

الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي والبحث العلمي

People's Democratic Republic of Algeria

Ministry of Higher Education and Scientific Research

University of Algiers 1 Benyoucef BENKHEDDA



Faculté des Sciences

Département de Mathématiques et Informatique

Mémoire de fin d'étude pour l'obtention du diplôme de Master en informatique

Spécialité : Ingénierie des Systèmes Informatiques Intelligents

Présenté par :

M. AGABI Rayane Younes

Melle. TIDAFI Asma

Thème

YouTaQA : Système de Questions-Réponses Intelligent basé sur le Deep Learning et la Recherche d'Information

Devant le jury composé de :

Mme. M.....	Pr. Université Alger 1	Président
Dr. ZIANI Amel	MCB. Université Alger 1	Encadrant
Dr. BOUADJENEK Mohamed Reda	Pr. Deakin University	Co-Encadrant
Mme. M.....	MCA. Université Alger 1	Examinateur
Mme. M.....	Professeur. Université Alger 1	Examinateur

Soutenu le/..../2020

Remerciements

Nous remercions tout d'abord le tout puissant ALLAH qui nous a toujours comblé de ses bienfaits et à qui nous adressons nos remerciements pour sa grâce infinie pour nous avoir éclairé et aidé dans la préparation et la réalisation de cette thèse.

En second lieu, nos reconnaissances et nos vifs remerciements vont particulièrement à nos encadrants M. Mohamed reda Bouadjenek et Mme. Amel Ziani qui ont bien voulu accepter de diriger et d'encadrer ce travail, également pour leur patience, leurs sacrifices, leurs conseils et l'aide qu'ils nous ont fourni tout au long de notre stage qui nous a été d'une grande utilité.

Nous remercions l'université de Deakin pour nous avoir donné l'opportunité d'effectuer un stage au cours de notre dernière année Master, ainsi que tout le personnel qui nous ont apporté aide et assistance et donné toutes les informations dont nous avions besoin pour la réalisation de cette thèse.

Nous présentons notre gratitude aux membres du jury qui ont bien voulu examiner et évaluer notre travail et qui nous font l'honneur de participer à la soutenance.

Nos remerciements s'adressent aussi à tous les enseignants de l'université d'Alger 1 Ben Youcef Ben Khedda qui nous ont formé durant ces cinq dernières années.

Dédicaces

Ce travail est dédié à ma très chère maman et au meilleur des pères. Grâce à leurs tendres encouragements et leurs grands sacrifices, ils ont pu créer le climat affectueux et propice à la poursuite de mes études. Aucune dédicace ne pourrait exprimer mon respect, ma considération et mes profonds sentiments envers eux. Je prie le bon Dieu de les bénir, de veiller sur eux, en espérant qu'ils seront toujours fiers de moi.

A mes sœurs et mes frères qui m'ont soutenu durant tout mon cursus.

A mon encadrant Dr.BOUADJENEK Mohamed Reda qui a toujours été présent et m'a donnée les meilleurs conseils pour pouvoir compléter le travail comme je l'ai toujours souhaité, je le remercie pour sa patience, son aide et la confiance qu'il nous a témoignée.

TIDAFI Asma.

Dédicaces

A ma chère maman, qui a œuvré pour ma réussite, de par son amour, son soutien, ses précieux conseils ; je ne pourrai jamais la remercier assez pour toute sa présence dans ma vie. Reçois à travers ce travail aussi modeste soit-il, l'expression de mes sentiments et de mon éternelle gratitude, je t'aime ma meilleure.

À mon cher père, qui n'a jamais cessé de m'encourager ni de me guider tout au long de mes études ; je le remercie infiniment pour ses sacrifices consentis et pour des valeurs nobles qu'il m'a apprises, l'éducation et le soutien permanent venant de sa part, je t'aime mon meilleur.

À ma grande soeur Asmaa, tu as été à mes côtés pendant toutes les étapes de ce travail, je t'en suis très reconnaissant. Je te dédie ce travail en témoignage de ma profonde affection en souvenirs de notre indéfectible union qui s'est tissée au fil des jours.

À ma petite soeur Maroua, une sœur comme on ne peut trouver nulle part ailleurs, Puisse Allah te protéger, garder et renforcer notre fraternité. Je te souhaite tout le bonheur du monde.

À ma tante Djamila, celle qui a toujours joué le rôle d'une deuxième maman pour moi, et qui m'a toujours soutenu, ainsi que mes tantes Noria, Dehbia et Radia et à mon oncle Mohamed.

Votre soutien, votre gentillesse sans égal, vos profonds attachements, vos conseils et encouragements m'ont motivé dans les moments les plus difficiles. À mon cher oncle Karim AGABI et Tata Evelyne.

À mon cousin Iheb Tekkour qui est un grand frère pour moi, ma tante Nassira , papa Ahmed, Saliha, Soumia et Islem Boulacheb que j'aime beaucoup ainsi que toute ma grande famille que j'aime.

À mes professeurs du primaire, CEM, Lycée et de l'université particulièrement Mme. Bassai, Mme. Aoudia, M. Krouri, M. Guernah, Mme. Louati, Mme. Touil, Mme. Taibouni, M. Zemali, M. Derias, M. Abbas, M. Tali et M.Boutaleb, je vous remercie d'avoir enrichi mes connaissances et de m'avoir guidé durant tout mon parcours étudiantin.

La passion pour votre travail est contagieuse ! C'est avec un réel plaisir que j'ai travaillé avec vous et que je vous ai eu comme encadrant. Ce travail est dédié à Bouadjenek Mohamed Reda.

À mon binôme TIDAFI Asma et à toute sa famille pour tout ce qu'on a partagé durant notre stage.

À tous mes amis : Mehdi Belhoucine, Akram Arar, Mounir Grar, Yazid AitAlala, Fares Aliliche, Oussama Hamada, Rayane Krimi, farid belmareg, Khaled Chenouf, Islem Krim, Chakib Kessai, Rami Naidji, Amine Yahouni, Anis Amrouche, Abdelfetah fetouhi ainsi que tous mes amis que je n'ai pas pu citer, je vous remercie d'avoir toujours été là pour moi.

À la toute première promo MI de la fac centrale 2015/2016, elle a été sans aucun doute la meilleure promo, pleine d'énergie et de collaboration, je vous souhaite du bonheur et de la réussite dans vos vies.

AGABI Rayane Younes.

Résumé

Le besoin des utilisateurs du confort et la demande d'avoir des réponses exactes à leurs questions sont présents de nos jours ce qui a donné un nouvel objectif à l'intelligence artificielle. Les moteurs de recherches les plus connus comme Google tendent à offrir une brève réponse aux questions dites «factoid». Cette tâche est considérée difficile en terme de complexité des requêtes voire leurs réponses qui peuvent être la combinaison de plusieurs passages.

Pour ceci, dans cette thèse, notre objectif repose sur la conception et réalisation d'un système de questions-réponses pouvant surpasser les difficultés citées et qui est apte de répondre aux questions dans plusieurs domaines de façon exacte et précise en utilisant la base de connaissances de Wikipédia. Le système réalisé durant ce travail est nommé YouTAQA et qui commence par la collecte des passages qui peuvent répondre à la requête entrée par l'utilisateur et termine par faire l'extraction du début et la fin de la réponse exacte en utilisant l'apprentissage approfondi. Ceci dit, notre système représente un pipeline complet, à partir de la collecte des passages pertinents, jusqu'à l'extraction de la réponse finale en prenant la question comme entrée. Les modules d'apprentissage approfondi du système proposé ont été implémentés en utilisant le modèle pré-entraîné BERT qui a été conçu pour réaliser différentes tâches de traitement du langage naturel.

Le moteur de recherche a pu atteindre un score MRR et MAP de **0.20** ce qui veut dire que le bon document se trouve en moyenne dans l'un de cinq premiers résultats produits. Le modèle de classification des passages a été entraîné en utilisant le dataset QNLI et qui a atteint une accuracy de **78%** et une précision de **80%** tandis que le modèle d'extraction de réponse a affiché un exact match de **87%** et un F1 score de **92%**.

Les expérimentations sur l'ensemble de données proposé démontrent l'efficacité de la méthode proposée, et les résultats de la comparaison montrent que l'architecture proposée donne un plus au domaine du Question-Answering.

Mots clés : Recherche d'Information, Apprentissage Approfondi, Traitement de langage naturel, Bidirectional Encoder Representations from Transformers, Apprentissage par transfert.

Table des matières

Introduction générale	1
Contexte générale	1
Problématique et motivation	2
Contribution	3
Plan du mémoire	4
1 Etat de l'art	5
1.1 Introduction	5
1.2 Les systèmes de Questions-Réponses	5
1.3 Classification des systèmes	6
1.3.1 Classification par domaine d'application	6
1.3.2 Classification par source de données	8
1.3.3 Classification par type de questions	8
1.4 Les jeux de données disponibles	11
1.5 Conclusion	12
2 Généralités	13
2.1 Recherche d'information	13
2.1.1 Les modèles RI	14
2.1.2 Les métriques d'évaluation	16
2.1.3 Outil de développement	19
2.2 Deep Learning en Traitement du Langage Naturel	20
2.2.1 Le mécanisme d'attention	21

TABLE DES MATIÈRES

2.2.2	Les Transformateurs	21
2.2.3	BERT (Bidirectional Encoder Representations from Transformers)	24
2.2.4	Keras	26
2.2.5	Les métriques d'évaluation	26
2.3	Conclusion	27
3	Conception et implémentation de YouTaQA	28
3.1	Introduction	28
3.2	Architecture globale du système YouTaQA	28
3.3	Le choix des jeux de données	29
3.3.1	SQuAD	29
3.3.2	Wikipedia	30
3.4	Moteur de recherche MRI	31
3.4.1	Pré-traitement de la base Wikipédia	31
3.4.2	Indexation des articles	32
3.4.3	Méthodes de recherche adoptées	34
3.5	Module de classification MC	34
3.6	Module d'extraction de réponses MER	36
3.7	Déploiement du système YouTaQA	37
3.8	Conclusion	38
4	Analyse et discussion des résultats	39
4.1	Introduction	39
4.2	Prétraitement et fractionnement des données	39
4.3	Résultats du module de recherche d'information MRI	40
4.3.1	Méthodes de recherche employées	40
4.3.2	Discussion des performances des méthodes de recherche	41
4.4	Résultats du module de classification MC	43
4.5	Résultats du module d'extraction des réponses MER	45
4.6	Résultats globaux	47
4.7	Conclusion	48

TABLE DES MATIÈRES

Conclusion Générale	49
Perspectives	50
Bibliographie	51

Table des figures

0.1 Schéma global du système YouTaQA	3
1.1 La taxonomie de l'état de l'art des QAS.	7
2.1 Processus de recherche d'information [Baeza-Yates and Ribeiro-Neto, 2011].	15
2.2 Architecture de base des transformateurs [Tra, 2018].	22
2.3 Architecture de l'encodeur du transformateur [Tra, 2018].	23
2.4 Principe de self-attention [Sel, 2020].	24
2.5 Représentation des entrées et sorties du modèle BERT [Devlin et al., 2018].	25
3.1 Schéma global du système YouTaQA	29
3.2 Structure XML d'un article Wikipédia	31
3.3 Arborescence des fichiers XML.	32
3.4 Schéma représentatif des sections d'un article Wikipédia	33
3.5 La phase de tokenisation des entrées.	34
3.6 La classification du texte avec BERT.	35
3.7 Extraction du début et fin de la réponse avec BERT.	36
3.8 Capture d'écran de l'application web YouTaQA	38
4.1 Histogrammes d'évaluation du Module de Recherche d'Information.	41
4.2 Graphs d'évaluation du Module de Recherche d'Information.	42
4.3 Graphes d'évaluation du Module de Classification.	44
4.4 Matrices de confusion du MC (Train/Validation).	44
4.5 Matrice de confusion du MC (Test).	45

TABLE DES FIGURES

4.6	Graphes d'évaluation du Module d'Extraction de Réponse (Start).	46
4.7	Graphes d'évaluation du Module d'Extraction de Réponse (End).	46
4.8	Exemple d'utilisation du système YouTaQA	47

Liste des tableaux

1.1	Classification des QAS selon plusieurs axes	10
1.2	Les différents jeux de données disponibles.	11
2.1	Matrice de confusion.	17
4.1	Le fractionnement du jeu de données SQuAD.	39

Liste des abréviations

API	Application Programming Interface.
BERT	Bidirectional Encoder Representations from Transformers.
BOW	Bag Of Words.
CDQAS	Closed-Domain Question-Answering Systems.
DL	Deep Learning.
EM	Exact Match.
GPT	Generative Pre-trained Transformer.
IR	Information Retrieval.
KB	Knowledge Bases.
LSTM	Long Short-Term Memory.
MAP	Mean Average Precision.
MC	Module de Classification.
MER	Module d'Extraction des Réponses.
MRI	Module de Recherche d'Informations.
MRR	Mean Reciprocal Rank.
NLIDB	Natural Language Interface to DataBases.
NLP	Natural Language processing.
ODQAS	Open-Domain Question-Answering Systems.
QAS	Question-Answering System.
QNLI	Question Natural Language Inference.
RI	Recherche d'Information.
RNN	Recurrent Neural Network.
SPA	Single Page Application.
SQAS	Social Question-Answering Systems.
SQuAD	Stanford Question Answering Dataset.
TF-IDF	Term Frequency - Inverse Document Frequency.
TREC	Text REtrieval Conference.
UI	User Interface.
VSM	Vector Space Model.
XML	eXtensible Markup Language.

Introduction générale

Contexte générale

L'un des principaux défis de l'informatique est de construire des systèmes plus intelligents et capables de comprendre les êtres humains sans qu'on leur dise explicitement ce qu'ils doivent faire. Depuis les années 60, une percée majeure dans ce domaine se présente sous la forme de systèmes Questions-Réponses (Question-Answering Systems ou QAS). Un QAS est, comme son nom l'indique, un système qui peut répondre à des questions au lieu d'encombrer l'utilisateur avec des documents ou même des passages correspondants, comme le fait la plupart des systèmes de recherche d'information basiques [Ojokoh and Adebisi, 2019].

Dès leur début, les majeurs défis des QAS sont la précision, l'habileté à répondre à toutes les questions complexes correctement avec une performance semblable à celle des humains. Pour avoir une vision plus claire sur les systèmes questions-réponses actuels, prenons d'abord un moment pour comprendre la structure du problème et pourquoi les solutions existantes ne sont pas tout à fait suffisantes pour répondre à des questions complexes. Les QAS sont généralement classés en deux grandes catégories : les QAS pour le domaine ouvert ODQAS et les QAS pour le domaine fermé CDQAS (voir la section 1.3).

