École Polytechnique / Sorbonne Université

Étude comparative de modèles de Question-Answering

M2 Probabilité et Finance

Écrit par : NourElhouda KOUKI, Nadir AMEUR, Anass ELMOUBARAKI, Hamza LEMALES

Résumé : *Ce rapport explore plusieurs techniques de Question-Answering sur le dataset NQ-Open. Dans un premier temps, nous nous intéresserons à une approche de retrieval par questions proxy, qui consiste à identifier une question similaire dans une base de validation afin d'exploiter les documents associés pour formuler une réponse. Ensuite, nous étudierons quatres architectures de type RAG, en évaluant leur performance en fonction du modèle de langage pré-entraîné utilisé. L'accent sera mis sur le prompt engineering mais aussi sur le choix de la base documentaire (fichier Lucene ou Faiss). Enfin, nous analyserons les résultats obtenus à travers les métriques F1-score et Exact Match.*

Mots clefs : *Question-Answering, RAG, LLM, Prompt Engineering*

Table des matières

1 Retrieval par question proxy

1.1 Description

Cette méthode tire parti des embeddings des 3610 questions de la base de validation, préalablement calculés à l'aide d'un modèle de similarité de phrases, ici all-MiniLM-L6-v2, disponible sur Hugging Face.

Lorsqu'une nouvelle question est posée :

- Son embedding est calculé et comparé à ceux de la base de validation à l'aide de la distance cosinus.
- La question de validation la plus proche est sélectionnée.
- L'article le plus pertinent pour répondre à cette question est récupéré grâce au module FakeSearch.
- Ce document est ensuite fourni à un LLM (ChatGPT-3.5 Turbo) chargé de donner une réponse à la question initiale en se basant uniquement sur le contenu de ce dernier.

1.2 Méthodologie

Tout d'abord, si l'on se contente des extraits et non des documents, on n'obtient aucun résultat intéressant. Une étape essentielle consiste donc à retrouver les documents complets.

1ère idée : En se référant à l'annexe, on constate que l'index précalculé est wiki_dpr. D'après la documentation disponible sur Hugging Face, le titre de l'article est récupérable via le docid ou le texte. Toutefois, quelle que soit la version, le dataset est trop volumineux pour être téléchargé intégralement, et une exploration séquentielle s'est révélée trop lente. De plus, aucun paramètre ne permet de charger directement une entrée spécifique.

2ème idée : Utiliser l'API de Google pour retrouver l'article à partir de l'extrait, ceux-ci étant souvent bruités voire obsolètes, puis exploiter l'API de Wikipedia pour en extraire le contenu. Cependant, dès que nous avons voulu tester cette méthode à plus grande échelle, notre IP a rapidement été bannie par Google.

3ème idée : Utiliser l'API de ChatGPT pour retrouver les pages, car le modèle a déjà été entraîné dessus, avant de se servir de l'API de Wikipedia pour extraire le contenu de l'article correspondant. C'est cette approche que nous adopterons par la suite.

La deuxième étape est celle du prompt engineering. Étant donné que la métrique choisie, "exact match", est très restrictive, il est crucial d'aligner le format des réponses avec celui du dataset. Cependant, ce dernier manque de cohérence avec ses propres règles, rendant tout post-traitement compliqué à mettre en place.

Remarque : Nous parlons dans le paragraphe précédent de détection de motifs récurrents dans les réponses du dataset : c'est typiquement une tâche de deep learning. Une idée intéressante serait d'explorer comment finetuner un modèle préentraîné avec un entraînement basé sur des paires (réponse initiale, réponse corrigée) afin d'apprendre un mapping des réponses brutes vers un format plus conforme aux attentes.

