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

Participation de l'équipe TTGV à DEFT 2023 : Réponse automatique à des QCM issus d'examens en pharmacie

Andréa Blivet[1*] Solène Degrutère[1*] Barbara Gendron[2*] Aurélien Renault[3*]
Cyrille Siouffi[2*] Vanessa Gaudray-Bouju[1*] Christophe Cerisara[2*]
Hélène Flamein[1*] Gaël Guibon[2*] Matthieu Labeau[3*] Tom Rousseau[1*]
(1) DTIPG SNCF, 1-3 avenue François Mitterrand, 93210 Saint-Denis, France
(2) LORIA, Université de Lorraine, CNRS, 54000 Nancy, France
(3) LTCI, Télécom Paris, Institut Polytechnique de Paris, 19 place Marguerite Perey, 91120 Palaiseau, France

andrea.blivet@sncf.fr, solene.degrutere@sncf.fr, barbara.gendron@loria.fr,
aurelien.renault@polytechnique.edu, cyrille.siouffi@loria.fr,
ext.vanessa.gaudray-bouju@sncf.fr, christophe.cerisara@loria.fr,
helene.flamein@sncf.fr, gael.guibon@loria.fr,
matthieu.labeau@telecom-paris.fr, tom.rousseau@sncf.fr

1 Introduction

Cette année, l'équipe TTGV (TAL à Très Grande Vitesse) se (re)forme autour de l'équipe TGV qui a participé au défi DEFT en 2022. Comptant désormais 11 membres issus de la SNCF, de Télécom Paris et du LORIA, notre équipe s'est attelée à la résolution des deux tâches proposées : l'identification du nombre de réponses supposément justes à un QCM et la prédiction de l'ensemble de réponses correctes parmi les cinq proposées pour une question donnée. Ces questions proviennent du corpus FrenchMedMCQA, qui regroupe 3 105 questions fermées extraites des annales d'examens de pharmacie en français. Les travaux de (Labrak *et al.*, 2022) décrivent plus en détail les particularités de ce corpus et présentent les premières expérimentations pour aborder ces tâches. Dans notre étude, nous avons décidé de nous appuyer sur différentes approches, tout en tirant des enseignements de la *baseline* établie par (Labrak *et al.*, 2022).

La première partie de notre article se focalisera donc sur les différentes méthodologies mises en oeuvre, explorant ainsi un large éventail d'approches et de techniques pour aborder d'abord la distinction entre les questions appelant une seule ou plusieurs réponses avant de s'interroger sur l'identification des réponses correctes. Nous détaillerons les différentes méthodes utilisées, en mettant en exergue leurs avantages et leurs limites respectives. Ensuite, nous présenterons les résultats obtenus pour chaque approche. Enfin, nous discuterons des limitations intrinsèques aux tâches elles-mêmes ainsi qu'aux approches envisagées dans cette contribution.

*. Contributions égales.

Actes du Défi Fouille de Textes@TALN2023, pages 23–38.

2 Méthodologie

La répartition du nombre de réponses attendues pour chaque question dans les jeux de données (*train*, *dev* et *test*) est illustrée dans la figure 1 en annexe. Parmi les 3 105 questions du corpus, on observe que 35% (soit 1 080 questions) nécessitent uniquement une réponse. Alors que le *train* contient davantage de questions demandant trois réponses, les questions ne demandant qu'une seule réponse sont majoritaires dans le *dev* et le *test*. D'une manière générale, la répartition des nombres de réponses suit une tendance similaire entre le *dev* et le *test*. Étant donné la répartition des données, nous allons d'abord nous concentrer sur la distinction des questions n'appelant qu'une seule réponse (dites simples) et celles qui en exigent plusieurs (dites multiples).

2.1 Classification des questions simples et multiples

La formulation des questions et des propositions de réponses permet d'estimer si la question attend une ou plusieurs réponses. Un rapide parcours du jeu de données met notamment en lumière les particularités suivantes :

- **Questions** : Certains mots – principalement les dérivés du radical « quel »– donnent un indice sur le nombre de réponses attendues (quel(s), le(s)quel(s), etc.).
- **Réponses** : Dans certains cas, les réponses présentent des affirmations, parfois incompatibles, sur des thèmes apparemment disjoints. En supposant qu'exactement une affirmation par thème est vraie, le nombre de thèmes disjoints dans les réponses pourrait correspondre au nombre de réponses attendues pour la question concernée.

Trois stratégies peuvent être proposées pour estimer le nombre de réponse à une question : 1) **Expressions régulières (ER)** : Cette approche détecte la présence de mots-clefs (pronoms, pronoms démonstratifs...) dans le libellé d'une question pour estimer, selon l'accord en nombre, si elle admet une ou plusieurs réponses. 2) ***Topic modelling* (LSI & LDA)** : Cette approche vise à identifier des thèmes disjoints à partir des mots composant assertion, sans prendre en compte leur sémantique. Parmi les algorithmes reconnus pour effectuer ce type de tâche, nous avons exploré le *Latent Semantic Indexing* (LSI) (Hofmann, 1999) et le *Latent Dirichlet Allocation* (LDA) (Dumais *et al.*, 2004). 3) **Régression Logistique (RL)** : Cette approche statistique est couramment utilisée pour modéliser des événements binaires en fonction de variables indépendantes (Kleinbaum *et al.*, 2002).

S'agissant de tâches de classification – binaire ou multi-classe –, nous évaluons la performance des méthodes proposées avec les indicateurs usuels : précision, rappel, F_1-score.

Analyse des questions par expressions régulières La présence de pronoms – démonstratifs ou non – dans le libellé des questions donne un indice sur le nombre de réponses attendues. Les expressions régulières permettent de les détecter efficacement. Quelques exemples tirés du jeu d'entraînement sont présentés dans l'annexe B.3. Les exemples n^{os} 68 et 2 161 illustrent que la seule détection des dérivés de « celui » et de « quel » conduit parfois à des ambiguïtés : il faut également assurer qu'ils n'apparaissent pas simultanément au singulier et au pluriel dans le libellé de la question. Nous avons alors créé un système d'expressions régulières permettant de capter les instances de mots dérivés précédemment cités uniquement au singulier (cf. annexe B.1).