En ce qui concerne la source de connaissances¹ des QAS et la façon avec laquelle ces derniers s'en servent, plusieurs approches ont vu le jour durant l'évolution des techniques et des sources de données. Parmi ces approches, nous trouvons les QAS basés sur le texte, les faits, le Web et la recherche d'information (Information Retrieval ou IR) [Mervin, 2013]. Pour notre modèle, nous allons opter pour l'approche de la recherche d'informations dans une collection de données basée sur les articles de Wikipedia² seule-

1. **Source de connaissances** ("Knowledge source" en anglais) : C'est la source dans laquelle les QAS fouinent à la recherche d'une réponse à une question donnée

2. <https://www.wikipedia.org/>

ment. La recherche d'informations, contrairement aux autres approches, utilise des sources de données qui ne sont pas forcément structurées ce qui permet une meilleure flexibilité dans le cas d'ajout et d'extension des sources de recherche.

Problématique et motivation

De nos jours, suite à l'utilisation croissante des appareils mobiles, tels que les smartphones, pour accéder à l'information et recevoir des réponses directes à des questions pour laquelle les requêtes traditionnelles consistant à spécifier des mots-clés ne sont pas très conviviales ; la tâche de réponse aux questions de manière précise est devenu l'une des fonctions les plus désirables pour les consommateurs d'information.

La majorité des connaissances humaines qui représentent les besoins d'informations détaillés d'un utilisateur sont uniquement représentées par le langage naturel. Ils sont accessibles aux humains, qui peuvent comprendre les textes en langage naturel et répondre à des questions relatives à leur contenu, mais ne sont pas accessibles et compréhensibles pour les machines. Ces dernières ne peuvent donc pas comprendre et interpréter les énoncés des requêtes en langage naturel.

La tâche de l'extraction automatisée d'informations spécifiques à partir d'une source de connaissances, en tant que réponse à une question en langage naturel, n'est pas simple, même pour des ressources d'informations relativement réduites. La question doit être représentée comme une requête et la réponse doit être courte et précise. Nous pouvons extraire des informations factuelles explicites à partir d'un texte, mais l'extraction d'informations conceptuelles qui nécessitent également une compréhension du discours reste un objectif complexe. Pour obtenir des réponses précises, il faut formuler le besoin d'informations de manière exacte et bien exprimée [Kolomiyets and Moens, 2011a], au-delà d'un petit ensemble de termes vagues, comme c'est généralement le cas pour la recherche de documents. Cette dernière fait d'une part la réduction des requêtes en langage naturel à des recherches basées sur des mots-clés. D'autre part, les bases de connaissances sont interrogées avec des requêtes claires obtenues à partir des questions en langage naturel, et les réponses sont obtenues par raisonnement.

Le langage naturel est ambigu (une phrase peut avoir un ou plusieurs sens) et syntaxiquement riche car un seul et même sens peut être véhiculé par de nombreuses expressions du langage naturel. La tâche de trouver une réponse à une question, lorsque les deux sont en langage naturel repose d'abord sur l'utilisation des techniques de recherche d'information pour sélectionner les passages pertinents. Ensuite extraire des passages courts (contextes) suite à une classification par rapport à la probabilité de leur pertinence et de

l'existence de l'information recherchée. Le système doit retourner uniquement les informations qui ont été spécifiquement demandées. Or, les demandes peuvent être complexes et narratives, ce qui signifie qu'il sera plus difficile pour le QAS d'y répondre avec précision. De plus, les passages peuvent provenir de différents documents, nous devons donc les combiner pour fournir des réponses pertinentes, il se peut alors que nous ayons besoin d'un raisonnement complexe. Il sera donc difficile de formuler des réponses en langage naturel.

Contribution

Nous contribuons par ce projet proposé par l'université de **Deakin**³ (Située à Victoria, Australie) à la mise en place d'un système QAS automatique complet en commençant par un moteur de recherche, passant par un classifieur de documents jusqu'à l'extraction des réponses, ce pipeline à pour but d'offrir un service de questions-réponses exhaustif nommé YouTaQA (figure 3.1).

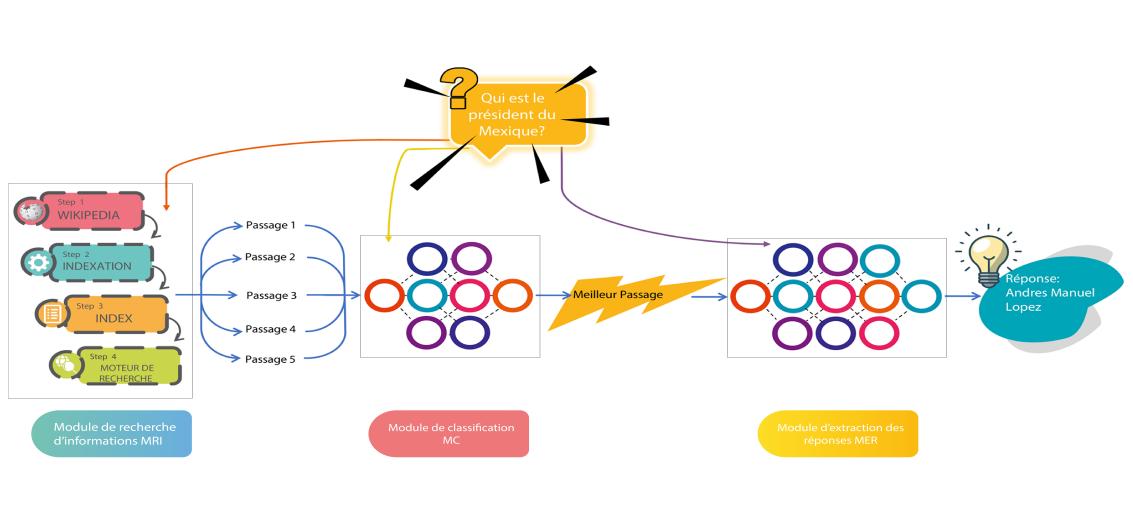


FIGURE 0.1: Schéma global du système YouTaQA

Il existe des systèmes questions-réponses qui ont le même but global, mais qui se contentent d'offrir la partie extraction des données à leurs utilisateurs en les obligeant à fournir les documents nécessaires ce qui n'est pas vraiment pratique, notre système sera donc une version améliorée de ce qui existe, en permettant aux utilisateurs d'avoir des réponses exactes à leurs questions uniquement en se basant sur notre moteur de recherche, ce qui épargnera à l'utilisateur de fournir autre chose que la question, et éventuellement leur

3. <https://www.deakin.edu.au/>

faciliterai la tache.

Durant ce travail, nous nous sommes concentrés sur les interactions entre l'extraction des réponses à l'aide du Deep Learning, le traitement du langage naturel et la recherche d'informations. Plus précisément, notre but est de mettre en œuvre une architecture générale d'un QAS en utilisant des collections et des ensembles de données de référence sur lesquels nous comptons baser les réponses du système.

Plan du mémoire

Ce présent manuscrit de thèse de Master est composé de quatre chapitres principaux qui sont :

Chapitre 1 : Ce premier chapitre dresse un état de l'art des systèmes de questions-réponses existants. Le chapitre se terminera par une étude bibliographique et une comparaison de ces systèmes selon plusieurs axes dans un tableau général.

Chapitre 2 : Ce chapitre est divisé en deux parties, dans la première nous introduisons la recherche d'information. Nous décrivons dans la deuxième partie l'aspect théorique du Deep Learning et du traitement du langage naturel ainsi que l'architecture du modèle utilisé BERT.

Chapitre 3 : Le troisième chapitre est consacré à notre contribution et la conception de la solution proposée. Nous décrivons les différentes opérations de prétraitements effectuées sur l'ensemble de données de Wikipédia, nous présentons aussi la structure de notre index. De plus, nous détaillons dans ce chapitre l'architecture et les paramètres utilisés pour notre classifieur des passages et de notre module d'extraction des réponses.

Chapitre 4 : Le dernier chapitre présente les résultats expérimentaux, leurs interprétations et enfin une discussion de ces derniers.

Enfin, le manuscrit se termine par nos conclusions sur le travail effectué. Tout travail de recherche introductif étant imparfait, cette section présente spécifiquement les améliorations possibles et offre donc des perspectives possibles de poursuite de ce travail.

Etat de l'art

1.1 Introduction

Le Question-Answering est un domaine de recherche qui a connu un intérêt remarquable durant ces dernières années, ce qui a permis une avancée majeure par les chercheurs de ce domaine. Ce chapitre résume l'étude bibliographique effectuée. Il porte sur la qualité des systèmes existants en général et les multiples dimensions qui permettent de les caractériser, les évaluer et de les classifier afin de connaître l'état de l'art de ce domaine. Nous avons aussi présenté les ensemble de données les plus utilisés pour leur développement.

1.2 Les systèmes de Questions-Réponses

Les systèmes de réponse aux questions est une forme sophistiquée de **Recherche d'Information** caractérisée par des besoins d'information qui sont au moins partiellement exprimés sous forme d'énoncés ou de questions en langage naturel, et constitue l'une des formes les plus naturelles d'interaction entre l'homme et l'ordinateur. Leur défi consiste à fournir une réponse concise à une question en langage naturel, étant donné la grande quantité de documents textuels. La réponse à une question est une tâche de **Recherche d'Information** limitée par l'expression de tout ou partie du besoin d'information sous la forme d'un ensemble de questions ou d'énoncés en langage naturel [Kolomiyets and Moens, 2011a]. L'utilisateur est intéressé par une réponse concise, compréhensible et correcte par exemple : "Who is the inventor of the telephone ?" la réponse sera : "Alexander Graham Bell".

1.3 Classification des systèmes

Dans le domaine des QAS, nous faisons souvent référence à leur classification selon leur domaine d'application, c-à-dire le domaine dans lequel ils opèrent et répondent aux questions. Comme le montre la taxonomie illustrées dans la Figure 1.1, il existe principalement deux classes qui sont : Closed-domain QAS et Open-domain QAS.

Dans notre étude bibliographique, nous avons constaté que les systèmes QA peuvent aussi être catégorisé selon les sources de connaissances sur lesquelles ils se basent pour extraire les informations afin de répondre aux questions. Nous avons également remarqué que les systèmes peuvent répondre à différents formes de questions, ce qui nous a incité à aussi les catégoriser par types de questions. Dans ce qui suit, nous allons présenter une vue globale puis détaillée des différentes catégories de classification des systèmes QA.

1.3.1 Classification par domaine d'application

1- Open-domain QAS

Les QAS du domaine ouvert (START, QuALIM, DeepQA, FALCON, Lasso, DrQA, YodaQA, AskMSR) ne sont pas limités à un domaine spécifique et fournissent une réponse courte à une question, traitée en langage naturel. En outre, les questions peuvent être sur quasiment n'importe quel sujet [ElKafrawy et al., 2018]. Ces systèmes recherchent généralement des réponses au sein d'une vaste collection de documents. Il existe un grand nombre de questions qui peuvent être posées par des utilisateurs occasionnels dans les systèmes questions-réponses du domaine ouvert, et afin de répondre à ces questions, ce type de systèmes exploite l'ontologie générale et la connaissance du monde dans leurs méthodologies pour générer des réponses. En général, la qualité des réponses fournies par ces systèmes n'est pas aussi précise que les systèmes questions-réponses du domaine fermé. Les QAS du domaine ouvert ne nécessitent pas de vocabulaire spécifique au domaine. Ils recherchent des réponses dans des grandes collection de documents [Reddy and Madhavi, 2017]. En contrepartie, ils permettent aux utilisateurs la possibilité de poser des questions sans connaître les mots clés du domaine spécifique pour formuler des questions. Ceci favorise l'utilisation des QAS du domaine ouvert par tous les utilisateurs des différents niveaux d'instruction et des différents domaines de spécialisation. De plus, Ces systèmes ne nécessitent pas le dictionnaire d'un domaine spécifique, ce qui veut dire que Wikipédia peut être utilisée comme source d'information.

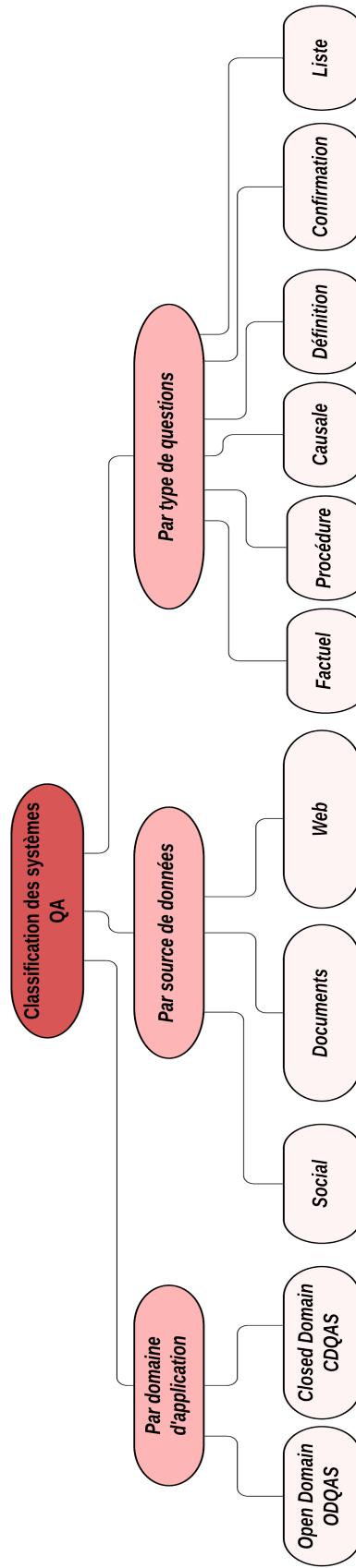


FIGURE 1.1: La taxonomie de l'état de l'art des QAS.

2- Closed domain QAS

Les systèmes QAS du domaine fermé (BASEBALL, LUNAR, MedQA, MYCIN, HONqa, EAGLi, askHERMES, KAAS, WEBCOOP) permettent de répondre aux questions relatives à un domaine particulier (médecine, cinématographie, aquariophilie, etc) en se basant sur les connaissances spécifiques aux domaines souvent formalisés dans des ontologies. Par ailleurs, des types limités de questions sont acceptés dans ces systèmes [ElKafrawy et al., 2018]. Ce domaine nécessite une disposition linguistique pour comprendre le texte en langue naturelle afin d'apporter une réponse précise aux requêtes [Ojokoh and Adebisi, 2019]. Malgré la précision que ces systèmes ont atteint puisqu'ils sont spécialisés dans des domaines précis, leur restriction les rend moins utiles au moment de vouloir avoir des réponses à des questions dans divers spécialités. Les QAS du domaine fermé peuvent être combinés pour créer des QAS de domaine ouvert afin de fournir des réponses dans plusieurs domaines avec une précision meilleure [Ojokoh and Adebisi, 2019].

