seulement des passages de 100 mots. Par ailleurs, on observe à nouveau une baisse des résultats sur le test par rapport au développement, notamment en termes d'*Exact Match Ratio*. Cela peut s'expliquer par un nombre de questions *multiples* plus important.

Après la phase de test, nous avons réappris le modèle Flan-T5 sur l'ensemble apprentissage+développement sur 3 époques (meilleur paramètre lors de la phase de dev), nous avons améliorer légèrement les résultats sur le Test avec un *Hamming* de **45,84** et un *Exact Match Ratio* de **23,15**.

Les résultats de notre dernier système sont présentés en Table 4.

Dev		Test	
Hamming	EMR	Hamming	EMR
44,42	24,36	35,47	18,49

TABLE 4 – Méta-système qui combine toutes les features

Les résultats indiquent que le système a obtenu des performances modérées. Il a réussi à identifier correctement la réponse pour environ la moitié des paires de questions et de réponses sur les données de développement (*Hamming* de **44,42**), mais n'a pas été en mesure de trouver toutes les réponses pour la plupart des questions sur les données de test (*Hamming* de **35,47**). Il est essentiel d'analyser les raisons derrière ces résultats et de cibler les points faibles afin d'améliorer les performances. Plusieurs facteurs, tels que la qualité et la taille des données d'entraînement, la complexité des modèles et des algorithmes utilisés, peuvent tous influencer les performances du système. Cependant, pour améliorer les performances et obtenir des correspondances exactes, nous devrons aborder ces facteurs limitants de manière approfondie. L'amélioration de la mesure de similarité entre les questions et les réponses pourrait être une piste à explorer. De plus, une analyse plus rigoureuse de la sélection des caractéristiques pourrait permettre d'identifier des aspects plus pertinents et discriminants pour améliorer la précision des correspondances.

5 Conclusion

En conclusion, nous avons proposé trois systèmes. Le premier système est basé sur une approche avec une base de connaissances qui utilise la similarité cosinus entre vecteurs de poids TF-IDF et l'API de recherche Wikipédia pour identifier les articles pertinents. Les réponses sont ensuite classées en fonction du nombre d'unigrammes et de bigrammes qui se trouvent dans ces articles. Le système utilise ensuite des règles pour prédire la ou les réponses correctes. Le deuxième système utilise Flan-T5, un modèle de langage pré-entrainé sur plus de 1 000 tâches, pour générer la séquence de mots la plus probable pour chaque réponse. Le modèle est spécialisé pour les tâches de question à choix multiples en fournissant des instructions pour fournir une seule ou plusieurs réponses. Le troisième système utilise une approche de classification. Le modèle est entraîné sur un ensemble de descripteurs pour prédire la ou les réponses correctes à une question donnée. Après avoir étudié plusieurs architectures, c'est le modèle Flan-T5 qui se distingue le plus. Le nombre important de paramètres lui permet d'obtenir de meilleurs résultats. La base de connaissances qu'est Wikipédia nous permet d'atteindre des résultats équivalents aux systèmes *baseline* mais les règles appliquées rencontrent des limites comme notamment la gestion des questions *multiples* et les négations dans les réponses. Pour finir, les descripteurs calculés à partir de DrBERT ne se sont pas montrés à la hauteur de nos espérances. En effet, le classement des descripteurs par l'algorithme de boosting de notre méta-modèle a montré que le descripteur biomédical était le moins pertinent de tous.

Références

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 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.

DEVLIN J., CHANG M.-W., LEE K. & TOUTANOVA K. (2019). Bert : Pre-training of deep bidirectional transformers for language understanding.

D'HOFFSCHMIDT M., VIDAL M., BELBLIDIA W., BRENDL'E T. & HEINRICH Q. (2020). Fquad : French question answering dataset. *ArXiv*, **abs/2002.06071**.

KABBADJ A. (2018). Something new in french text mining and information extraction (universal chatbot) : Largest qa french training dataset (110 000+). [Online ; posted 11-November-2018].

KARPUKHIN V., OGUZ B., MIN S., LEWIS P., WU L., EDUNOV S., CHEN D. & YIH W.-T. (2020). Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, p. 6769–6781, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2020.emnlp-main.550.

KERARON R., LANCRENON G., BRAS M., ALLARY F., MOYSE G., SCIALOM T., SORIANO-MORALES E. & STAIANO J. (2020). Project PIAF : building a native french question-answering dataset. In *LREC*, p. 5481–5490 : European Language Resources Association.

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.

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.

MARTIN L., MULLER B., SUÁREZ P. J. O., DUPONT Y., ROMARY L., DE LA CLERGERIE É. V., SEDDAH D. & SAGOT B. (2020a). Camembert : a tasty french language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

MARTIN L., MULLER B., SUÁREZ P. J. O., DUPONT Y., ROMARY L., DE LA CLERGERIE É. V., SEDDAH D. & SAGOT B. (2020b). Camembert : a tasty french language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*.

RAFFEL C., SHAZEER N., ROBERTS A., LEE K., NARANG S., MATENA M., ZHOU Y., LI W. & LIU P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer.

WENG L. (2020). How to build an open-domain question answering system ? *lilianweng.github.io*.