Ce système de détection permet de distinguer efficacement les questions à réponse simple ou multiple. En effet, nous obtenons un taux d'exactitude (*accuracy*) de 94% sur le jeu d'entraînement, 89% sur

le jeu de développement. Les scores de classification détaillés en tableau 3 mettent en lumière les principales lacunes de ce classifieur : davantage de faux positifs sur la classe simple ; davantage de faux négatifs sur la classe multiple. A titre d'illustration, quelques exemples de faux positifs dans la classe simple ont été ajouté en annexe B.3. Nous avons alors exploré une approche complémentaire, s'intéressant aux contenu des réponses possibles.

Analyse des réponses par *topic modelling* La formulation des réponses peut indiquer combien d'entre elles sont acceptables, *via* le nombre de thèmes différents qu'elles abordent. Le *topic modelling* s'intéresse à la construction automatique de thèmes représentés dans un corpus. Certains algorithmes, à l'instar de LSI et LDA (cf. annexe B.2), résolvent cette tâche par apprentissage statistique. Dans notre contexte, ces deux approches nous ont paru justifiées (i) par la nature monosémique du vocabulaire médical, (ii) par le caractère laconique des réponses, (iii) par le fait que des « thèmes » pourraient se manifester dans les propositions de réponses de plusieurs questions. Sous ces hypothèses, nous avons alors modélisé les thèmes comme suit :

- **Constitution du corpus** : Chacune des cinq réponses proposées pour les 2 171 questions est considérée comme un document. Le corpus regroupe donc $5 \times 2\,171 = 10\,855$ documents.
- **Prétraitement** : Les documents sont prétraités selon les étapes suivantes : découpage des phrases en unités lexicales (*tokenization*) en excluant la ponctuation et mots parasites (*stop-words*), extraction des racines de chaque unité (*stemming*), constitution d'un dictionnaire, transformation des documents en sacs de mots (*Bag of Words*, ou *BoW*), enfin normalisation du corpus transformé *via* TF-IDF, afin d'occulter les mots communs à trop de documents.
- **Hyper-paramétrage** : Estimation du meilleur nombre N de thèmes pour le LSI et le LDA. Pour le LDA, le nombre optimal de thèmes obtenu est de 57.
- **Inférence** : Pour chaque nouvelle question, les cinq réponses proposées sont concaténées en un seul document. Le modèle (LSI ou LDA) en extrait alors des thèmes, que l'on filtre par probabilité d'occurrence *via* un seuillage adaptatif. Le nombre de thèmes retenus correspond alors à notre prédiction de nombre de réponses pour cette question.

Compte tenu de l'écart important de performances entre les meilleurs LSI et LDA trouvés [1], nous ne présentons que les résultats obtenus pour le LDA, donnés dans le tableau 3. En dehors des faibles performances de ce modèle, on constate un fort déséquilibre entre précision et rappel sur chaque classe, que la classification soit binaire ou multiple. Notons aussi que la classe 4 n'a jamais été prédite sur le jeu de développement, alors que ce n'est pas la moins représentée.

Toutefois, l'analyse des réponses apporte des indications utiles, complémentaires à celles tirées du système d'expressions régulières. C'est pourquoi nous avons tenté de croiser les analyses des questions et des réponses en recoupant les prédictions du système d'expressions régulières (appliqué aux questions) et du LDA (appliquée aux réponses). Le croisement a été opéré de la façon suivante : si les expressions régulières prédisent que la réponse sera unique, on se fie à cette prédiction ; sinon, on utilise le LDA pour prédire le nombre de réponse. Les performances de classification sont effectivement meilleures que celles obtenues avec le seul LDA. En effet, le taux d'exactitude sur les deux tâches de classification s'améliore nettement, ce qui tient au fait que la classe 1 est mieux prédite. Cependant, comme l'illustre le tableau 3, cela ne suffit pas à améliorer significativement les performances sur les autres classes, diluant ainsi le gain précédemment relevé.

1. La phase d'hyperparamétrage du LSI n'a pas convergé, contrairement à celle de la LDA.

Analyse des réponses par *topic modelling* Notre dernière approche pour essayer de distinguer les questions simples des questions multiples s'appuie sur la régression logistique de *scikit-learn* (Pedregosa *et al.*, 2011). Grâce à cette bibliothèque, nous avons procédé à la tokenisation et à la vectorisation des questions en utilisant un *CountVectorizer*. Cela a permis de représenter les questions sous forme de vecteurs en indiquant quels mots sont présents dans chaque question. Pour réduire la dimensionnalité de la représentation vectorielle, nous avons conservé uniquement les 10 000 mots les plus fréquents dans le corpus pour construire une matrice des vecteurs représentant chaque question. Le modèle de régression logistique a été entraîné sur les données du jeu d'entraînement avec l'algorithme *liblinear*, particulièrement recommandé lorsque les données sont limitées. Étant donné qu'il y avait un déséquilibre entre les étiquettes (plus de questions multiples que de questions simples), nous avons effectué une balance des classes en ajustant les poids des échantillons lors de l'entraînement du modèle. Le modèle a finalement été utilisé pour prédire dans le jeu de données de développement la nature des questions (simple ou multiple) et a obtenu un F_1-score de 0,94 (cf. tableau 3 en annexes).

Analyse croisée des résultats La régression logistique s'est révélée être le meilleur modèle pour prédire si une question est de type simple ou multiple. Cependant, ses performances ne sont pas satisfaisantes lorsqu'il s'agit d'aller plus loin dans la classification des types de questions. En ce qui concerne la résolution de la tâche annexe proposée dans le cadre du défi, nous avons pris la décision de soumettre les prédictions du modèle combinant ER et LDA ainsi que les prédictions du modèle utilisant la régression logistique qui présente un bon F_1-score. Ce dernier ne permettant que de faire la distinction entre les questions simples et multiples, nous avons décidé de soumettre ses prédictions en considérant par défaut que les questions de type multiple appellent deux réponses.