1.3.2 Classification par source de données

Les QAS peuvent être classés selon leur source de données. Tous les QAS ont besoin d'une source de connaissances (Knowledge source) dans lequel ils fouinent à la recherche des passages. Pour répondre à ce besoin, quelques systèmes utilisent une base fermée, e.g., Wikipedia pour (START, QuALIM, Lasso, DrQA, BASEBALL, LUNAR, MedQA, MYCIN, HONqa, EAGLi, askHERMES, KAAS, WEBCOOP). D'autres systèmes utilisent le web (QuALIM, DeepQA, YodaQA, AskMSR). Tandis que d'autre répondent aux questions en mettant en relation les utilisateurs et interroger leur réseau social [Bouadjenek et al., 2016]. Cette méthode diffère des moteurs de recherche traditionnels afin de faire la collecte du contenu nécessaire pour répondre aux questions. Les systèmes se basant sur cette technique sont appelés "Social QAS" (Aardvark, Yahoo ! Answers, StackOverflow).

1.3.3 Classification par type de questions

Il existe plusieurs types de questions sur lesquelles les systèmes QA peuvent répondre. Nous citons 6 types dans cette Section : (i) Factuelles : Commencent généralement par un mot interrogé par Wh- (What, When, Where, Who) et requièrent comme réponse un fait exprimé dans le corps du texte (START, QuALIM, DeepQA, FALCON, Lasso, DrQA, YodaQA, AskMSR, BASEBALL, LUNAR, HONqa, askHERMES, KAAS, MYCIN). (ii) Causales : Nécessite une explication d'un événement ou d'un artefact, comme Pourquoi (DrQA, askHERMES, WEBCOOP). (iii) De confirmation : Requièrent un Oui ou un Non comme réponse à un événement exprimé dans la question (BASEBALL, askHERMES, WEBCOOP,

START, DrQA). (iv) De liste : Requièrent comme réponse une liste d'entités ou de faits (START, DrQA, LUNAR, askHERMES). (v) De définition : Nécessitent de trouver la définition du terme dans la question et commence normalement par "Qu'est-ce que" (LUNAR, MedQA, HONqa, EAGLi, askHERMES, WEBCOOP, START, DrQA). (vi) De procédure : Requièrent comme réponse une liste d'instructions pour l'accomplissement de la tâche mentionnée dans la question (DrQA, HONqa, askHERMES).

Nous illustrons dans le tableau 1.1 un récapitulatif de la classification des QAS selon les critères cités ci-dessus : Le domaine d'application du système qui peut être un domaine de type ouvert ou de type fermé, le type de questions prises en charge et enfin la source des données.

Système	Domaine d'application		Type de question		Source de données	
	Ouvert	Fermé	Factuelle	Causale	Confirmation	Liste
BASEBALL [Green et al., 1961]	X	X	X	X	X	X
LUNAR [Woods, 1973]		X	X	X	X	X
MedQA [Lee et al., 2006]		X		X	X	X
MYCIN [Shortliffe, 1977]		X	X		X	X
HONqa [Olvera-Lobo and Gutiérrez-Artacho, 2011]	X	X	X	X	X	X
EAGLi [Cao et al., 2011]		X		X	X	X
askHERMES [Cao et al., 2011]		X	X	X	X	X
KAAS [Diekema et al., 2004]		X	X	X	X	X
WEBCOOP [Benamara and Saint-Dizier, 2003]		X	X	X	X	X
START [Olvera-Lobo and Gutiérrez-Artacho, 2011]	X		X	X	X	X
QuALIM [Kaisser, 2005]	X		X		X	X
Aardvark ^a	X		X	X	X	X
DeepQA [Ferrucci et al., 2010]	X		X	X	X	X
FALCON [Harabagiu et al., 2001]	X		X			
Lasso [Katz et al., 2002]	X		X			X
DrQA [Chen et al., 2017]	X		X	X	X	X
YodaQA [Baudis, 2015]	X		X			X
AskMSR [Banko et al., 2002]	X		X			X
Yahoo! Answers ^b	X		X	X	X	X

TABLE 1.1: Classification des QAS selon plusieurs axes

^{a.} [https://en.wikipedia.org/wiki/Aardvark\(searchengine\)](https://en.wikipedia.org/wiki/Aardvark(searchengine))^{b.} <https://answers.yahoo.com/>

Nous remarquons dans le tableau 1.1 que les systèmes du domaine fermé se basant sur une source de données sociale permettent de fournir un moyen pour répondre à plusieurs types de questions (Aardvark, Yahoo! Answers). Nous remarquons aussi qu'aucun des systèmes de domaine fermé cités ne se base sur le web pour la recherche des réponses, et ceci pour que l'information soit correcte, exacte et venant d'une source de données fiable.

1.4 Les jeux de données disponibles

Nombreux sont les jeux de données destinés au développement des systèmes QA. Par conséquent, dans le tableau 1.2 nous présentons ceux fréquemment utilisés pour le développement et l'évaluation des QAS.

Nombreux sont les jeux de données destinés au développement des systèmes QA. Dans le tableau 1.2 nous présentons ceux fréquemment utilisés pour l'entraînement et l'évaluation des QAS.

Nom du Dataset	Source des questions	Source de connaissance	Taille du dataset
SQuAD [Rajpurkar et al., 2016]	Production participative ¹ (crowdsourced)	Wikipedia	100.000 questions avec réponses + 50.000 questions sans réponses
CNN/DailyMail [Chen et al., 2016]	Génération automatique des questions (Query logs)		879.000 questions
WikiQA [Yang et al., 2015]	Historique des requêtes des utilisateurs	Wikipedia	3047 questions
TREC-QA [Voorhees and Tice, 2000]	Historique des requêtes des utilisateurs (Query logs)		1479 questions
MCTest [Richardson et al., 2013]	Production participative		2640 questions

TABLE 1.2: Les différents jeux de données disponibles.

Les performances de la lecture automatique, en particulier, ont été considérablement améliorées ces dernières années avec l'introduction d'ensembles de données de compréhension de la lecture à grande échelle tels que CNN / DailyMail et SQuAD. Les systèmes utilisant ces ensembles de données se concentrent uniquement sur l'étape d'extraction des réponses, en supposant en fait que le passage pertinent du texte est déjà connu. WikiQA a été introduit comme un ensemble de données qui utilise les journaux de requêtes Bing comme source de questions. Il est utilisé principalement dans les systèmes qui répondent aux questions factuelles. Les réponses candidates ont été extraites à partir de la section de résumé des articles de Wikipédia. TREC est un ensemble de données pour la classification des questions consistant en des questions ouvertes, basées sur des faits et divisées en grandes catégories sémantiques. MCTest est un ensemble d'histoires et de questions associées librement disponibles destinées à la recherche sur la compréhension

automatique des textes. Cet ensemble de données exige des machines qu'elles répondent à des questions à choix multiples sur la compréhension de la lecture d'histoires fictives, s'attaquant ainsi directement à l'objectif de haut niveau de la compréhension machine en domaine ouvert.

Dans cette thèse nous allons concevoir et réaliser une architecture de QAS de domaine ouvert utilisant la base de connaissances fermée Wikipedia et entraîné sur l'ensemble de données SQuAD.

1.5 Conclusion

L'objectif de ce chapitre était de donner un aperçu global sur les systèmes réalisés jusqu'à présent dans le domaine du QA. De plus, nous avons fait une classification de ces systèmes selon différents axes : Domaine d'application, type de questions et source de données. Nous avons également présenté un comparatif entre les ensemble de données utilisés pour l'entraînement des QAS.

Chapitre 2

Généralités

2.1 Recherche d'information

La recherche d'information est un processus qui consiste à récupérer des informations stockées dans de grands ensembles de données pour répondre aux besoins d'information des utilisateurs. Baeza et ses collègues [Baeza-Yates and Ribeiro-Neto, 2011] ont défini la recherche d'information comme suit :

Définition : La Recherche d'Information (RI) est la science qui traite la représentation, le stockage, l'organisation et de l'accès aux éléments d'information afin de satisfaire les besoins des utilisateurs concernant ces informations.

Bien que la caractérisation des besoins de l'utilisateur ne soit pas une tâche simple, les utilisateurs précisent généralement leurs exigences sous la forme de requêtes que le système de RI doit traiter pour déterminer et présenter les documents qui correspondent à leurs besoins. Google, Bing et Yahoo ! sont certainement les systèmes RI les plus connus. Dans ces systèmes, les utilisateurs expriment leurs besoins sous forme de mots-clés, qui sont généralement considérés comme un résumé des besoins d'information de l'utilisateur. En réponse à une requête, le système de RI tente, en suivant un ensemble de processus, de récupérer des informations qui peuvent être pertinentes pour les utilisateurs.

Un système de RI est évalué en fonction de sa précision et de sa capacité à récupérer des informations et des documents de haute qualité, qui maximisent la satisfaction des utilisateurs, c'est-à-dire que plus le les réponses correspondent aux attentes des utilisateurs, plus le système est performant.

D'un point de vue architectural, le processus de RI se compose principalement des deux sous-processus

complémentaires suivants :

- Un processus hors ligne illustré dans la partie droite de la Figure 2.1. La collection de documents est explorée et parcourue afin de retrouver tous les documents grâce aux liens potentiels qui relient ces documents entre eux. Pour chaque document récupéré, un traitement est appliqué consistant principalement à réduire son ensemble de mots à un ensemble de termes d'index.
- Un processus en ligne illustré dans la partie gauche de la Figure 2.1 qui prend en charge la requête de l'utilisateur. La requête est envoyée généralement sous forme de mots clés et est réduite par le moteur de traitement des requêtes suivant la même stratégie que celle du traitement et l'indexation des documents. L'ensemble des termes de la requête utilisateur qui en résulte est souvent affiné par la suppression de certains termes [Kumaran and Carvalho, 2009]. Ensuite, la requête est traitée pour obtenir un ensemble de documents en utilisant la structure d'index précédemment construite. Cette liste est composée de documents qui sont liés aux termes de la requête. Après cela, les documents récupérés sont classés selon leur pertinence par rapport à la requête et par l'utilisateur, du plus pertinent au moins pertinent. Il s'agit de l'étape la plus critique car la qualité des résultats, telle que perçue par les utilisateurs, dépend fondamentalement du classement. Enfin, les documents les mieux classés sont ensuite formatés pour être présentés à l'utilisateur.

2.1.1 Les modèles RI

La modélisation en RI consiste à définir un modèle conceptuel pour la représentation des documents et des requêtes. De nombreux modèles de RI ont été proposés parmi lesquels : le modèle booléen, le modèle vectoriel spatial (VSM) et le modèle BM25. Ces modèles de RI sont bien décrits par la suite. Dans cette thèse, nous nous appuyons principalement sur le modèle BM25 pour son large usage et ses hautes performances [Baeza-Yates and Ribeiro-Neto, 2011].

- Modèle vectoriel VSM (Pondération TF-IDF) : Nous avons choisi d'utiliser la mesure $TF - IDF$ pour calculer ce poids et la similarité en cosinus pour calculer la similarité entre ces vecteurs. $TF - IDF$ est égale à la multiplication des deux mesures $TF_{t,d} \cdot IDF_t$ tel que $TF_{t,d}$ ou la Fréquence du Terme représente le nombre d'occurrences d'un terme t dans le document d . Tandis que IDF_t ou la Fréquence Inverse de Document mesure l'importance du terme t dans l'ensemble des documents D .

$$TF_{t,d} \cdot IDF_t \Rightarrow \begin{cases} TF_{t,d} = \log(1 + f_{t,d}) \\ IDF_t = \log\left(\frac{\|D\|}{\|D_t\|}\right) \end{cases}$$

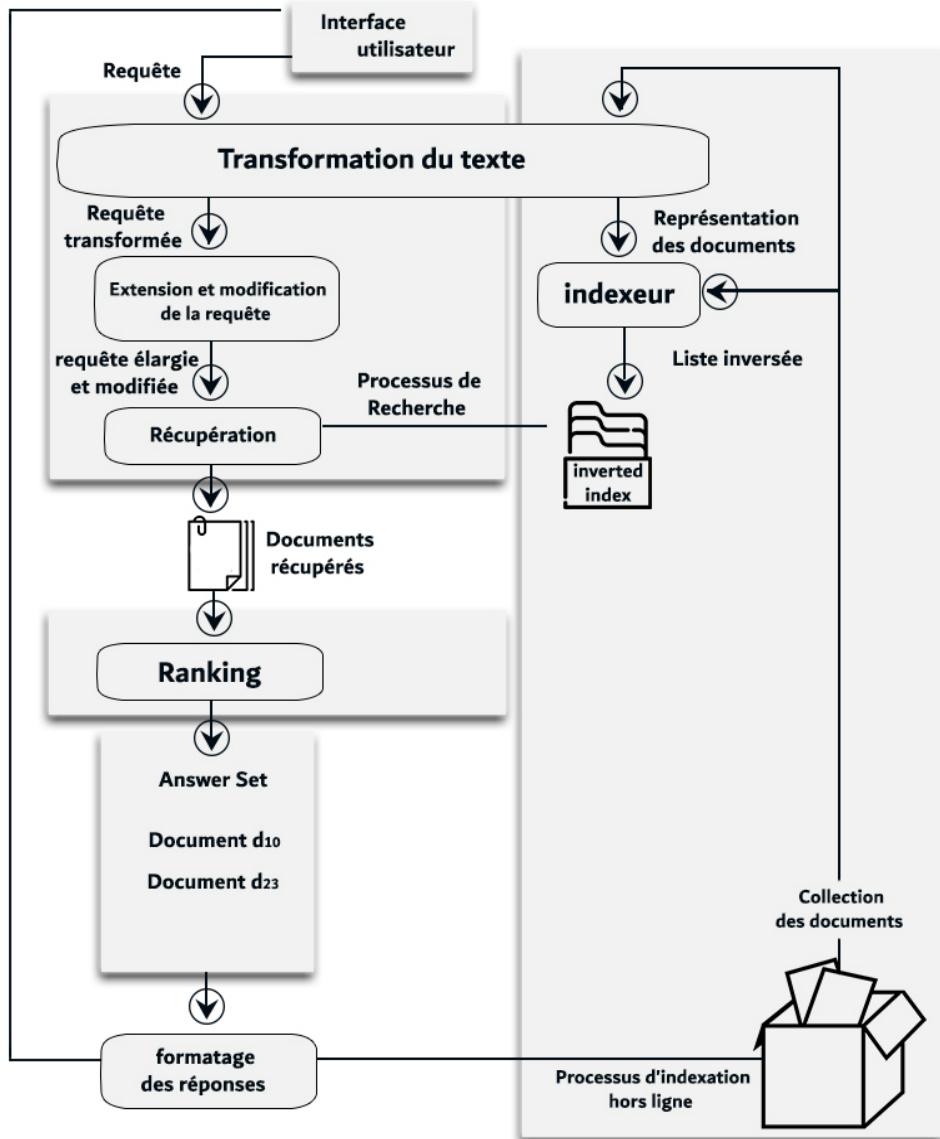


FIGURE 2.1: Processus de recherche d'information [Baeza-Yates and Ribeiro-Neto, 2011].