Ainsi, en appliquant des principes fondamentaux (donner des instructions claires, utiliser des délimiteurs pour structurer l'input, fournir directement les données et inclure quelques exemples

pertinents), nous aboutissons au prompt suivant qui est un bon compromis entre performance et réduction des hallucinations du modèle (avec la température réglée à 0).

```
You are an AI assistant that extracts **only precise answers** from
   provided excerpts.
You must **strictly** rely on the given excerpts to answer the question.

### **Expected Answer Format:**
- Your answer **must be inside square brackets**: `['answer']`.
- Respond only with a word or a phrase, never a full sentence.
- If the answer **cannot** be determined from the excerpts, respond with
    **['Error']**.

### **Question:**
{question}

### **Excerpt:**
{excerpt}

### **Examples:**
when was the last summer olympics in the usa  --> ['1996']
who sings blame it on the bossa nova --> ['Eydie Gorm ']
how many seasons has greys anatomy been on tv --> ['14']
where does sex and the city take place --> ['New York City']
do you need a permit or license to own a gun in florida --> ['No']
what are the five compulsory prayers in islam --> ['Fajr', 'Dhuhr', 'Asr
   ', 'Maghrib', 'Isha']

Now, provide your answer:
```

FIGURE 1 – Prompt utilisé pour la génération de réponses.

1.3 Résultats

Nous allons tester notre approche sur 1000 questions de la base de test (différentes de celles de la base de validation) en utilisant d'abord Lucene, puis Faiss, pour sélectionner les documents pertinents. Nous comptabiliserons également à la main, pour chacun des deux, le nombre d'hallucinations commises par notre modèle parmi les bonnes réponses.

Data	Précision	Nombre d'hallucinations
Lucene	0,082	19/82
Faiss	0,09	21/90

TABLE 1 – Retrieval par question proxy

1.4 Discussion

Les résultats obtenus montrent que notre approche peine à atteindre un niveau de précision satisfaisant, ne dépassant pas les 10 %. Cette performance limitée s'explique en partie par le choix

de la métrique d'évaluation, l'exact match, qui impose une correspondance stricte avec les réponses attendues. Cependant, au-delà de cette contrainte, il apparaît que le dataset de validation n'est pas suffisamment représentatif de l'ensemble des questions du dataset NQ-Open, ce qui est rédhibitoire pour notre modèle.

Par ailleurs, nous avons constaté un taux d'hallucination d'environ 20%, ce qui n'est pas surprenant car, comme mentionné précédemment, il a été entraîné sur les données de Wikipedia donc quand le document est proche de contenir la réponse mais qu'elle n'y figure pas, il va avoir tendance à 'bien' extrapoler.

2 Approche RAG

2.1 Méthodologie

Notre approche RAG repose sur le module Fakesearch, qui facilite la recherche d'informations en attribuant à chacune des 3 610 questions du dataset de validation (NQ-Open) une liste de 10 extraits, classés selon un score de pertinence. Ces résultats sont organisés en deux dictionnaires distincts, correspondant à des modèles de recherche différents : Lucene (sparse) et FAISS (dense).

À partir d'un dataset de 500 questions de validation, nous testons notre méthode en utilisant quatre modèles que nous présenterons par la suite. Pour chaque question, nous fournissons au modèle les extraits sélectionnés, puis nous comparons la réponse produite avec la réponse attendue.

Pour étudier leur performance, nous nous baserons sur deux métriques : le F1-score et l'Exact Match, une métrique binaire vérifiant si la réponse générée correspond exactement à la réponse réelle. Nous prendrons également en considération le temps d'exécution.

Enfin, l'exécution de nos algorithmes se fait sur Google Colab, permettant l'utilisation du GPU et la parallélisation des tâches.

2.2 Présentation des modèles

2.2.1 Modèle MdeBerta

Le modèle MDeBERTa de Microsoft améliore légèrement les modèles BERT classiques en intégrant le mécanisme de disentangled attention, qui prend en compte à la fois les mots et leur position. Cette approche permet de mieux comprendre des formulations complexes et de distinguer les nuances, ce qui en fait un bon candidat pour du question-answering.

2.2.2 Modèle DistilBert

DistilBERT est une version allégée de BERT, conservant 95% de ses performances tout en étant 60% plus petit et 40% plus rapide. C'est une alternative plus rapide que le modèle précédent.

2.2.3 Modèle T5

Pour avoir un modèle avec une meilleure robustesse, nous explorons T5 (Text-to-Text Transfer Transformer), pré-entraîné sur le Question-Answering via une approche de Span Correction. Conçu pour traiter toutes les tâches NLP sous forme texte-texte, T5 modélise finement la structure linguistique et capture des relations complexes entre les mots, lui permettant de mieux s'adapter aux reformulations.