Si la tâche annexe n'est pas résolue à ce stade, le modèle RL pourra trouver son utilité dans les autres méthodes pour aider la résolution de la tâche principale. En outre, les approches les plus encourageantes pour la résolution de la tâche principale – telle que l'approche multi-classe décrite dans la section suivante – pourront également être utilisées pour répondre à la tâche annexe.

2.2 Approche multi-classe

Dans cette partie, nous reprenons l'approche multi-classe proposée dans (Labrak *et al.*, 2022) ; de cette manière, le problème revient, pour chaque question, à prédire la bonne classe parmi les 31 possibles, i.e. le nombre total de combinaisons différentes des 5 réponses possibles. Quant au choix de la représentation, nous avons choisi d'uniquement travailler avec le modèle **DrBERT** (Labrak *et al.*, 2023), un modèle de langue français dérivé de **CamemBERT** (Martin *et al.*, 2020) et spécialisé sur des données biomédicales. Le tableau 1 montre le gain de performances de cette architecture comparée à d'autres moins spécialisées.

Ajout d'un contexte À l'instar de (Labrak *et al.*, 2022), nous considérons que l'apport de connaissances extérieures constitue un levier intéressant à étudier dans le cadre de cette tâche. Comme eux, nous avons fait le choix de nous appuyer sur le Wikipédia français pour construire des contextes plus denses en informations autour des questions et des réponses de chaque entrée de la base.

Pour mieux cibler la récupération des contextes, une première étape de sélection des termes les plus représentatifs des questions et des propositions de réponses a été intégrée. L'hypothèse ici est que les syntagmes porteurs des informations les plus significatives peuvent être identifiés grâce à la fréquence d'apparition de leurs lemmes dans l'ensemble du jeu de données. Le vocabulaire employé dans le corpus présentant de nombreuses récurrences (« Parmi les propositions suivantes »,

« laquelle », « quelle », « méthode », « indiquer », etc.), nous considérons que plus les lemmes sont fréquents dans le corpus, moins ils seront considérés comme spécifiques. À l'inverse, des lemmes plus rares seront plus susceptibles de représenter les notions centrales des questions et des réponses. En s'appuyant donc sur la fréquence moyenne d'apparition des lemmes dans le corpus et sur leur étiquetage morphosyntaxique, seuls les syntagmes nominaux composés entièrement de termes dont la fréquence d'apparition est supérieure à la moyenne sont retenus. En l'occurrence, cette moyenne s'élève à 12 dans les données du jeu de *test* pour un total de 3 233 lemmes. Cette approche permet de récupérer, pour chaque question, les syntagmes nominaux les plus pertinents en excluant les termes pas suffisamment spécifiques.

Pour extraire des éléments de contexte de Wikipédia, le module Cohere [2] semblait le plus approprié. En effet, il a été utilisé pour vectoriser les paragraphes de millions de pages Wikipédia en de nombreuses langues et permet de récupérer un ou plusieurs paragraphe(s) à partir d'une entrée textuelle grâce à des scores de similarité, obtenus par un produit scalaire. Nous avons testé trois configurations différentes pour récupérer les contextes :

- pour chacune des réponses proposées, concaténation de la question et de la réponse pour retrouver les trois paragraphes les plus similaires pour chacune des cinq entrées ;
- idem mais en ne récupérant qu'un seul paragraphe pour chaque entrée ;
- pour chacune des réponses proposées, concaténation des syntagmes nominaux de la question et de la réponse pour retrouver le paragraphe le plus similaire pour chacune des cinq entrées.

Les paragraphes récupérés sont ensuite concaténés de façon à former un contexte unique par question.

Le tableau 1 montre que dans la configuration actuelle l'ajout d'un contexte ne permet pas d'améliorer les performances ; au contraire, le modèle ne semble pas être en mesure d'extraire les informations pertinentes du contexte fourni. En effet, il semblerait que ces contextes restent encore trop longs pour le modèle. Des tentatives ont été entreprises pour les réduire et les rendre plus concis en incluant des informations pertinentes, mais jusqu'à présent, elles n'ont pas abouti. Parallèlement, nous avons envisagé de limiter nos contextes aux pages Wikipédia provenant des portails Pharmacie et Médecine, mais le manque de temps nous a empêchés d'explorer pleinement cette piste.

Approche contrastive Nous proposons également une extension contrastive à cette approche dans le but d'infuser de la connaissance durant la phase de *fine-tuning*, plus spécialement au niveau des entités nommées (Xiong *et al.*, 2020). Pour ce faire, nous obtenons des annotations d'entités en *finetunant* le susmentionné **DrBERT** sur un dataset de NER biomédicale français (QUAERO (Névéol *et al.*, 2014)). Les entités imbriquées ne sont pas gérées, i.e. seule la mention correspondant à l'entité la plus longue est conservée. Durant l'inférence, un mot reçoit une annotation d'entité si au moins le premier *token* de ce dernier est annoté. Ensuite, k exemples négatifs sont créés en remplaçant 50% des entités par d'autres entités du même type. Enfin, nous ajoutons un objectif supplémentaire dont le but est de reconnaître des entités ayant été substituées des entités « réelles ». Si e une entité, C son contexte associé et E^+ l'ensemble des vraies mentions d'entités, la tête *contrastive* revient à minimiser la fonction suivante :

$$\mathcal{L}_{\text{contrastive}} = \mathbb{1}_{e \in E^+} \log P(e|C) + (1 - \mathbb{1}_{e \in E^+}) \log(1 - P(e|C))$$
$$\mathcal{L} = \mathcal{L}_{\text{classif}} + \lambda \mathcal{L}_{\text{contrastive}}$$

Dans le cas où une substitution d'entités se serait effectuée sur une entité contenue dans une réponse vraie, la réponse est retirée de l'ensemble des réponses correctes (dans la limite d'avoir toujours

2. https ://txt.cohere.com/embedding-archives-wikipedia/

au moins une bonne réponse). Le tableau 1 montre que l'approche, pour $k = 5$, n'améliore pas significativement les performances sur le jeu de *test*.