Où :

- t : Le terme t .
- d : Le document d .
- $f_{t,d}$: Le nombre d'occurrences du terme t dans le document d .
- $\|D\|$: Le nombre total de documents.
- $\|D_t\|$: Le nombre de documents contenant le terme t .

$$\text{sim}(\vec{q}, \vec{d}_j) = \frac{\vec{q} \cdot \vec{d}_j}{\|\vec{q}\| \cdot \|\vec{d}_j\|} \quad (2.1)$$

Où :

- q : La requête q .
- d_j : Le document j .

En notant que la similarité cosinus est particulièrement utilisée dans l'espace positif, où le résultat est clairement délimité dans $[0, 1]$. Ainsi, si deux vecteurs ont la même orientation et sont égaux, nous avons une similarité en cosinus de 1, mais si les deux vecteurs sont diamétralement opposés, nous avons une similarité de 0 [Baeza-Yates and Ribeiro-Neto, 2011].

- **BM-25** : BM25 est une fonction de recherche de mots qui peut classer un groupe de documents en fonction des termes de recherche qui apparaissent dans chaque document, quelle que soit leur proximité avec le document [Robertson and Zaragoza, 2009]. Le score BM-25 est calculé comme suit :

$$\text{Score}_{BM-25}(d, Q) = \sum_{i=1}^{\|Q\|} \text{IDF}_{q_i} \frac{TF_{q_i, d} (k_1 + 1)}{TF_{q_i, d} + k_1 \left(1 - b + b \frac{\|d\|}{\text{avgdl}}\right)} \quad (2.2)$$

Où :

- Q : La requête Q .
- $\|Q\|$: La taille de la requête Q .
- q_i : Le mot $q_i \in Q$.
- d : Le document d .
- $\|d\|$: Le nombre total de mots du document d .
- avgdl : La longueur moyenne des documents dans la collection considérée.
- k_1 et b : Des paramètres libres pouvant être optimisés selon les cas d'usage (ils sont généralement fixés à $k_1 \in [1.2, 2.0]$ et $b = 0.75$).

2.1.2 Les métriques d'évaluation

Cette Section est en partie un résumé du chapitre 4 du livre Modern Information Retrieval [Baeza-Yates and Ribeiro-Neto, 2011]. Une définition correcte est donnée concernant l'évaluation des algorithmes et des systèmes de recherche d'information.

Définition : L'évaluation de la recherche est un processus qui consiste à associer systématiquement une mesure quantitative aux résultats produits par un système de IR en réponse à un ensemble de demandes de l'utilisateur. Cette mesure doit être directement associée à la pertinence des résultats pour l'utilisateur. Une approche commune pour calculer une telle mesure consiste à comparer les résultats produits par le système avec les résultats suggérés par les humains pour ce même ensemble de requêtes. Notez que l'évaluation de l'extraction signifie ici l'évaluation de la qualité des résultats, et non des performances du système en termes de vitesse de traitement des requêtes.

De nombreuses mesures différentes ont été proposées pour évaluer la qualité de l'extraction des systèmes et des algorithmes de IR, c'est-à-dire la qualité des résultats. Ces mesures nécessitent un ensemble de documents et de requêtes. Toutes les mesures courantes décrites ici reposent sur une notion de pertinence : chaque document est connu pour être pertinent ou non pertinent par rapport à une requête particulière. En pratique, les requêtes peuvent être mal posées et il peut y avoir différentes nuances de pertinence. Dans ce qui suit, nous définissons six métriques d'évaluation qui seront utilisées tout au long de cette thèse.

- **Matrice de confusion :** Il s'agit d'une matrice décrivant les performances globales du modèle. Supposons que nous ayons un problème de classification binaire. Nous avons quelques échantillons qui se répartissent en deux catégories : oui ou non.

	Predicted Negative	Predicted Positive
Actual Negative	True Negative	False Positive
Actual Positive	False Negative	True Positive

TABLE 2.1: Matrice de confusion.

La matrice de confusion permet d'extraire et de lire quatre informations importantes qui sont :

- **TP** : Nombre d'échantillons correctement prédit appartenant à la catégorie “Positive”.
- **FP** : Nombre d'échantillons dans la catégorie ”Positive” qui n'ont pas été correctement prédits.
- **TN** : Nombre d'échantillons de la catégorie “Négative” correctement prédits.
- **FN** : Nombre d'échantillons de la catégorie “Négative” qui n'ont pas été correctement prédits.

- Précision et Rappel : La précision est la proportion d'instances pertinentes dans les instances récupérées, le rappel est la proportion du nombre total d'instances pertinentes qui sont réellement récupérées. Par conséquent, la précision et le rappel reposent sur la compréhension et la mesure de la pertinence [Ting, 2010]. En d'autres termes, la précision représente le pourcentage de documents prédit correctement par rapport au nombre de documents erronés retournés, le rappel quant à lui, donne le pourcentage des documents corrects qui sont donnés sans se préoccuper du nombre de documents erronés retournés.

$$\text{Précision} = \frac{TP}{TP + FP} \quad (2.3)$$

$$\text{Rappel} = \frac{TP}{TP + FN} \quad (2.4)$$

- R-précision : La R-précision représente le nombre de documents qui pertinents pour une requête q_i donnée [Craswell, 2009b]. En d'autres termes, s'il y a R documents pertinents parmi les documents les plus recherchés, alors la R-précision pour q_i examine les r premiers documents renvoyés, compte le nombre de documents pertinents et transforme ce nombre en fraction de pertinence :

$$R - precision = \frac{r}{R} \quad (2.5)$$

- Mean Average Precision : MAP (Mean Average Precision) est une mesure populaire utilisée pour calculer la performance des modèles de recherche d'information. L'idée principale de cette métrique est de générer un résumé du classement en une seule valeur en faisant la moyenne des valeurs de précision obtenus après l'observation de chaque nouveau document pertinent [Beitzel et al., 2009]. La précision moyenne AP_i pour la requête q_i est définie comme suit :

$$AP_i = \frac{1}{R_i} \sum_{k=1}^n (P(k) \cdot rel(k)) \quad (2.6)$$

Où :

- R_i : Le nombre total de documents pertinents pour la requête q_i .
- n : Nombre de documents retrouvés.
- $P(k)$: La précision du document k .
- $rel(k)$: Fonction d'indication de pertinence du document k ; 1 si le document k est pertinent, 0 sinon.

la MAP sur un ensemble de requêtes est alors définie comme suit :

$$MAP = \frac{1}{\|Q\|} \sum_{i=1}^{\|Q\|} AP_i(q_i) \quad (2.7)$$

Où :

- $\|Q\|$: Le nombre total de requêtes.
- q_i : La requête i de l'ensemble de requêtes Q .

- Mean Reciprocal Rank : Le Rang moyen de réciprocité (MRR ou Mean Reciprocal Rank en anglais) est une mesure permettant d'évaluer les systèmes qui renvoient une liste classée de réponses aux requêtes [Craswell, 2009a]. Cette métrique est calculée sur la base de la formule suivante :

$$MRR = \frac{1}{\|Q\|} \sum_{i=1}^{\|Q\|} \frac{1}{\text{rank}_i} \quad (2.8)$$

Où :

- rank_i : La position du premier document pertinent pour la requête i .
- $\|Q\|$: Le nombre total de requêtes.

Afin de calculer les métriques décrites précédemment, plusieurs outils sont disponibles. Parmi eux, nous citons *TREC Eval*¹ et que nous avons utilisé tout au long de cette thèse.

TREC Eval est un programme conçu comme une série d'ateliers dans le domaine de la recherche d'information. Il a débuté en 1992 dans le cadre du projet TIPSTER. Son but est d'encourager les travaux dans le domaine de la recherche d'information en fournissant l'infrastructure nécessaire à une évaluation objective à grande échelle des méthodologies de recherche textuelle [Teufel, 2007]. *TREC Eval* fournit une évaluation complète et exhaustive d'un moteur de recherche en offrant des dizaines de métriques qui permettent de juger un système RI.

2.1.3 Outil de développement

Pour élaborer un système de recherche d'information, plusieurs bibliothèques sont disponibles dans plusieurs langages de développement (Apache Solr, Apache Lucene, Sphinx, Xapian, Whoosh, etc). Dans ce qui suit, nous allons présenter l'un des outils de développement les plus utilisés dans le monde de la recherche d'information qui est *Apache Lucene*.

1. **TREC Eval** : https://github.com/usnistgov/trec_eval

*Apache Lucene*² est une bibliothèque qui sert à développer des moteurs de recherche textuelle très performants et complets, entièrement écrite en Java. Elle est capable d'effectuer des recherches plein texte dans des documents, c'est donc une technologie qui convient à toute application nécessitant cette fonctionnalité, surtout si elle est multi-plateforme. Apache fournit aussi une interface python de Lucene nommée *PyLucene* que nous allons utiliser durant cet œuvre.

2.2 Deep Learning en Traitement du Langage Naturel

L'apprentissage profond (DL ou Deep Learning en Anglais) est un sous-domaine de l'apprentissage machine qui concerne les algorithmes inspirés par la structure et le fonctionnement du cerveau appelés réseaux neuronaux artificiels.

Le traitement du langage naturel (NLP ou Natural language processing en anglais) est le processus de compréhension automatique du langage humain. Dans l'intelligence artificielle, le NLP permet aux machines et aux applications de comprendre le sens du langage humain, puis de générer des réponses appropriées pour former un flux de conversation naturel [Jain et al., 2018].

Il existe plusieurs techniques de traitement de langage naturel telles que les réseaux de neurones récurrents (RNN). Un RNN est un modèle d'apprentissage approfondi très populaire qui est utilisé pour effectuer un certain nombre de tâches de DL comme le traitement de langage naturel, le traitement d'images, etc [Sak et al., 2014]. Les RNN traitent les entrées d'une expression en langage naturel de manière séquentiel, c-à-dire les informations des jetons passent par tout le chemin pour se retrouver en fin de séquence. Mais d'un point de vue pratique, ce mécanisme reste imparfait et inefficient, dû au risque de «Gradient vanishing and exploding problems» ou perte d'informations lors de la mise à jour des poids du réseau, ce qui empêchera le poids de modifier sa valeur voire empêcher le réseau de poursuivre son entraînement et de perdre l'information pertinente [Hochreiter, 1998].

À ce stade, le mécanisme d'attention est apparu pour permettre d'examiner la phrase en tenant compte de tous les états précédents. Ces derniers sont ensuite pondérés en fonction d'une mesure apprise de la pertinence du jeton actuel, fournissant ainsi des informations plus précises sur les jetons pertinents lointains.

2. **Apache Lucene** : <https://lucene.apache.org/>

2.2.1 Le mécanisme d'attention

Dans le domaine de traitement du langage naturel, les éléments qui composent le texte source se caractérisent par le fait qu'ils ont chacun une pertinence différente par rapport à la tâche à accomplir. Par exemple, dans l'analyse des sentiments basée sur les aspects, les mots clés tels que "bon" ou "mauvais" peuvent être pertinents pour certains aspects à l'étude, mais pas pour d'autres. Dans la traduction automatique, certains mots du texte source pourraient ne pas être pertinents pour la traduction du mot suivant [Vaswani et al., 2017]. Par exemple, la traduction anglais-français, le premier mot de la sortie française dépend probablement beaucoup du début de l'entrée anglaise. Cependant, afin de produire le premier mot de la sortie française, le modèle ne reçoit que le vecteur d'état du dernier mot anglais. Théoriquement, ce vecteur peut coder des informations sur l'ensemble de la phrase à traduire, mais en pratique ces informations ne sont souvent pas bien préservées. Pour cela, il est important de prendre en compte la notion de pertinence, de manière à concentrer les ressources de calcul sur un ensemble restreint d'éléments importants. Le mécanisme d'attention est une approche de plus en plus populaire qui consiste à apprendre par machine la pertinence des éléments d'entrée. De cette façon, les architectures neurales pourraient automatiquement évaluer la pertinence de n'importe quelle région de l'entrée, et considérer ce poids lors de l'exécution de la tâche principale [Bahdanau et al., 2015]. Lorsque ce mécanisme est ajouté aux RNN, le modèle peut apprendre à tenir en compte l'état des premiers mots anglais lorsqu'il produit le début de la phrase française et donc des gains de performance importants [Vaswani et al., 2017].

L'introduction du transformateur a mis en lumière le fait que les mécanismes d'attention étaient puissants en eux-mêmes, et que le traitement séquentiel récurrent des données n'était pas nécessaire pour obtenir les gains de performance des RNN avec attention.

2.2.2 Les Transformateurs

Le Transformateur est un modèle de DL utilisé principalement dans le domaine du NLP. Comme les RNNs, les transformateurs sont conçus pour traiter des données séquentielles, comme le langage naturel, pour des tâches telles que la traduction et la classification de textes [Vaswani et al., 2017].

Ils utilisent un mécanisme d'attention sans être un RNN, en traitant tous les jetons en même temps et en calculant les poids d'attention entre eux. Le fait que les transformateurs ne reposent pas sur un traitement séquentiel et se prêtent très facilement à la parallélisation permet de les former plus efficacement sur des ensembles de données plus importants. Ils ont remplacé les anciens modèles de RNN tels que les LSTM

[Greff et al., 2016].

«Les transformateurs» est une architecture qui utilise l’attention pour augmenter la vitesse à laquelle ces modèles peuvent être formés. Le plus grand avantage, cependant, vient de la façon dont le transformateur se prête à la parallélisation. Ils reposent sur une architecture encoder-decoder (Figure 2.2). Le composant d’encodage et de décodage sont des piles du même nombre.

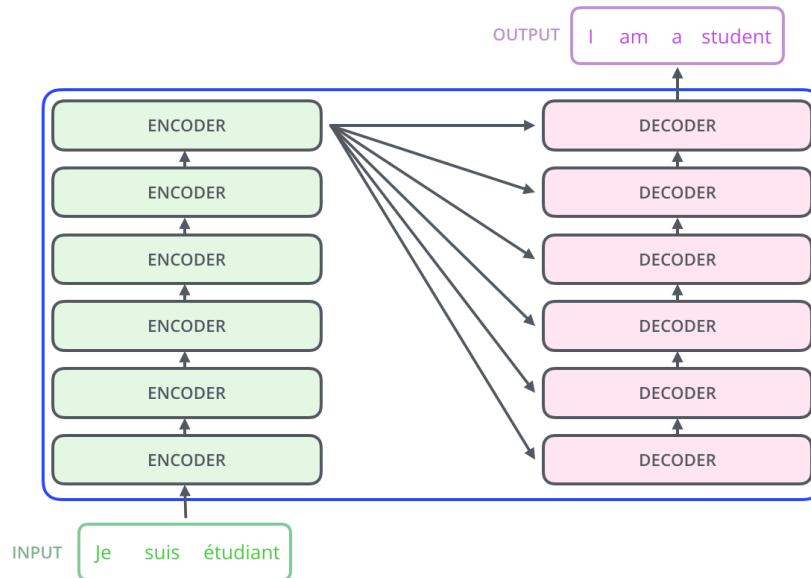


FIGURE 2.2: Architecture de base des transformateurs [Tra, 2018].