2.2.4 Modèle GPT-3.5-turbo

Le modèle GPT-3.5 Turbo d'OpenAI, optimisé pour la compréhension et la génération de texte, traite des requêtes complexes tout en s'adaptant dynamiquement au contexte. Il excelle dans la recherche d'information et la génération de réponses cohérentes, même face aux variations de formulation.

2.3 Résultats

Nous évaluons les performances des modèles MdeBERTa, DistilBERT, T5 et GPT-3.5 Turbo à l'aide de deux métriques. Exact Match (EM) mesure le pourcentage de réponses parfaitement correctes, tandis que le F1-score évalue leur similarité avec les réponses attendues. Nous regardons également le temps d'exécution sur les 500 questions de validation, ainsi que l'impact du nombre d'extraits explorés sur la qualité des réponses.

Top k	Retrieval	EM	F1	Time(s)	Time
3	lucene	12.61744966442953	18.0713097491621	2072.183930158615	34min 32s
5	lucene	13.6	20.016863136863147	3513.1569242477417	58min 33s
3	faiss	19.6	28.118926628926634	1977.5556542873383	32min 57s
5	faiss	21.4	29.502394272394284	3481.6667540073395	58min 1s

Table 2 – Performances du RAG alimenté par le modèle MdeBerta

Top k	Retrieval	EM	F1	Time(s)	Time
3	lucene	10.63063063063063	15.3979863979864	605.6071989536285	10min 5s
5	lucene	11.0	16.920209790209793	1015.801837682724	16min 55s
3	faiss	16.8	24.06189754689755	573.2882468700409	9min 33s
5	faiss	17.2	23.76014557338087	994.9214992523193	16min 34s

Table 3 – Performances du RAG alimenté par le modèle DistilBert

Top k	Retrieval	EM	F1	Time(s)	Time
3	lucene	12.4	18.477619047619047	5224.757395982742	1h 27min 4s
5	lucene	12.4	18.67619047619048	5494.538256645203	1h 31min 34s
3	faiss	22.2	27.86142857142858	5152.478444099426	1h 25min 52s
5	faiss	23.2	29.276666666666664	5462.092473268509	1h 31min 2s

Table 4 – Performances du RAG alimenté par le modèle T5

Top k	Retrieval	EM	F1	Time(s)	Time
3	lucene	19.6	26.856191503714744	309.29626631736755	5min 9s
5	lucene	24.2	32.046366736119076	301.61009669303894	5min 1s
3	faiss	39.8	49.05323393004199	408.04129910469055	6min 48s
5	faiss	41.2	51.45220659118384	304.824059009552	5min 4s

Table 5 – Performances du RAG alimenté par le modèle GPT

2.3.1 Discussion

L'analyse des résultats met en évidence l'importance du seuillage et du nombre de documents considérés, avec une amélioration constante des scores F1 et Exact Match en passant de 3 à 5 documents, malgré un coût computationnel accru. L'approche dense surpasse systématiquement l'approche probabiliste, confirmant la nécessité d'un scoring robuste pour maximiser la pertinence des extraits sélectionnés.

Hiérarchie des modèles :

- **MdeBERTa** : Performances faibles avec 21% d'Exact Match et 30% de F1-score, et un temps d'exécution élevé (30 min pour 3 extraits par question, 1h pour 5).
- **DistilBERT** : Trois fois plus rapide que MdeBERTa, il conserve 80 à 90% de sa performance, le rendant plus attractif en termes de compromis temps/précision.
- **T5** : Légère amélioration sur Exact Match (+1 à 2 %), mais une complexité temporelle élevée (5 questions/minute), le rendant moins efficace que DistilBERT.
- **GPT-3.5 Turbo** : Meilleure performance globale, avec un Exact Match de 41.2 % (soit 2x plus précis que T5), et 2x plus rapide que DistilBERT. Il représente ainsi le meilleur compromis robustesse/efficacité pour notre tâche.