Modèle	Hamming	EMR
Camembert-base	33,80	14,31
Dr-bert-7GB	**39,08**	**17,68**
Dr-bert-7GB$_{\text{contexte}}$	35,31	15,27
Dr-bert-7GB$_{\text{contrast}}$	36,06	16,40

TABLE 1 – Résultats des approches multi-classe sur le jeu de *test* ; les scores affichés sont, pour chaque modèle, la médiane sur 3 différentes seeds aléatoires

2.3 Approches multi-étiquettes

L'approche multi-étiquettes neuronale provient de deux hypothèses principales : 1) l'intégration des questions dans l'encodage peut guider le modèle vers la ou les bonnes classe(s) et 2) chaque classe doit être considérée indépendemment avec une probabilité dédiée.

Ici, on considère qu'une classe ne correspond pas à la réponse finale (par exemple, *a|c*) mais on cherche à prédire indépendamment chaque étiquette (*a* d'une part et *c* d'autre part). Cette approche entend notamment améliorer le score de Hamming, qui valorise chaque bonne réponse trouvée, et intégrer en son sein une hiérarchie et un contrôle de la sortie. Le seuil (*threshold*) à partir duquel un score de probabilité sera considéré comme une prédiction viable est un hyper-paramètre déterminant qu'il est nécessaire d'optimiser. Puisque chaque classe est prédite indépendamment, il faut intégrer pour chacune une fonction de coût de sigmoïde ainsi qu'une cross entropie binaire. Nous avons mis en place cette approche multilabel de manière non neuronale, en considérant des classifieurs plus classiques. Nous avons notamment mis l'accent sur les LGBM (Ke *et al.*, 2017) pour y rechercher les meilleurs paramètres. Malheureusement, toutes ces approches peinent à dépasser les 27% en hamming score, entraînant ainsi la nécessité de considérer des approches neuronales. Lors de la mise en place du classifieur multi-étiquettes neuronal, nous avons fait face à plusieurs obstacles dont la difficulté du choix de la représentation initiale du modèle, le choix du palier pour l'attribution des classes, l'évaluation de ces dernières, et l'équilibre entre les deux métriques d'évaluation. Pour tous nos tests, nous avons considéré l'encodage joint de la question et des réponses associées, en entraînement.

Le modèle : un *fine-tuning* à l'aide de *transformers* additionnels. Notre approche neuronale consiste en l'utilisation d'un modèle de langue comme représentation initiale, en l'occurrence un BERT Mini (Turc *et al.*, 2019; Bhargava *et al.*, 2021) adapté au domaine pharmaceutique mais aux poids non mis à jour, couplé à l'adjonction de cinq couches d'encodeurs transformers ayant chacun 8 têtes d'attention et un dropout entre les couches de 10%. Nous appliquons cet encodeur additionnel en sortie du modèle de langue, c'est-à-dire avant la couche de pooler, qui permet d'unifier les représentations à l'aide d'une fonction d'activation en tangente hyperbolique (tanh). Une particularité de notre approche est également d'utiliser deux fonctions d'activation pour casser la linéarité du modèle, la tangente et le ReLU à la suite avant un dropout de 50%. Ces paramètres extrêmes ont été estimés nécessaires compte tenu de la différence en termes de langue et de domaine. Enfin, un classifieur linéaire suivi d'une fonction sigmoïdale permet de normaliser les sorties entre 0 et 1 et ainsi d'obtenir l'équivalent d'une probabilité pour chaque classe.

Choix de la représentation initiale. Notre modèle utilise *in fine* un modèle de langue BERT Mini adapté au domaine. Définir la bonne représentation initiale à adopter est un choix déterminant eu égard aux travaux liés comme la baseline (Labrak *et al.*, 2022) et les autres approches de l'équipe. Pour ce faire, nous avons procédé à une approche empirique en partant du principe que la langue n'est pas un facteur déterminant (Labrak *et al.*, 2022). Ce choix de représentation entraîne toutefois un conflit entre les connaissances généralistes bien encodées dans le modèle de langue et les connaissances spécifiques au domaine (médecine), voire celles spécifiques à la spécialité (pharmacie). Ce conflit entraîne intuitivement une propension forte au *catastrophic forgetting* (Kirkpatrick *et al.*, 2017). C'est dans cette optique que nous avons pallié ce problème par la comparaison empirique d'une représentation vectorielle à partir de zéro (couche d'*embeddings*), à partir d'*embeddings* statiques (FastText (Joulin *et al.*, 2016)), ou encore par l'intégration d'*embeddings* contextuels avec et sans *fine-tuning* (CamemBERT, DistilCamembert, bert, etc.). De manière surprenante, ni le *fine-tuning*, ni la langue cible n'ont apporté un bénéfice certain, les modèles peinant à dépasser la barre des 30% en Hamming. C'est en mettant en place une approche d'adaptation au domaine par un *pre-training* (MLM) continu sur les données d'entraînement (Konlea & Jannidisa, 2020; Wu *et al.*, 2021) que nous avons pu passer ce seuil. Pour cela, il a été nécessaire de trouver une taille adéquate du modèle de langue compte tenu de la taille des données d'entraînement. C'est BERT Mini qui a donné les meilleurs résultats, malgré son entraînement initial réalisé uniquement sur de l'anglais. Notre phase d'adaptation permet donc non seulement au modèle de voir du français mais en même temps de s'adapter à la spécialité et au format des questions et des réponses du jeu d'entraînement. C'est ce MiniBERT adapté qui est fourni en tant que représentation initiale à notre modèle. D'autres données médicales anglaises ont été envisagées mais mises de côté par manque de temps.

Protocole expérimental. Le *threshold* choisi dans notre implémentation est l'hyper-paramètre permettant de favoriser la métrique d'EMR ou celle de Hamming. En effet, un *threshold* habituel est 0.5, mais ce dernier s'avère insuffisant, voire trop élevé. Par tests empiriques, nous sommes arrivés à la conclusion que le meilleur *threshold* était de 0.4. Ce dernier permettant au modèle d'essayer davantage d'étiquettes, il nous a permis d'augmenter significativement le score de Hamming (+ 5-7%) au détriment du score d'EMR, quasi-inexistant.