L'encodeur est composé de 6 couches identiques. Chaque couche a deux sous-couches. La première est celle de «Multi-head Self-Attention» [Voita et al., 2019] qui permet au modèle de s'occuper des informations provenant de différents positions. La seconde est un simple réseau «feed-forward» [Chen et al., 2001] entièrement connecté en fonction de la position.

Le décodeur à son tour, composé de 6 couches identiques, contient également deux sous-couches similaires à celles de l'encodeur, mais entre elles se trouve une troisième sous-couche qui réalise une «Multi-head attention» sur l'output de l'encodeur. Cette couche aide le décodeur à se concentrer sur les parties pertinentes de la phrase d'entrée.

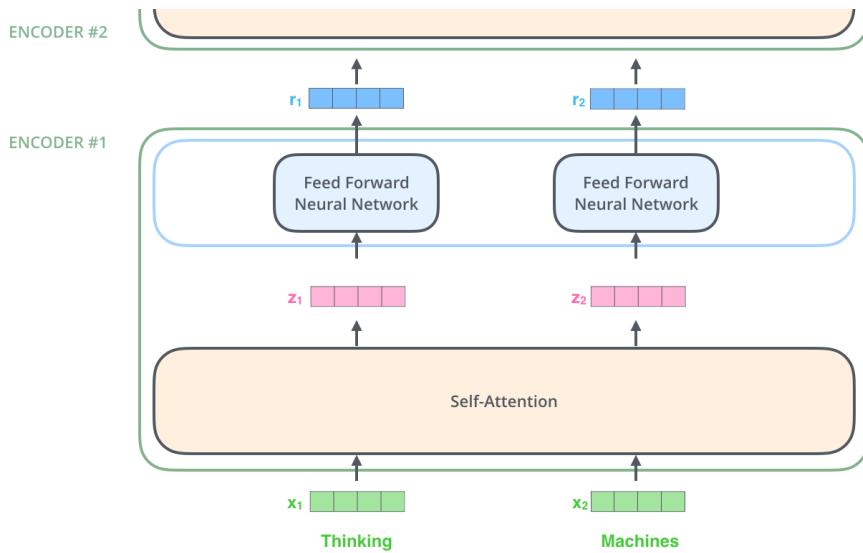


FIGURE 2.3: Architecture de l'encodeur du transformateur [Tra, 2018].

En phase d'encodage, comme le montre la Figure 2.3 :

1. L'entrée est l'encapsulation des mots pour la première couche. Pour les couches suivantes, ce sera la sortie de la couche précédente.
2. À l'intérieur de chaque couche, la Multi-head self-attention est calculée en utilisant les entrées de la couche comme vecteurs de clés, requêtes et valeurs ; puis le résultat est envoyé à la couche feed-forward.

Le principe de «Self Attention» repose sur l'idée que tous les mots seront comparés les uns aux autres afin d'avoir le sens exact de l'input comme le montre la Figure 2.4.

Et donc, contrairement aux RNN, les transformateurs n'exigent pas que les données séquentielles soient traitées dans l'ordre. Grâce à cette caractéristique, le Transformateur permet une parallélisation

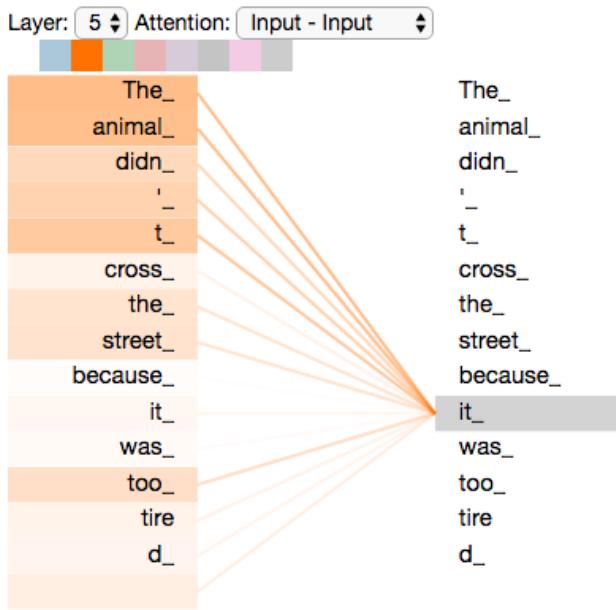


FIGURE 2.4: Principe de self-attention [Sel, 2020].

beaucoup plus importante que les RNN et donc des temps de formation réduits.

Les transformateurs sont devenus le modèle de choix pour résoudre de nombreux problèmes en NLP. Cela a conduit au développement de systèmes pré-entraînés tels que BERT (Bidirectional Encoder Representations from Transformers) [Devlin et al., 2018] et GPT (Generative Pre-trained Transformer) [Radford et al., 2018], qui ont été formés avec d'énormes ensembles de données en langage général et peuvent être adaptés à des tâches linguistiques spécifiques.

2.2.3 BERT (Bidirectional Encoder Representations from Transformers)

Les transformateurs est une architecture d'encodeur-décodeur, c-à-dire ils prennent une entrée et produisent une sortie. BERT [Devlin et al., 2018] est une architecture qui n'utilise que la partie d'encodeur des transformateurs pour réaliser de multiples tâches. Ce processus est nommé le **Transfer Learning** [Pan and Yang, 2010] qui est une méthode d'apprentissage automatique dans laquelle un modèle développé pour une tâche est réutilisé comme point de départ pour un modèle sur une deuxième tâche. Cette approche est utilisée en Deep Learning avec les modèles pre-trained comme BERT. Ils sont conçus pour réaliser une des tâches spécifiques (de traitement d'images, traitement de langage naturel...) avec changement de

l'input suivant la tâche à réaliser.

Il existe deux modèles de BERT, Base et Large :

- **BERT Base** (Cased et Uncased) : contient 12 couches, hidden size=768 et 12 self-attention heads.
- **BERT Large** (Cased et Uncased) : contient 24 couches, hidden size=1024 et 16 self-attention heads.

«Uncased» réalise une mise en minuscules avant la tokenisation (ex. John Smith devient john smith) et supprime également tout marqueur d'accent. «Cased» signifie que la casse et les accents sont conservés. En général, le modèle «Uncased» est préférable sauf si la casse est importante.

BERT est un modèle de compréhension linguistique à usage général sur un grand corpus de textes (comme Wikipédia) utilisé dans les tâches NLP. BERT surpassé les méthodes précédentes parce qu'il s'agit du premier système de préformation en NLP non supervisé et profondément bidirectionnel.

Comme le montre la Figure 2.5, la représentation d'entrée utilisée par BERT est capable de représenter une ou plusieurs phrases dans une seule séquence de jetons. Le token [CLS] désigne le début de la séquence. Chaque phrase est représentée sous forme de tokens. Les différentes phrases de la séquence sont séparées par le token [SEP].

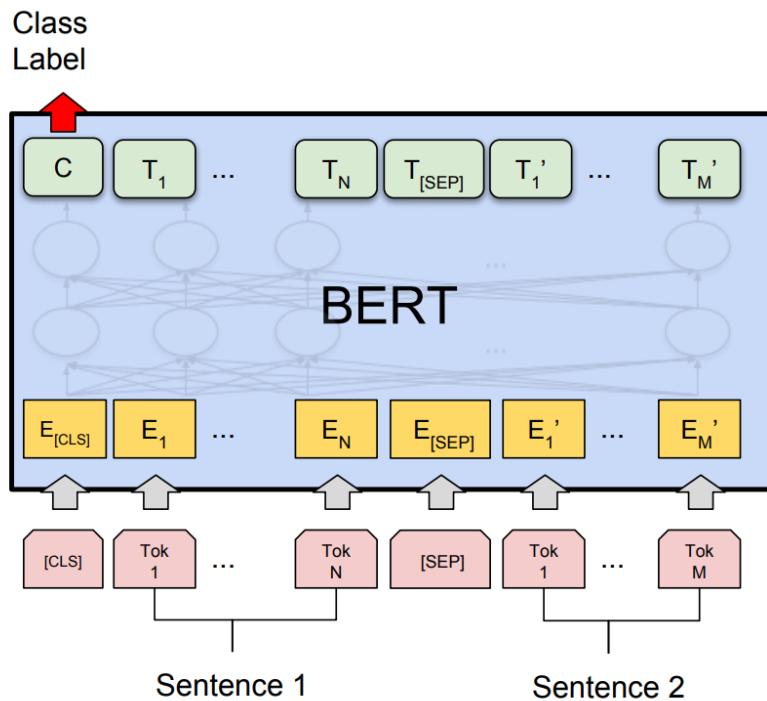


FIGURE 2.5: Représentation des entrées et sorties du modèle BERT [Devlin et al., 2018].

Le vocabulaire de BERT contient 30,522 tokens. Afin de traiter les mots inconnus, BERT utilise la décomposition en sous-mots.

2.2.4 Keras

Pour le développement de nos modèles de DL nous avons utilisé la librairie de python appelée Keras [Chollet et al., 2015]. C'est une API open-source de Deep Learning écrite, fonctionnant sur la plate-forme d'apprentissage automatique TensorFlow. Elle a été développée dans le but de permettre un apprentissage rapide.

2.2.5 Les métriques d'évaluation

Tout comme les systèmes de recherche d'information, les modèles réalisés en Deep Learning doivent être évalués afin de mesurer leur efficacité et leur performance ainsi que pour la sélection des bons hyperparamètres. Dans cette Section, nous présentons les principales métriques utilisées pour l'évaluation des modèles intelligents composants le système YouTaQA.

- Accuracy : Représente le nombre d'enregistrements correctement prédits parmi tous les points de l'ensemble de données N . Plus formellement, elle est définie comme suit :

$$\text{Accuracy} = \frac{TP + TN}{N} \quad (2.9)$$

- Loss : Désigne la moyenne des différences au carré entre les valeurs prédites et les valeurs réelles. Représente le taux d'erreur que le modèle a commis lors de la prédictions des résultats.

- F1 score : Le score F, également appelé *score F1*, est une mesure de la précision d'un test. Il est défini comme la moyenne harmonique pondérée de la précision et du rappel. Cette métrique est nécessaire pour trouver un équilibre entre la précision et le rappel.

$$F1 = 2 \cdot \frac{\text{Précision} \cdot \text{Rappel}}{\text{Précision} + \text{Rappel}} \quad (2.10)$$

- Exact Match EM : La métrique de correspondance exacte est une valeur binaire qui prend la valeur 1 si la réponse prédite et la réponse vraie sont exactement égales sans compter la ponctuation et les articles, zéro sinon. Cette métrique étant inconnue pour le grand public, elle est utilisée spécifiquement dans le domaine des QAS [Rajpurkar et al., 2016].

2.3 Conclusion

Dans ce chapitre, nous avons abordé la théorie de base du domaine de recherche d'information telle que le processus d'indexation et requêtage d'informations suivie des techniques d'évaluation des moteurs de recherche. De plus, nous avons procédé à une explication claire du traitement du langage naturel et du Deep Learning afin d'introduire le lecteur pour le prochain chapitre qui présentera la conception et l'implémentation du système YouTaQA.

Chapitre 3

Conception et implémentation de YouTaQA

3.1 Introduction

Dans ce présent chapitre, nous allons présenter la conception de la solution proposée. Nous décrivons les différentes opérations de prétraitement effectuées sur l'ensemble de données de Wikipédia, nous présentons aussi la structure de notre index. De plus, nous détaillons dans ce chapitre l'architecture et les paramètres utilisés pour notre classifieur des passages et de notre module d'extraction des réponses.

3.2 Architecture globale du système YouTaQA

Notre système est basé sur le Deep Learning et de la recherche d'information. Son but principal est de permettre aux utilisateurs d'avoir des réponses exactes à leurs questions uniquement en se basant sur un moteur de recherche qui dispenserait l'utilisateur de fournir des documents ou autre chose mis à part la question. Afin d'atteindre l'objectif de notre système, comme illustré dans la Figure 3.1, nous avons conçu une architecture composée de trois modules de base et une interface pour interagir avec l'utilisateur : (i) Un Moteur de Recherche d'Information (MRI) qui sert à fournir les 5 passages les plus pertinents à une question donnée.(ii) Un module de classification (MC) des passages basé sur le Deep Learning pour choisir et identifier parmi les 5 résultats du moteur de recherche le meilleur passage susceptible de contenir la bonne réponse à la question. (iii) Un module d'extraction des réponses (MER) basé sur le Deep Learning qui permet d'extraire la réponse exacte à partir du passage choisi par le classifieur dans l'étape précédente.

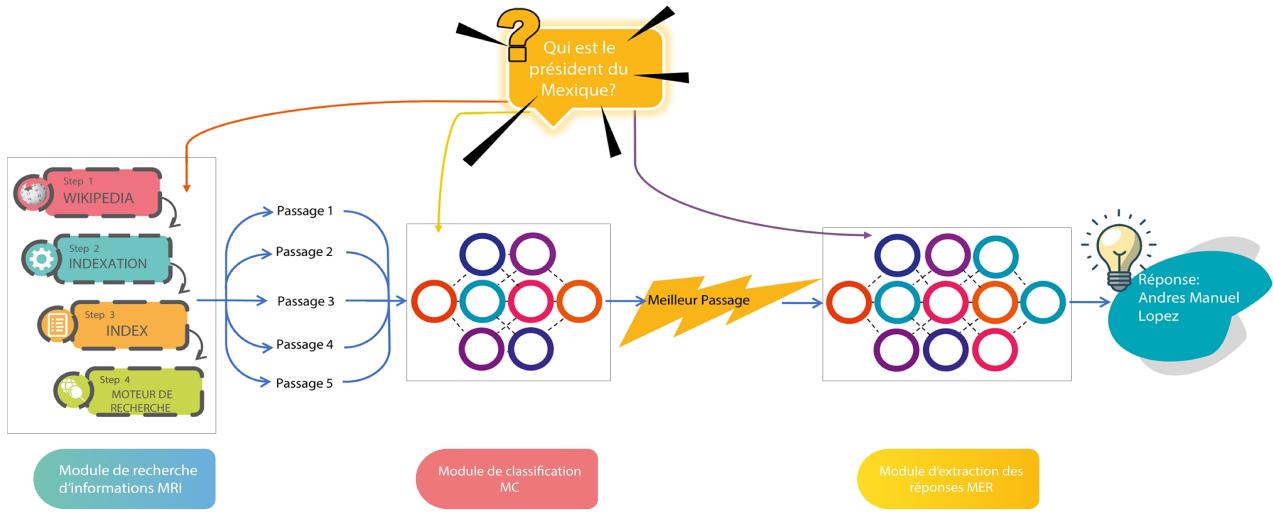


FIGURE 3.1: Schéma global du système YouTaQA

3.3 Le choix des jeux de données

Il existe plusieurs jeux de données utilisés pour l'apprentissage des QAS (Section 1.4). Durant notre projet, pour l'entraînement du système proposé, nous allons utiliser **SQuAD** (Stanford Question Answering Dataset).