Voici le premier prompt que nous avons utilisé pour GPT-3.5 Turbo :

```
"""
    You are a question-answering system.
    You have access to the following context (excerpts from documents):

    CONTEXT:
    {context}

    INSTRUCTIONS:
    1. Read the context carefully.
    2. Extract only the words from the context that answer the question below.
    3. Make your answers as short and concize as possible.
    4. Do not add any additional words or phrases that do not appear in the
       context.
    5. When there are multiple correct answers, choose only one of them.
    6. If the context does not contain the answer, return "No answer found."

    QUESTION:
    {question}

    YOUR ANSWER (give short and concise answers):"""
```

Pour renforcer encore la précision de GPT-3.5, nous intégrons des techniques avancées de prompting, combinant :

- **Few-Shot Prompting** : ajout d'exemples contextuels pour guider la génération des réponses.
- **Chain of Thought Reasoning** : incitation au raisonnement explicite pour améliorer la cohérence des réponses.

Cette approche est appliquée dans la meilleure configuration identifiée (retrieval dense avec 5 extraits par question). Voici le prompt amélioré :

```
few_shot_examples = [
    {
        "context": "John loves tennis. Mary is from Sweden.",
        "question": "Where is Mary from?",
        "answer": "Sweden"
    },
    {
        "context": "Movie X was filmed in Budapest, Paris, Tunisia, New York.",
        "question": "Where was Movie X filmed?",
        "answer": "Budapest"
    }
]
```

```
prompt = f"""
You are a question-answering system. When possible, you copy the exact words
   from the provided context to answer the question.
Below are some examples of how you should answer:

{few_shot_portion}

You have access to the following context (excerpts from documents):

CONTEXT:
{context}

INSTRUCTIONS:
    1. Read the context carefully.
    2. Extract only the words from the context that answer the question below.
    3. Make your answers as short and concise as possible.
    4. Do not add any additional words or phrases that do not appear in the
       context.
    5. When there are multiple correct answers, choose only one of them.
    6. If the context does not contain the answer, return "No answer found."

Here is your real question:

QUESTION:
{question}

YOUR ANSWER:
"""
```

```

# Call GPT with chain-of-thought
    response = openai.chat.completions.create(
        model="gpt-3.5-turbo",  # or "gpt-4" if you have access
        messages=[
            {"role": "system", "content": "You are a helpful assistant strictly
               extracting answers from context that does your reasoning silently
                and then present only a short final answer."},
            {"role": "user", "content": prompt}
        ],
        max_tokens=50,  # Increase if you need more tokens for reasoning
```

```
        temperature=0.0
    )

    # Extract the chain-of-thought + final answer from the response
    predicted_answer = response.choices[0].message.content.strip()
```

Grâce à ces optimisations, le temps d'exécution est réduit à 3 minutes 16 secondes, offrant une accélération significative par rapport aux configurations précédentes. De plus, la métrique Exact Match augmente de 2%, confirmant l'impact positif de ces améliorations sur la précision du modèle.

3 Conclusion et perspectives

Dans cette étude, nous avons comparé différentes approches pour la tâche de Question-Answering et avons observé que le meilleur modèle était GPT-3.5 Turbo. Parmi les pistes d'amélioration, nous aurions pu explorer une approche multimodale, qui vise à exploiter simultanément plusieurs modalités de manière intégrée pour renforcer la pertinence des réponses générées.

Dans notre cas, les documents et les questions seraient représentés sous forme de vecteurs d'embeddings dans un espace latent partagé. Cet espace constituerait une représentation vectorielle unifiée, où la proximité entre les vecteurs traduirait directement leur pertinence sémantique. Ainsi, une question et les documents contenant des réponses pertinentes se retrouveraient naturellement proches, facilitant une recherche optimisée et un meilleur alignement question-document.

Pour mettre en œuvre cette approche, deux fonctions d'embedding seraient nécessaires. Une pour projeter les documents dans l'espace latent et une autre pour projeter les questions dans ce même espace. De plus, l'entraînement reposerait sur une approche contrastive, rapprochant les paires positives (questions associées à leur document le plus pertinent parmi les 10 plus proches voisins) et éloignant les paires négatives (questions associées à des documents non pertinents, exclus des 10 plus proches voisins).

Cependant, cette approche nécessiterait un fine-tuning des modèles d'embedding, ce qui impose des contraintes techniques difficiles à surmonter sans ressources adéquates. Ainsi, bien qu'elle représente une piste prometteuse, son implémentation exigerait une infrastructure plus importante.