Face à la cupidité du modèle et au *threshold* bas, nous avons considéré un modèle de contrôle du comportement prédictif de celui-ci. Pour ce faire, un modèle de classification a été mis en place afin de déterminer si connaître le nombre de bonnes réponses (tâche annexe) permettrait d'augmenter l'EMR. Seul un modèle de classification binaire utilisant BERT a été mis au point par contrainte de temps. Ce modèle atteint 86% d'*accuracy*, ce qui demeure moins que l'approche par régression logistique susmentionnée, qui atteint 94% d'*accuracy* et qui a donc été choisie.

Un contrôle de la prédiction utilisant la sortie de ce classifieur binaire a été intégré. Si le type prédit est simple, alors l'étiquette avec la plus forte probabilité est prédite. Si le type prédit est multiple, alors une prédiction classique de *multi-label* est réalisée. Cette logique assez simple permet de pallier légèrement le problème du score d'EMR inexistant.

2.4 Méthode baseline avec GPT-2 & BioGPT avec traduction avec Opus

Avec l'objectif de voir s'il serait possible d'agrandir le corpus d'entraînement de certains modèles, de se servir des ressources plus vastes qui sont proposées en anglais et afin de profiter des modèles spécialisés en biologie, en partant de l'hypothèse que le vocabulaire technique comporte énormément de similarités en anglais et en français, nous avons essayé d'utiliser des modèles basés sur GPT-2

(Radford *et al.*, 2019) comme BioGPT (Luo *et al.*, 2022), après avoir effectué une traduction sur le *dataset* (originellement français), et de les comparer à sa référence.

Nous avons utilisé un modèle de traduction spécialisé dans la conversion du français vers l'anglais, Opus-MT (Tiedemann & Thottingal, 2020), et avons f*ine-tuné* les modèles GPT sur le jeu d'entraînement fourni après traduction en classification mono-label, multi-output, multi-classe, comme dans la *baseline* proposée. Avec cette approche sur GPT-2 simple, nous avons constaté un *overfitting* très tôt dans l'entraînement, peu importe les hyperparamètres utilisés, et évoqué l'hypothèse que le modèle pourrait soit manquer de connaissances sur les sujets évoqués dans les questions, soit que la tâche de classification telle que définie n'était pas adaptée à l'objectif. En effectuant la même expérience sur BioGPT, nous observons exactement le même schéma et avons décidé de supposer qu'il ne s'agissait pas d'un problème de connaissances du modèle.

2.5 Approches génératives

Galactica (1.3b) Dans cette partie, nous avons essayé de voir, avec une approche *zero-shot* sur un modèle génératif supposé spécialisé dans le domaine des sciences, si nous pouvions simplement lui faire répondre aux questions. Nous avons utilisé les méthodes avec lesquelles a été entraîné Galactica (Taylor *et al.*, 2022), qui sont spécifiées plus en détails dans le dépôt GitHub du modèle, pour lui poser les questions du sujet. Nous l'avons d'abord tenté en brut, en essayant d'extraire les réponses de la sortie du modèle génératif, qui n'était pas toujours dans le même format.

Nous avons aussi cherché s'il existait une forme de prompt plus adapté pour les questions, en tentant diverses approches, basées sur *Language Mostly Know What They Know* (Kadavath *et al.*, 2022), qui cherche à déterminer si l'on peut savoir si un modèle de langage connaît la réponse à la question, ce qui aurait pu permettre d'effacer ou d'ajuster les réponses incertaines du résultat.

BloomZ Dans cette partie, nous donnons au modèle BloomZ-176b (Muennighoff *et al.*, 2022) un prompt de la forme : « Question : ... Choix : (A) ... (B) ... (C) ... (D) ... (E) ... Réponses : »en copiant tels quels les éléments de chaque question à la place des « ... », puis nous demandons au modèle de générer les 20 tokens suivants les plus probables, sans sampling. Nous détectons ensuite dans ces 20 *tokens* les lettres majuscules A B C D E : dès qu'une lettre apparaît, la réponse correspondante est considérée comme donnée. Dans les détails, nous avons testé plusieurs variantes, presque toutes en mode *zero-shot*, donc sans utiliser le corpus d'apprentissage de DEFT :

- Nous avons testé d'autres modèles pré-entraînés qui n'ont pas subi d'*instruction fine-tuning* : Bloom-176b, Bloom-7b1 et Vicuna-13b (Chiang *et al.*, 2023). Ces 3 modèles donnent de mauvais résultats, car ils n'arrivent pas à répondre correctement à la question posée sous cette forme. La phase d'*instruction-tuning* réalisée sur BloomZ est donc primordiale pour que le modèle puisse générer une réponse sous la forme attendue.
- Nous avons testé une poignée de variantes légèrement différentes du prompt, par exemple avec ou sans espaces, retour-chariot, etc. De plus, les instructions du prompt peuvent être en anglais (« Question : », « Choices : », « Correct answers : »), tout en gardant le contenu de la question en français ; les résultats sont toujours à peu près les mêmes.
- Nous avons testé en ajoutant de 1 à 5 phrases de contexte, avant la question elle-même et en préfixant par « Contexte : ». Ces phrases en anglais sont issues de Wikipedia par une recherche neuronale sémantique via l'API de Cohere-AI, en lui donnant le texte de la question et des réponses. Les résultats sont à peu près les mêmes, légèrement meilleurs mais pas

significativement. Ceci suggère que l'ajout d'information contextuelle n'apporte rien, et donc que BloomZ-176b possède déjà les informations demandées. Nous pensons que le goulot d'étranglement en termes de performance est lié à l'interprétation de la question par le modèle.

- Ajouter 5 questions-réponses aléatoires du corpus d'apprentissage, formatées de la même manière, en mode *in-context few-shot learning* n'améliore pas les performances.
- Nous avons testé en ajoutant un unique vecteur de paramètres au début du contexte (*soft-prompt-tuning*), que nous avons appris sur le corpus d'apprentissage de DEFT. Les résultats étaient un peu meilleurs, mais très peu, et au vu du coût bien plus élevé de cette approche, nous ne l'avons pas choisie pour le modèle final.
- Nous avons fait des expériences en modifiant l'ordre des réponses, ce qui donne des résultats différents, parfois un peu meilleurs, parfois moins bons. BloomZ-176b n'est donc pas sensible aux variations mineures du prompt (y compris la langue des instructions), mais est sensible aux modifications majeures du *prompt*. Nous n'avons pas optimisé le prompt pour avoir les meilleurs résultats possibles sur le corpus de développement, mais nous avons exploité cette variabilité dans notre système final comme une forme de « mesure de confiance ».