3.3.1 SQuAD

SQuAD a été proposé par l'université de Stanford. Il contient un nombre impressionnant de questions (100.000 questions posées par des gens sur plus de 500 articles de différents domaines sur Wikipedia). Les passages dans SQuAD ont été extraits des articles de Wikipédia et couvrent un large éventail de sujets dans des domaines variés, allant des célébrités de la musique aux concepts abstraits. De plus, les questions sans réponses étaient le talon d'achille des jeux de données des systèmes questions-réponses, là encore, SQuAD fournit 50.000 questions sans réponses posées aléatoirement par la foule qui ont pour but de ressembler à des questions qui n'ont pas de réponses.

Un passage est un paragraphe d'un article d'une longueur variable. Chaque passage de SQuAD est accompagné de plusieurs questions. Ces questions sont basées sur le contenu du passage et qui peuvent avoir des réponses en lisant le passage. Enfin, pour chaque question, il existe une ou plusieurs réponses. Etant

donné que les réponses sont des segments des passages, cela permettra au système d'apprendre de manière optimale la façon dont il doit extraire les réponses de ces passages. De plus, dans SQuAD, les mots des questions sont souvent des synonymes de mots dans le passage, il s'agit d'une variation lexicale en raison de la synonymie.

Pour avoir une vision plus claire sur la structure de SQuAD, nous présentons ci-dessous un exemple d'une question extraite du jeu de données :

```
"question" : "When did Beyonce start becoming popular?",  
"id" : "56be85543aeaaa14008c9063",  
"answer" : "in the late 1990s",  
"answer_start" : 269,  
"is_impossible" : false,  
"context" : "Beyonce Giselle Knowles-Carter is an American singer, songwriter, record  
producer and actress. Born and raised in Houston, Texas, she performed in various singing  
and dancing competitions as a child, and rose to fame in the late 1990s".
```

L'exemple ci-dessus est un exemple d'une question sur SQuAD à partir d'un passage "**context**" d'un article de Wikipédia. Comme nous pouvons le voir, chaque question est identifiée par un "**id**". de plus, nous avons une variable booléenne "**is_impossible**" qui permet de préciser si le passage contient une réponse à la question ou non. Dans le cas où le passage contient une réponse à la question, la variable "**answer_start**" indique l'index du début de la réponse dans le champs "context".

3.3.2 Wikipedia

Puisque les questions proposées dans le jeu de données SQUAD sont basées sur des articles de Wikipedia, en vue d'établir un moteur de recherche, nous avons utilisé Wikipedia comme base de documents. Cette dernière offre l'intégralité de ses articles en plusieurs langues, parmi eux l'anglais. Wikipedia est disponible en ligne gratuitement en format XML¹ et comprend plus de 6.1 millions d'articles [Wik, 2020]. Comme nous pouvons le voir dans la Figure 3.2, notre base de documents a une structure XML qui permet de pourvoir des informations tel que le titre de l'article, la date de création, l'identifiant unique de l'article et le plus important qui est le contenu de l'article fractionné en sections.

1. <https://dumps.wikimedia.org/enwiki/latest/>

```
<mediawiki xmlns="http://www.mediawiki.org/xml/export-0.10/" xml:lang="en">
<siteinfo>
  <sitename>Wikipedia</sitename>
  <dbname>enwiki</dbname>
  <base>https://en.wikipedia.org/wiki/Main_Page</base>
  <generator>MediaWiki 1.29.0-wmf.12</generator>
  <case>first-letter</case>
  <namespaces>
    ...
  </namespaces>
</siteinfo>

<page>
  <title>Anarchism</title>
  <ns>0</ns>
  <id>12</id>
  <revision>
    <id>766348469</id>
    <parentid>766047928</parentid>
    <tstamp>2017-02-19T18:08:07Z</tstamp>
    <contributor>
      <username>GreenC bot</username>
      <id>27823944</id>
    </contributor>
  </revision>
  <content>
    .....
  </content>
</page>
</mediawiki>
```

FIGURE 3.2: Structure XML d'un article Wikipédia

3.4 Moteur de recherche MRI

Un moteur de recherche est un programme basé sur la recherche d'information et qui collecte et organise un ensemble de documents afin de faciliter la quête d'un ou plusieurs documents. De base, les utilisateurs saisissent une requête sur ce qu'elles aimeraient trouver et le moteur fournit le contenu qui correspond à ce qu'elles veulent. Pour notre moteur de recherche, nous avons suivie la même philosophie. Nous nous sommes basé sur la collection exhaustive d'articles fournie par Wikipédia.

3.4.1 Pré-traitement de la base Wikipédia

Fractionnement des articles

Après avoir choisi la base de wikipédia, et afin de simplifier la manipulation de la grande quantité des données de notre base de données, nous avons procédé à une répartition des articles de wikipédia sur trois niveaux d'arborescence de dossiers. Pour cela, chaque article sera répertorié suivant son **id** unique (eg. L'article avec l'id = 00020201 sera placé dans le répertoire 00/02/02/ sous le nom 00020201.xml) comme le montre la Figure 3.3.

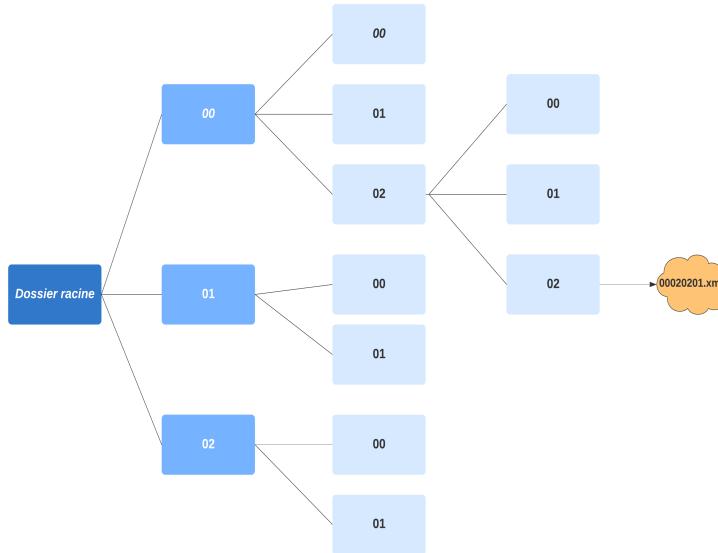


FIGURE 3.3: Arborescence des fichiers XML.

Interprétation de la syntaxe de Wikipédia

Dans ses articles, Wikipédia utilise souvent une syntaxe spéciale nommée “WikiText”² qui sert à maquiller ses articles (par exemple, appliquer du gras sur le mot “Bonjour” revient à écrire “’’’BonJour’’’” dans ses articles en format brut), ce qui nous a causé une certaine difficulté. Afin de remédier à ce problème, nous avons appliqué un formatage du texte en utilisant le script WikiExtractor³ qui nous a permis de se débarrasser de la mise en page de Wikipédia.

3.4.2 Indexation des articles

Après avoir traité les articles de Wikipédia, nous avons indexé ces articles en utilisant la bibliothèque *PyLucene*. Dans cette étape, nous avons procédé à l’implémentation d’un analyseur personnalisé pour notre index. Cet analyseur sert à appliquer les transformations du texte telles que la suppression des mots vides, la tokenisation, la normalisation et le stemming⁴ des mots de chaque article.

2. **WikiText** : Langage de balisage qui permet la mise en forme du contenu des articles de Wikipédia.

3. **WikiExtractor** : Un script Python qui permet de convertir la mise en page appliquée par wikipédia en un text pure sans syntaxe.

4. <https://en.wikipedia.org/wiki/Stemming>

Pour but de faciliter la tâche de notre future module d'extraction des réponses, et afin de booster les performances de notre système en termes de temps d'exécution, au lieu de considérer les articles en eux-mêmes comme des documents, nous avons eu l'idée de considérer les sections des articles comme documents afin de peaufiner la recherche suite à une requête donnée. Pour éclaircir tout ça, nous avons illustré dans la Figure 3.4 l'exemple d'une page Wikipédia et comment l'index va considérer cet article en divisant le même article en trois documents.

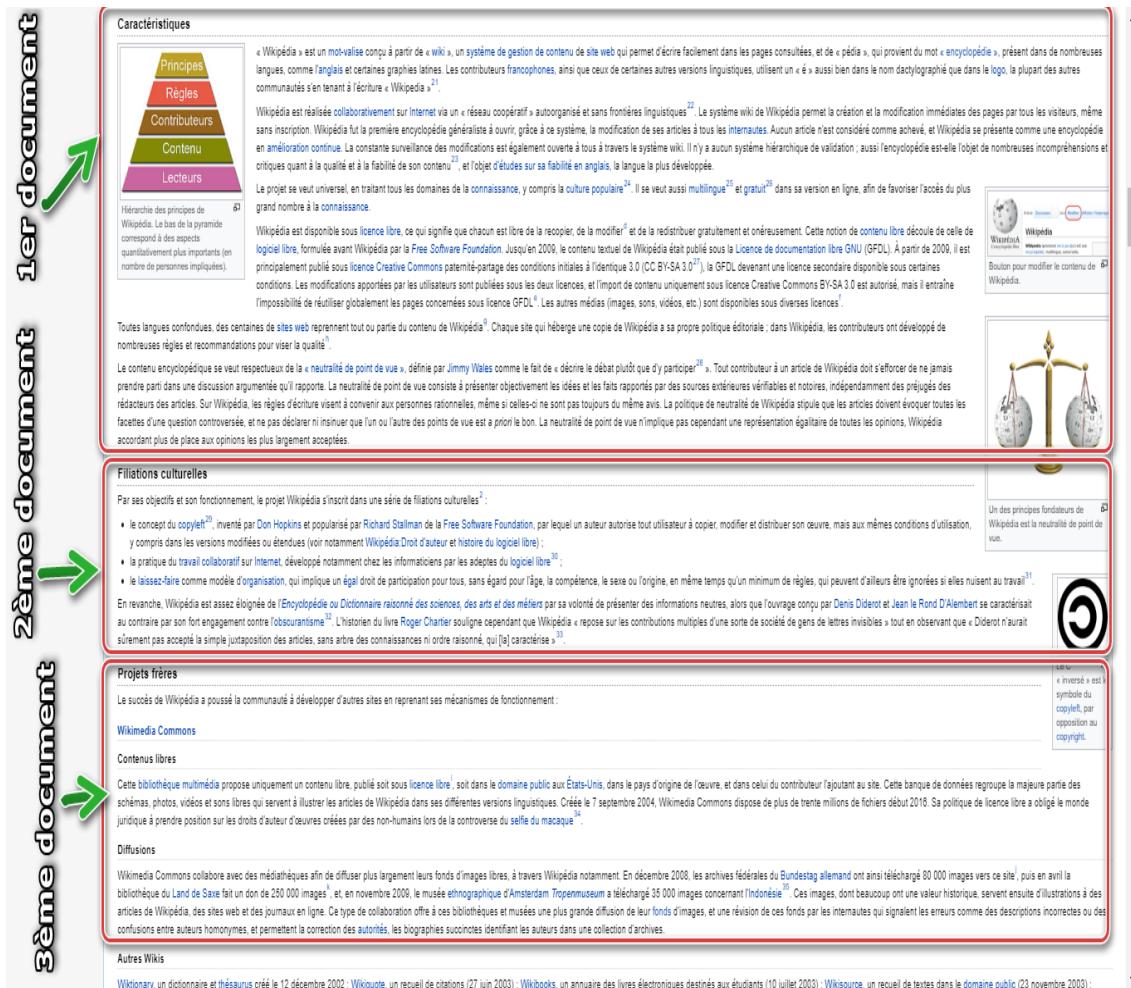


FIGURE 3.4: Schéma représentatif des sections d'un article Wikipédia

Notre index est constitué de 5 champs essentiels :

- L'identifiant de l'article qui est unique à chaque article (Champs de type LongPoint).
- Le titre de l'article (Champs de type StringField).
- L'identifiant de la section (Champs de type LongPoint).
- Le titre de la section (Champs de type StringField).

- Le contenu de la section, c'est le contenu principal de chaque document (Champs de type Text-Field).

3.4.3 Méthodes de recherche adoptées

Pour le processus de recherche, nous avons opté pour deux méthodes de recherche différentes applicables sur notre index.

1. **Méthode de recherche SimpleFieldSearch** : La première méthode de recherche, qui est la plus basique, sera en fait une méthode de recherche qui, après avoir extrait les mots clés d'une requête, permet de rechercher les mots clés dans le contenu des documents seulement.
2. **Méthode de recherche MultifieldsSearch** : Cette méthode consiste à faire une recherche sur à la fois le contenu et le titre.

3.5 Module de classification MC

Comme décrit dans la Section 2.2.3, BERT est un modèle pré-entraîné basé sur le "Transfer Learning" capable de réaliser plusieurs tâches NLP. Parmi ces tâches, nous retrouvons la classification de texte. Pour notre cas, nous avons utilisé **BERT Base Uncased** dans le but de réduire le temps d'exécution durant les expérimentations.

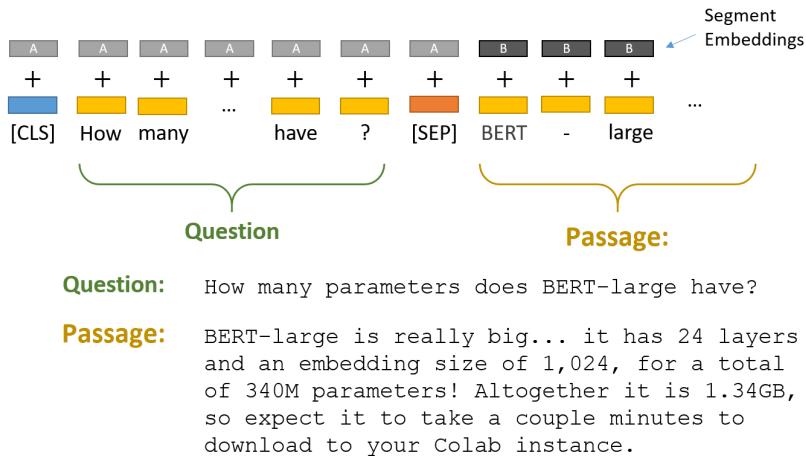


FIGURE 3.5: La phase de tokenisation des entrées.

Pour ajouter une tâche de classification de texte à BERT, nous intégrons à la fois la question et le passage dans la saisie. Comme première étape, nous commençons par la tokenisation de notre jeu de données afin

de le coder suivant le même format. La fonction de tokenisation parcourt les données et concatène chaque paire de **Question-Passage** comme illustré dans la Figure 3.5.