L'approche finalement choisie est du *zero-shot* basé sur BloomZ-176b, exécutée 3 fois pour chaque question : une exécution de base, décrite précédemment, que nous appelons ABCDE ; une exécution en inversant l'ordre des réponses, EDCBA ; et une exécution en translatant l'ordre des réponses, BCDEA. Nous extrayons ensuite les réponses communes des 2 variantes EDCBA et BCDEA : Si ces réponses n'apparaissent pas dans ABCDE, alors nous les y ajoutons. La réponse finale est la réponse de ABCDE ainsi complétée. Nous avons choisi cette approche car nous avons remarqué que BloomZ favorise les réponses uniques, donc intuitivement, lorsque les 2 variantes sont d'accord entre elles, nous pouvons considérer cette réponse commune comme ayant une bonne confiance.

3 Résultats

Afin de distinguer toutes ces approches et de choisir trois d'entre elles à envoyer, nous avons utilisé le score hamming comme métrique principale. La tâche annexe ayant servi a essayer d'améliorer les résultats de la tâche principale (voir sections 2.1 et 2.3), nous avons mis nos efforts en priorité sur cette dernière. Le tableau 2 montre les résultats finaux envoyés pour le défi. Ces résultats sont conformes à ce que nous avions prédit, à l'exception de l'approche *multi-class contrastive* qui obtenait de meilleurs résultats que l'approche *multi-label* sur le corpus de validation. Il serait intéressant d'en examiner la raison, sachant que la répartition des étiquettes est similaire (voir figure 1).

4 Limites et perspectives

Les approches que nous avons menées se sont heurtées à certaines limites provenant de la spécificité de la tâche. En effet, il ne s'agit pas d'un problème de *multi-label* ou de *multi-classes* classique : les différentes classes étudiées n'ont pas de réelle continuité (les différentes réponses a, par exemple, ne partagent pas davantage de caractéristiques communes qu'avec les autres classes). L'apprentissage ne doit donc pas se faire au niveau de l'étiquette. Pour avoir un apprentissage qui se concentre davantage sur le contenu en tant que tel, on pourrait préférer une approche basée sur une *loss* de *ranking* qui renseigne sur la qualité des prédictions. Une autre méthode serait de prendre une loss

Tâche principale

Modèle	Hamming	EMR
DrBERT Multiclass contrastive	37,22	15,43
DAPT Multilabel avec type prédit (LR)	39,15	11,58
BloomZ zero-shot	**41,54**	**23,95**

Tâche annexe

Modèle	macro F1	Accuracy
LDA	13,26	19,13
DrBERT Multiclass contrastive	**31,51**	60,45
Régression logistique	27,98	**62,54**

TABLE 2 – Résultats principaux

basée sur une métrique de classification exigeante, typiquement le MCC (*Matthews correlation coefficient*) (Matthews, 1975). Cette dernière a été mise en place à l'aide d'un simili seuil par fonction sigmoïdale (Abhishek & Hamarneh, 2021) mais les résultats n'ont pas été concluants sur la tâche de *multi-label*. Malgré tout, il semble légitime de se demander s'il existe une *loss* permettant d'apprendre une telle tâche, puisque que l'expression du hamming ne permet pas de le différencier d'une manière exploitable. Pour ces raisons, les solutions présentées ici semblent n'être que palliatives.

Par ailleurs, cette tâche nécessitant des connaissances scientifiques très spécifiques, il nous semble important de fournir au modèle le plus de savoir possible sur le domaine étudié. Nous avons notamment utilisé Cohere à ces fins mais avons vu que cette piste pourrait encore être améliorée, par exemple en filtrant davantage les résultats, afin de ne pas récupérer d'informations provenant de pages Wikipédia hors domaine, ou encore en affinant la recherche au niveau des termes, pour ne récupérer que des passages mentionnant ceux présents dans les propositions de réponse. Il serait en outre intéressant de réaliser des tests plus approfondis sur la longueur des contextes fournis.

L'autre approche envisagée, l'adaptation au domaine par spécialisation d'un mini-BERT, présente plusieurs limites, à commencer par un vocabulaire non-exhaustif et un pré-entraînement sur des données en anglais. De plus, l'entraînement du modèle et donc son évaluation dépend du *threshold* qu'on fixe pour la classification *multi-label*. Une méthode évoquée mais non implémentée pourrait consister en l'utilisation d'un *threshold* approximatif en appliquant une sigmoïde (Abhishek & Hamarneh, 2021). Finalement, on observe une absence de corrélation entre la fonction de coût et la tâche finale, ce qui pose un problème pour l'évaluation du modèle. Cela rejoint le point précédent sur la difficulté de construire un entraînement pertinent, d'autant plus que dans ce cas on utilise une BCELoss qui considère des classes réelles, ce qui est fondamentalement différent de la démarche du choix des réponses possibles. Pour améliorer ce point, il serait possible de considérer l'EMR en tant que fonction de coût, ce qui a plus de sens vis à vis de la tâche.

Au final, notre équipe s'est attelée à explorer différentes familles d'approches afin d'aborder au mieux cette tâche. Ces approches n'ont toutefois pas réussi à dépasser le score obtenu en *zero-shot* par Bloomz, que ce soit en Hamming ou en EMR. Nous avons toutefois bon espoir quant au fait qu'une fonction de coût dédiée à cette tâche permettrait l'apprentissage de modèles dédiés plus efficaces.

Remerciements

Ces travaux ont bénéficié d'un accès aux moyens de calcul de l'IDRIS au travers de l'allocation de ressources 2023-AD011011668R3 attribuée par GENCI. Certaines expériences présentées dans cet article ont été réalisées sur le banc d'essai Grid'5000, soutenu par un groupement d'intérêt scientifique hébergé par l'Inria et comprenant le CNRS, RENATER et plusieurs universités ainsi que d'autres organisations.

Références

ABHISHEK K. & HAMARNEH G. (2021). Matthews correlation coefficient loss for deep convolutional networks : Application to skin lesion segmentation. In *The IEEE International Symposium on Biomedical Imaging (ISBI)*.

BHARGAVA P., DROZD A. & ROGERS A. (2021). Generalization in nli : Ways (not) to go beyond simple heuristics.

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.

DUMAIS S. T. *et al.* (2004). Latent semantic analysis. *Annu. Rev. Inf. Sci. Technol.*, **38**(1), 188–230.

HOFMANN T. (1999). Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, p. 50–57.

JOULIN A., GRAVE E., BOJANOWSKI P., DOUZE M., JÉGOU H. & MIKOLOV T. (2016). Fasttext.zip : Compressing text classification models. *CoRR*, **abs/1612.03651**.

KADAVATH S., CONERLY T., ASKELL A., HENIGHAN T., DRAIN D., PEREZ E., SCHIEFER N., HATFIELD-DODDS Z., DASSARMA N., TRAN-JOHNSON E., JOHNSTON S., EL-SHOWK S., JONES A., ELHAGE N., HUME T., CHEN A., BAI Y., BOWMAN S., FORT S., GANGULI D., HERNANDEZ D., JACOBSON J., KERNION J., KRAVEC S., LOVITT L., NDOUSSE K., OLSSON C., RINGER S., AMODEI D., BROWN T., CLARK J., JOSEPH N., MANN B., MCCANDLISH S., OLAH C. & KAPLAN J. (2022). Language models (mostly) know what they know.

KE G., MENG Q., FINLEY T., WANG T., CHEN W., MA W., YE Q. & LIU T.-Y. (2017). Lightgbm : A highly efficient gradient boosting decision tree. In I. GUYON, U. V. LUXBURG, S. BENGIO, H. WALLACH, R. FERGUS, S. VISHWANATHAN & R. GARNETT, Éds., *Advances in Neural Information Processing Systems*, volume 30 : Curran Associates, Inc.

KIRKPATRICK J., PASCANU R., RABINOWITZ N., VENESS J., DESJARDINS G., RUSU A. A., MILAN K., QUAN J., RAMALHO T., GRABSKA-BARWINSKA A. *et al.* (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, **114**(13), 3521–3526.

KLEINBAUM D. G., DIETZ K., GAIL M., KLEIN M. & KLEIN M. (2002). *Logistic regression*. Springer.

KONLEA L. & JANNIDISA F. (2020). Domain and task adaptive pretraining for language models. *Proceedings http ://ceur-ws. org ISSN*, **1613**, 0073.

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.

LUO R., SUN L., XIA Y., QIN T., ZHANG S., POON H. & LIU T.-Y. (2022). BioGPT : generative pre-trained transformer for biomedical text generation and mining. *Briefings in Bioinformatics*, **23**(6). bbac409, DOI : 10.1093/bib/bbac409.

MARTIN L., MULLER B., ORTIZ SUÁREZ P. J., DUPONT Y., ROMARY L., DE LA CLERGERIE É., SEDDAH D. & SAGOT B. (2020). CamemBERT : a tasty French language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, p. 7203–7219, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2020.acl-main.645.

MATTHEWS B. W. (1975). Comparison of the predicted and observed secondary structure of t4 phage lysozyme. *Biochimica et Biophysica Acta (BBA)-Protein Structure*, **405**(2), 442–451.

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.

NÉVÉOL A., GROUIN C., LEIXA J., ROSSET S. & ZWEIGENBAUM P. (2014). The QUAERO French medical corpus : A ressource for medical entity recognition and normalization. In *Proc of BioTextMining Work*, p. 24–30.

PEDREGOSA F., VAROQUAUX G., GRAMFORT A., MICHEL V., THIRION B., GRISEL O., BLONDEL M., PRETTENHOFER P., WEISS R., DUBOURG V., VANDERPLAS J., PASSOS A., COURNAPEAU D., BRUCHER M., PERROT M. & DUCHESNAY E. (2011). Scikit-learn : Machine learning in Python. *Journal of Machine Learning Research*, **12**, 2825–2830.

RADFORD A., WU J., CHILD R., LUAN D., AMODEI D. & SUTSKEVER I. (2019). Language models are unsupervised multitask learners.

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.

TIEDEMANN J. & THOTTINGAL S. (2020). OPUS-MT – building open translation services for the world. In *Proceedings of the 22nd Annual Conference of the European Association for Machine Translation*, p. 479–480, Lisboa, Portugal : European Association for Machine Translation.

TURC I., CHANG M., LEE K. & TOUTANOVA K. (2019). Well-read students learn better : The impact of student initialization on knowledge distillation. *CoRR*, **abs/1908.08962**.

WU H., XU K., SONG L., JIN L., ZHANG H. & SONG L. (2021). Domain-adaptive pretraining methods for dialogue understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2 : Short Papers)*, p. 665–669, Online : Association for Computational Linguistics. DOI : 10.18653/v1/2021.acl-short.84.

XIONG W., DU J., WANG W. Y. & STOYANOV V. (2020). Pretrained encyclopedia : Weakly supervised knowledge-pretrained language model. In *International Conference on Learning Representations*.

A Données

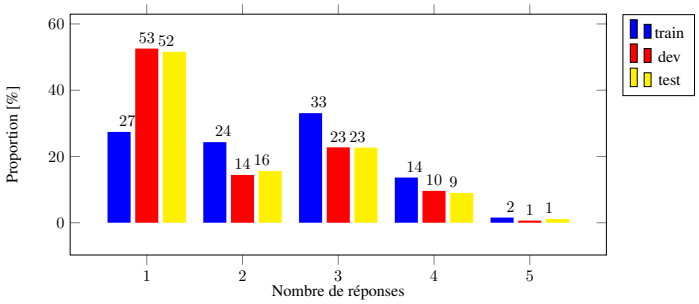

FIGURE 1 – Répartition du nombre de réponses dans les jeux de données fournis.