Chaque séquence (paire) commence par le token de classification spécial **[CLS]** en plus d'un autre token spécial **[SEP]** qui sépare les deux parties de l'entrée et permet ainsi à BERT de différentier la question et le passage. Afin que tous les tokens soient de la même taille, nous avons défini une taille maximale aux séquences d'entrée égale à 300, complété avec un remplissage (padding) avec le mot clé **[PAD]** dans le cas où la taille du passage concaténé à la question est inférieure à la taille de l'entrée fixée. BERT utilise “Segment Embeddings” afin de différentier la question du passage. Dans la Figure 3.5, A et B représentent les “Segment Embeddings” ajoutés aux tokens codés avant de les passer comme paramètres d'entrée.

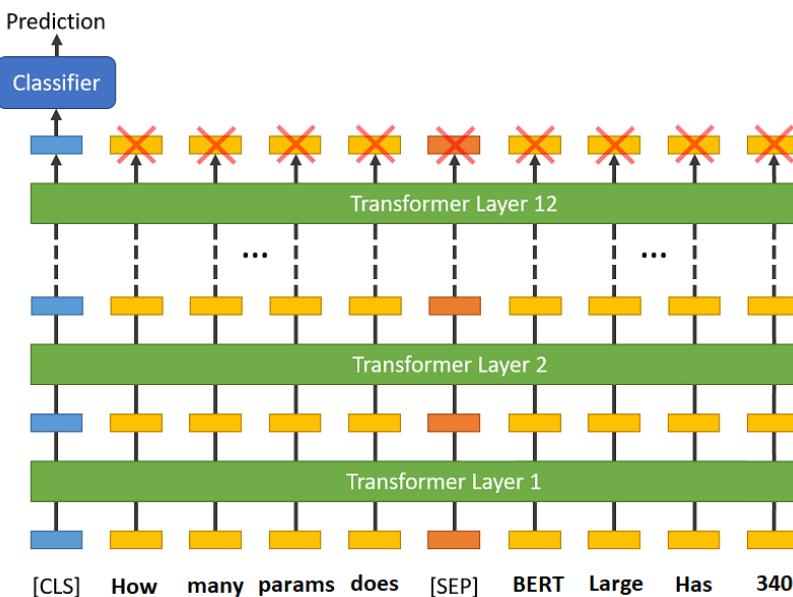


FIGURE 3.6: La classification du texte avec BERT.

Le premier token de chaque séquence d'entrée de BERT est toujours un token de classification **[CLS]** c-à-dire qu'il est utilisé pour les tâches de classification NLP de BERT [Devlin et al., 2018]. Le MC prend comme entrée la question et le passage, les passe sur les 12 couches de BERT Base et produit comme sortie une prédiction que le passage contient la réponse recherchée (Figure 3.6). La valeur de sortie est ensuite normalisée avec la fonction *Sigmoid*⁵ pour enfin avoir une classification binaire c-à-dire **1** si le passage contient la réponse et **0** sinon.

5. [https://fr.wikipedia.org/wiki/Sigmoïde_\(mathématiques\)](https://fr.wikipedia.org/wiki/Sigmoïde_(mathématiques))

3.6 Module d'extraction de réponses MER

En ce qui concerne le module d'extraction de la réponse, nous avons mis en oeuvre un modèle qui, étant donné une question et un passage reçu par le MC, BERT doit retourner la partie du texte correspondante à la bonne réponse. En d'autres termes, le modèle prédit le début et la fin de la réponse exacte à partir du passage donné qui est le plus susceptible de répondre à la question. Pour ce faire, notre modèle doit passer par l'étape d'entraînement en utilisant un ensemble de données destiné à ce cas de Figure. Dans ce cas, nous allons utiliser SQuAD.

Après avoir sélectionné le passage le plus pertinent parmi ceux retournés par le MRI, le module d'extraction le prend comme entrée, accompagné par la question posée. Comme c'était le cas pour le MC, le MER prend comme entrée la paire **Question - Passage candidat** tokenisée et produit en sortie la réponse jugée exacte à la question donnée et qui est extraite à partir du passage. Le MER proposé a été entraîné avec BERT Base Uncased.

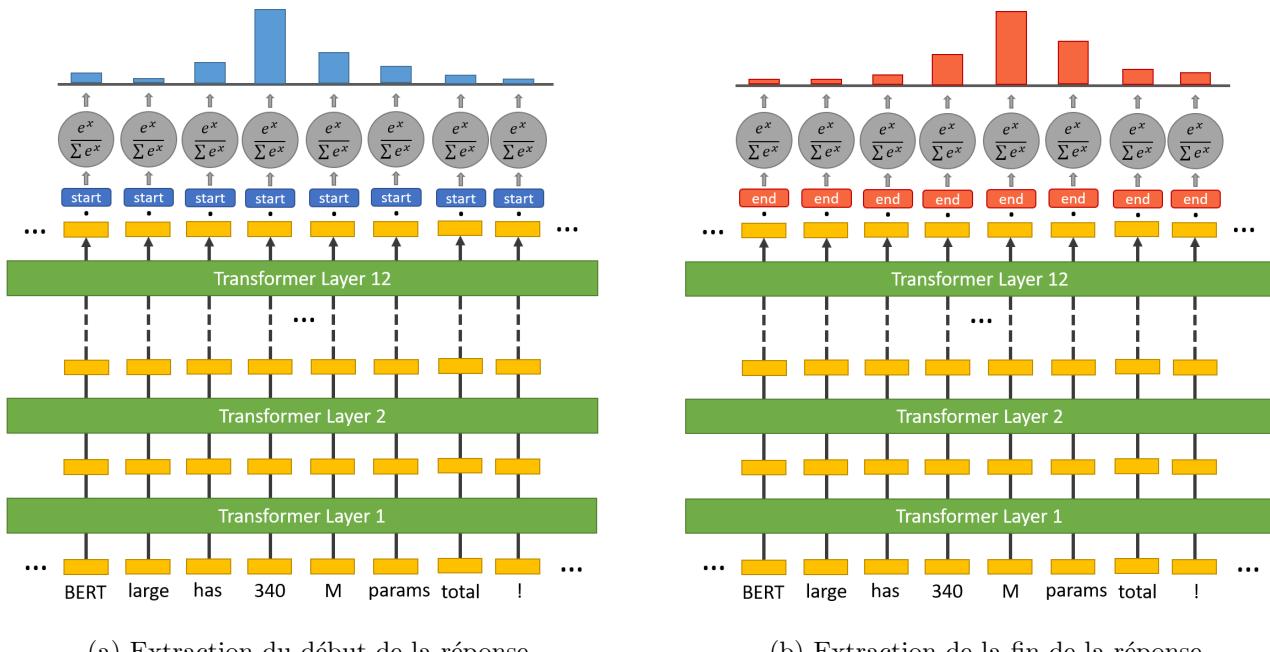


FIGURE 3.7: Extraction du début et fin de la réponse avec BERT.

Pour chaque token dans le passage choisi, nous l'introduisons dans le classificateur de token de début/fin. L'extracteur de début/fin passe l'entrée sur les 12 couches de BERT Base et produit un vecteur de sortie (les rectangles orange produits par la dernière couche dans la Figure 3.7) où chaque case représente la sortie d'un mot du texte. Ce vecteur est ensuite multiplié par une matrice de début/fin (représenté dans la Figure 3.7 par des rectangles bleus/rouges). La taille de cette matrice est $768*N$ tel que N représente la taille du passage d'entrée. Le produit est donc un vecteur de taille N sur lequel la fonction *Softmax*⁶ est appliquée pour avoir un nouveau vecteur de probabilités (dont la somme est égale à 1). Chaque $case_i$ du vecteur représente la probabilité que le mot_i tel que $i \in N$ représente le début/la fin de la réponse.

BERT englobe les deux modèles d'extraction de début et de fin qui s'entraînent séparément. Il comporte une dernière couche qui s'occupe de les fusionner pour délimiter à la fin la réponse à la question posée. Dans l'exemple illustré dans la Figure 3.7, le MER choisi le token **340** comme début de réponse et **M** comme fin de réponses suivant leurs probabilités.

3.7 Déploiement du système YouTaQA

Afin d'interagir avec les utilisateurs de notre système, nous avons mis en œuvre une application web SPA⁷ complète qui fournit une expérience utilisateur facile en appliquant les dernières normes UI⁸ et l'utilisation du Material Design⁹ fourni par Google comme nous pouvons le voir dans les captures d'écran de notre interface dans la Figure 3.8. Pour cela, nous avons utilisés pour le front-end un template Colorlib¹⁰ bâti en utilisant VueJs¹¹, Bootstrap¹² pour le front-end et le framework python Django¹³ pour le back-end.

6. https://fr.wikipedia.org/wiki/Fonction_softmax

7. **SPA (Single page application)** : c'est une application qui n'a pas besoin de recharger la page dans le navigateur pendant son utilisation.

8. **UI (user interface)** : signifie "interface utilisateur", c'est la présentation graphique d'une application.

9. **Material Design** : C'est un ensemble de règles de design proposées par Google et qui s'appliquent à l'interface graphique.

10. <https://colorlib.com/wp/templates/>

11. **VueJs** : c'est un framework pour construire des interfaces utilisateur à base de javascript.

12. **Bootstrap** : c'est un framework CSS pour construire des interfaces utilisateur.

13. **Django** : c'est un framework web gratuit écrit en Python qui sert à développer des applications web.

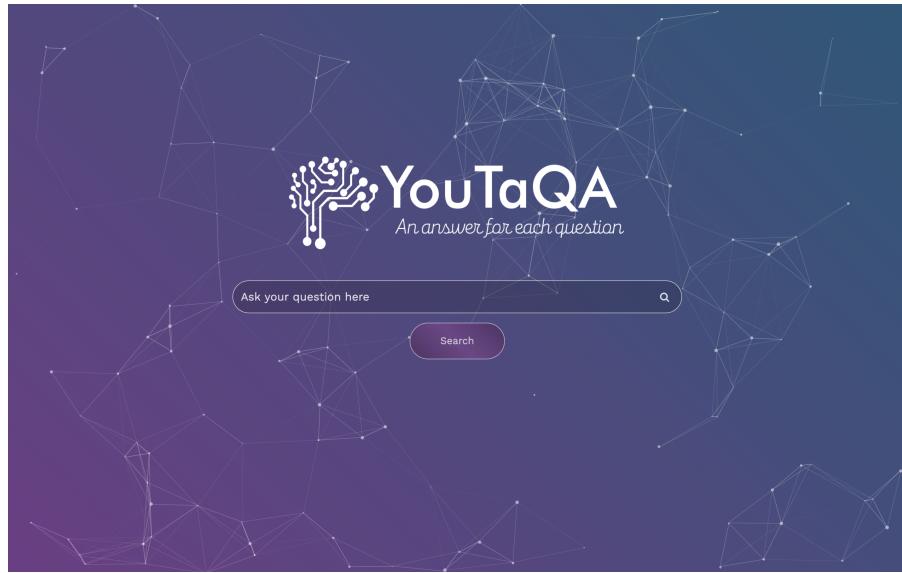


FIGURE 3.8: Capture d'écran de l'application web YouTaQA

3.8 Conclusion

Dans ce chapitre nous avons décrit l'architecture de notre système de la saisie de la question par l'utilisateur jusqu'à l'obtention des résultats en passant par plusieurs étapes qui sont : (i) La sélection des passages pertinents, (ii) La classification de ces derniers par probabilité et (iii) L'extraction de la réponse exacte à partie du meilleur passage choisi. Nous présenterons dans le chapitre suivant nos résultats ainsi que leurs discussions.

Analyse et discussion des résultats

4.1 Introduction

Durant ce dernier chapitre, nous présenterons les résultats de chaque module de notre système. La deuxième partie de ce chapitre consiste à présenter les résultats globaux de notre système YouTaQA ce qui nous permettra de positionner et de comparer ce dernier avec les travaux antérieurs et de mettre en évidence la valeur ajoutée que ce système apporte à l'état de l'art de ce domaine.

4.2 Prétraitement et fractionnement des données

Comme mentionné précédemment, le moteur de recherche et les modules de classification et d'extraction de réponses ont été entraîné et testé sur l'ensemble de données SQuAD. Les données de ce dernier ont été divisé en trois parties que nous pouvons résumer dans le tableau suivant :

	Positif	Negatif	Total
Train set	70664	34032	104 696
Validation set	16157	9466	25 623
Test set	5928	5945	11 873

TABLE 4.1: Le fractionnement du jeu de données SQuAD.

Le jeu de données "Train set" est utilisé pour l'entraînement des modèles de Deep Learning. Le "Validation set" est utilisé pour équilibrer le modèle et choisir les bons hyper-paramètres. Tans dis que le "Test set"

est utilisé pour évaluer les différents modules de système et voir l'amélioration des résultats obtenus.

4.3 Résultats du module de recherche d'information MRI

Après avoir établi un premier prototype de notre système, nous l'avons testé en utilisant l'évaluateur TREC-Eval en s'appuyant sur le dataset SQuAD. Nous avons utilisé les questions présentes dans ce dernier en les considérant comme requêtes pour notre moteur de recherche et comparer ses résultats avec les passages fournis par SQuAD.

4.3.1 Méthodes de recherche employées

Postérieurement, nous avons lancé l'évaluateur TREC-Eval avec les 4 méthodes de recherche afin de comparer les performances de chaque méthode et choisir à la fin la meilleure méthode à utiliser. Ces méthodes seront décrites dans ce qui suit.

VSM_SimpleSearch : Dans cette méthode de requêtage, nous avons utilisé le modèle de classement des résultats VSM (Voir la Section 2.1.1) tout en utilisant une recherche simple des mots clés de la requête dans le contenu des articles seulement.

VSM_MultiFieldsSearch : Cette méthode de recherche est basée sur la recherche multi-champs (Multi-Fields Search en anglais), c'est-à-dire la recherche des mots d'une requête est effectuée non seulement sur le contenu de l'article mais aussi sur le titre de ce dernier.

BM25_SimpleSearch : La présente méthode est basée sur une simple recherche dans le contenu des articles. Durant l'étape de classement des résultats, ici nous utilisons l'approche BM25 au lieu du VSM (Voir la Section 2.1.1).

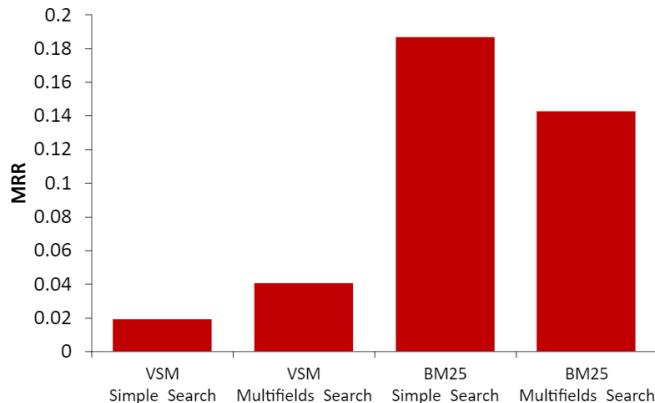
BM25_MultiFieldsSearch : Là, nous avons implémenté une méthode de recherche qui emploie la recherche multi-champs (recherche appliquée dans le contenu et le titre des articles au même temps). Là aussi, nous utilisons l'approche BM25.