B Détails de modélisation

B.1 Expressions régulières (ER)

Le classifieur binaire à expressions régulières utilisé fonctionne comme suit :

- `regexp_1 = "quel|quelle|lequel|laquelle|celui|celle"`
- `regexp_2 = "quels|quelles|lesquels|lesquelles|ceux|celles"`
- `regexp_3 = "\(s\)|\(nt\)"`
- `Détecteur = match(regexp_1) ET NON match(regexp_2) ET NON match(regexp_3)`
- `Classifieur = "simple" SI Détecteur == VRAI SINON "multiple"`

B.2 *Topic modelling*

***Latent Semantic Indexing* (LSI)** Cet algorithme s'appuie sur la décomposition en valeurs singulières (SVD) de la matrice terme-document M construite à partir d'un corpus : $M = U\Sigma V^\top$, avec U et V unitaires et Σ diagonale. Chaque composante singulière figurant dans U permet de relier un terme à un concept, un **thème**. Au-delà de permettre d'identifier des thèmes dans un corpus à partir des seuls mots qui le composent, LSI permet aussi de ne retenir que les thèmes les plus saillants et d'imposer le nombre maximum de termes associé à un thème donné.

***Latent Dirichlet Allocation* (LDA)** Cet algorithme génératif suppose qu'un corpus est le fruit du tirage de mots associés à un nombre de thèmes fixé : c'est un modèle de mélange probabiliste. Les tirages sont supposés suivre une loi de Dirichlet. Ses paramètres sont calibrés par apprentissage statistique.

B.3 Exemples d'erreurs de classification

- **Exemple n° 1** : « Parmi les affirmations suivantes, une seule est fausse, indiquer **laquelle** : » ;

- **Exemple n° 2** : « Parmi les propositions suivantes, indiquer **celle** qui est exacte. Le crack est une forme : » ;
- **Exemple n° 68** « Parmi les marqueurs suivants, indiquer celui (ceux) qui reflète(nt) l'activité ostéoblastique. » ;
- **Exemple n° 2161** : « Parmi les propositions suivantes concernant Escherichia coli, quelle(s) est(sont) celle(s) qui est(sont) exacte(s) ? ».
- **Exemple n° 1188** : « Concernant la validation des méthodes d'analyse, laquelle de ces propositions est exacte ? ». Cette question appelle explicitement une unique réponse, mais la correction en donne deux.
- **Exemple n° 1585** : « Quelle est la morphologie de Fasciola hepatica ? » admet trois bonnes réponses. Si la question ne le laisse pas présupposer, les réponses possibles donnent un indice : trois thèmes semblent se dégager parmi les assertions.
- **Exemple n° 1650** : « Dans le cadre d'une enquête épidémiologique [...] » appelle une seule réponse, alors que deux sont en réalité acceptées.

B.4 Résultats détaillés

Le tableau 3 montre l'ensemble des résultats obtenus pour la tâche de classification à l'aide des approches mises en place pour la classification des questions simples et multiples.

Pour décider du modèle pré-entraîné à prendre en compte dans nos expérimentations pour la tâche de *multilabel*, nous avons comparé plusieurs modèles sans pré-supposer de la prévalence de l'adéquation de la langue de pré-entraînement. Le tableau 4 nous montre une autre preuve, s'il en fallait une, que la langue n'est pas le critère principal pour ces données de spécialité pharmaceutique. Il s'en dégage alors deux candidats : le BERT-mini et le BERT-small. Le choix du BERT-mini s'est ensuite décidé à partir de leurs performances respectives après l'application de la seconde phase de pré-entraînement par *Masked Language Modelling*. Lors de celle-ci, BERT-mini a donné les meilleurs résultats, tandis que CamemBERT adapté a semblé faire preuve de *catastrophic forgetting*. Nous supposons que cela est dû au prior très éloigné du domaine de spécialité, qu'un modèle plus compact et à l'espace vectoriel moindre dans son état caché (256 au lieu de 512) favorise une adaptation plus efficace. Bien qu'il serait intéressant de mettre en place des tests supplémentaires à ce propos, nous nous sommes limités à ces derniers dans le cadre de ce défi.

Tâche	Classe	Précision	Rappel	F_1-score	Support
Binaire (ER)	Simple	1,00	0,79	0,88	164
	Multiple	0,81	1,00	0,89	148
	Macro moyenne	0,90	0,89	0,89	312
Binaire (LDA)	Simple	0,47	0,26	0,34	164
	Multiple	0,45	0,68	0,54	148
	Macro moyenne	0,46	0,47	0,44	312
Binaire (RL)	Simple	0,99	0,90	0,94	164
	Multiple	0,90	0,99	0,94	148
	Macro moyenne	0,94	0,94	0,94	312
Multiple (LDA)	1	0,47	0,26	0,34	164
	2	0,12	0,13	0,12	45
	3	0,11	0,03	0,04	71
	4	0,00	0,00	0,00	30
	5	0,01	1,00	0,03	2
	Macro moyenne	0,14	0,28	0,11	312
Multiple (ER + LDA)	1	0,63	0,50	0,56	164
	2	0,15	0,18	0,16	45
	3	0,14	0,03	0,05	71
	4	0,00	0,00	0,00	30
	5	0,00	0,00	0,00	2
	Macro moyenne	0,18	0,14	0,15	312

TABLE 3 – Performances des différentes combinaisons d'approches (ER, LDA, RL) pour les tâches de classifications binaire (réponse simple ou multiple) et multiple (nombre de réponses attendues, de 1 à 5) sur le jeu de développement.

Modèle pré-entraîné	Couches	États cachés	Hamming↑	Loss↓
bert-tiny	2	128	0.251	0.635
bert-mini	4	256	0.269	0.624
bert-small	4	512	**0.283**	**0.619**
bert-medium	8	512	0.277	**0.619**
camembert-base	12	512	0.210	0.682
camembert-base-wikipedia-4gb	12	512	0.195	0.682

TABLE 4 – Évaluation sur les données de validation pour différentes tailles de modèles BERT pré-entraînés puis évalués sur la tâche principale.