4.3.2 Discussion des performances des méthodes de recherche

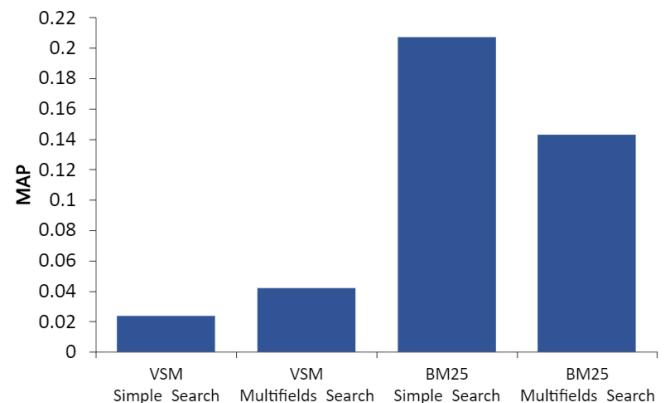
Après avoir lancé l'évaluation de toutes les méthodes, nous avons tracé des graphes et des histogrammes afin de comparer les résultats.

Histogrammes MAP, MRR et R-Précision

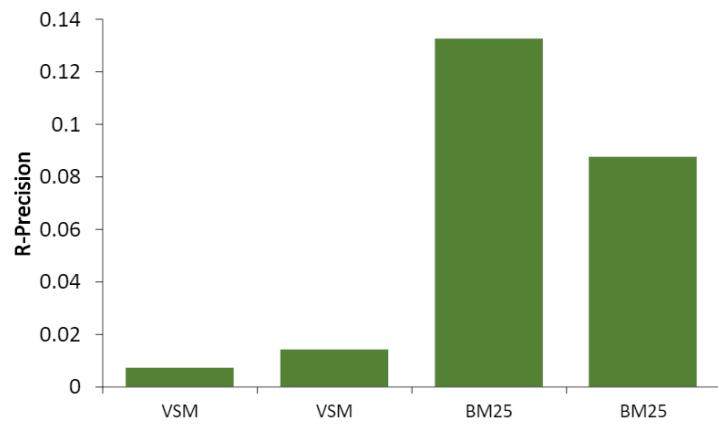
Les histogrammes dans la Figure 4.1 représentent le MRR, la MAP et le R.Précision de chaque méthode de recherche. Nous remarquons d'abord que toutes les méthodes qui utilisent la métrique BM25 comme métrique de classement des documents surpassent de loin les méthodes VSM en termes de précision moyenne. Cela dit, la métrique BM25 étant plus sophistiquée et étant plus précise, permet une recherche plus performante et plus assurée.



(a) Graphe du MRR des résultats.



(b) Graphe du MAP des résultats.



(c) Graphe du R-Précision des résultats.

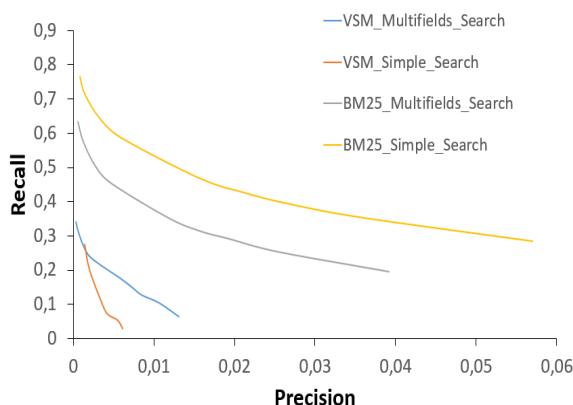
FIGURE 4.1: Histogrammes d'évaluation du Module de Recherche d'Information.

Maintenant, Après avoir comparé les deux métriques de classement, nous passons directement à la comparaison des méthodes qui utilisent BM25. Là encore, nous remarquons la supériorité de la méthode de recherche *BM25_SimpleSearch*.

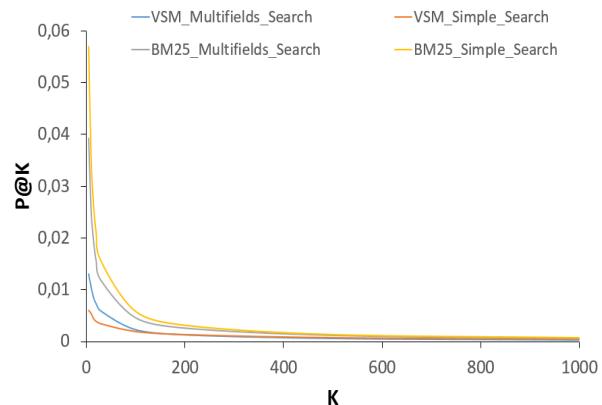
Une chose de plus à remarquer dans l'histogramme du MRR dans la Figure 4.1, la méthode de recherche *BM25_SimpleSearch* a un score MRR et MAP égal à 0.20. Ceci dit, cette méthode permet d'avoir en moyenne le bon document parmi les 5 premiers documents retournés, ce qui, d'après notre choix, fournit en général toujours le bon document que nous cherchons parmi les 5 premiers documents envoyés au classifieur.

Graphes Précision, Rappel et précision-rappel

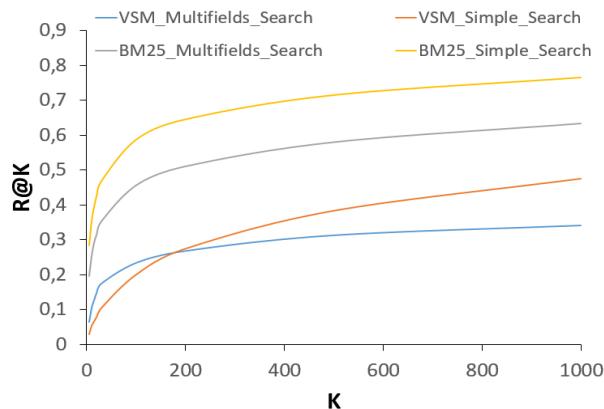
Ces autres métriques viennent confirmer ce que nous avons constatés précédemment.



(a) Graphe du Précision-Rappel des résultats.



(b) Graphe du Précision@K des résultats.



(c) Graphe du Rappel@K des résultats.

FIGURE 4.2: Graphs d'évaluation du Module de Recherche d'Information.

Pour les graphes de la précision@K et rappel@K présentés dans la Figure 4.2, où K représente le nombre de résultats retournés suite à une recherche, permettent d'affirmer les performances supérieures réalisées par la méthode *BM25_SimpleSearch*.

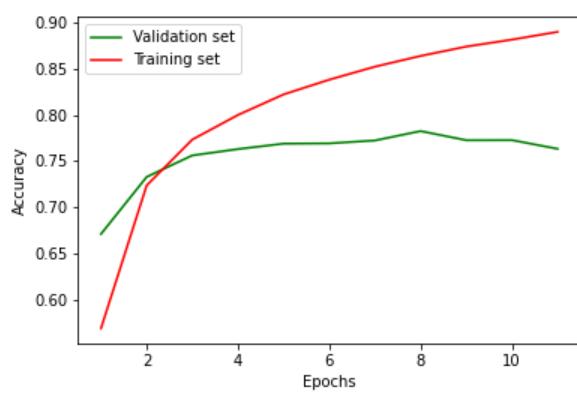
Par ailleurs, nous constatons une baisse considérable de la précision en augmentant le nombre K de résultats retournés. Quant au rappel, il augmente en augmentant le nombre K de résultats retournés. Cela est justifié par la façon dont la précision et le rappel sont calculé. D'après la définition des deux métriques, la précision représente le pourcentage de documents prédit correctement par rapport au nombre de documents erronés retournés. Le rappel lui, implique le pourcentage des documents corrects qui sont donnés sans se préoccuper du nombre de documents erronés retournés. Afin de remédier à tout ça, nous avons établis un nouveau graphe présent dans la Figure 4.2 qui mets en valeur la relation précision-rappel à chaque valeur de K. Encore une fois, la méthode *BM25_SimpleSearch* Figure comme étant la meilleure méthode de recherche en obtenant un rapport précision-rappel toujours au-dessus des autres méthodes.

À la fin, après avoir comparé les différentes méthodes de recherche présentées antérieurement, nous sortons avec une conclusion qui permet de passer à l'étape suivante qui est le classifieur des documents tout en lui fournissant 5 documents. Le choix du nombre de documents passés au classifieur vient suite aux performances affichées par la méthode de recherche *BM25_SimpleSearch*, qui en moyenne, permet d'avoir le bon document parmi les 5 premiers documents retournés par notre moteur de recherche (MRR = 0.20).

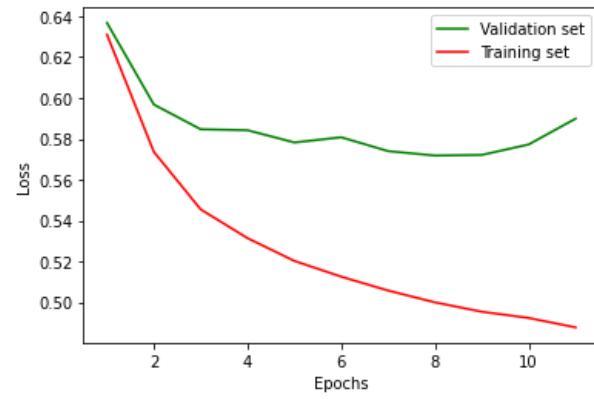
4.4 Résultats du module de classification MC

Après la mise en place et obtention des passages pertinents du moteur de recherche , il est nécessaire à présent, de tester le MC et suivre l'évolution du MRR à l'aide d'un ensemble d'expérimentations sur le même dataset "Test Set" utilisé à l'étape de précédente, ou nous vérifions si le système fonctionne, tout en classifiant correctement les réponses acquises.

L'évaluation du modèle a été faite sur la base de 2 métriques qui sont : **Loss** (Erreur) et **Accuracy**. Ci-dessous les graphes des résultats obtenus.



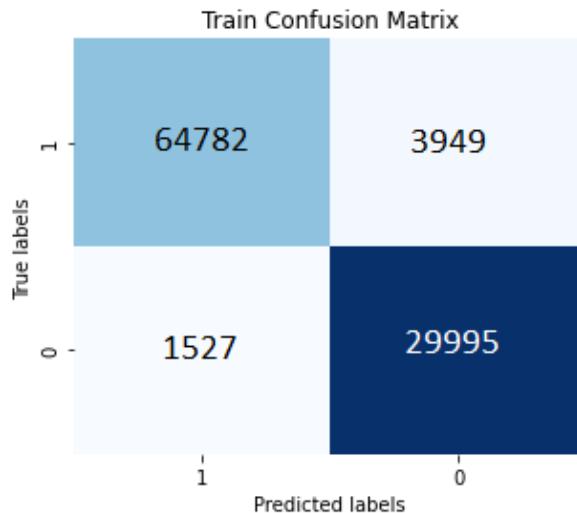
(a) Accuracy



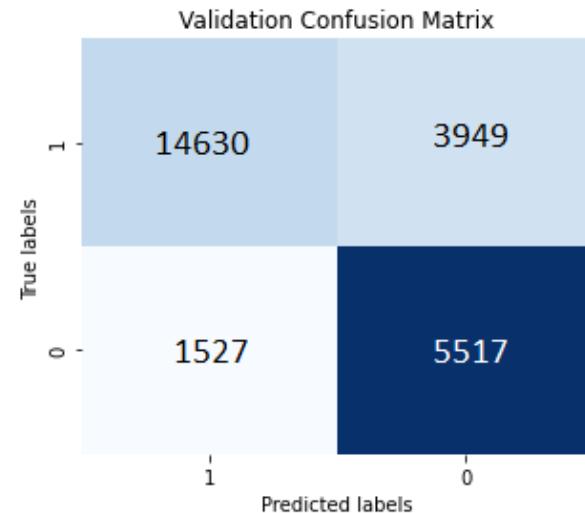
(b) Loss

FIGURE 4.3: Graphes d'évaluation du Module de Classification.

D'après la Figure 4.3, nous remarquons que l'*Accuracy* augmente avec le nombre d'époques, jusqu'à l'Epoch (6), avec un pourcentage de **90%** durant l'apprentissage (*Training*) et **78%** sur le *Validation dataset*. De même, l'erreur d'apprentissage (*Training*) et de la *Validation* varient avec le nombre d'Epochs, où le *Training Loss* diminue jusqu'à l'atteinte de **0.47** à la fin de cette étape et le *Validation Loss* se stabilise au bout de 6 Epochs, dans la valeur **0.59**.



(a) Training



(b) Validation

FIGURE 4.4: Matrices de confusion du MC (Train/Validation).

D'après la figure 4.4, nous remarquons qu'en fin d'apprentissage, la totalité des réponses mal classées est de **10 518** réponses, un taux d'erreur de **9%** et la totalité des réponses bien classées est de **109 964**, donc un taux de précision de **91%**. Nous notons également que le nombre de réponses mal classées durant la validation est de **1992** réponses ce qui est égal à un taux d'erreur de **20%** et le nombre de réponses bien classées est de **109 964**, de ce fait, un taux de précision de **80%**.

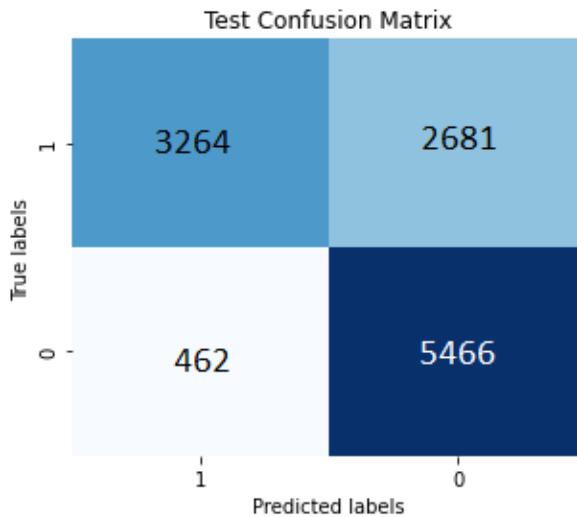


FIGURE 4.5: Matrice de confusion du MC (Test).

Le MC proposé a atteint une **Accuracy** de **74%** sur le Test-set. Les taux d'erreur et de précision sont égal à % et % resp (Figure 4.5).

4.5 Résultats du module d'extraction des réponses MER

Dans cette section, nous allons discuter l'exactitude des résultats du MER. Pour ce faire, nous avons calculé les deux métriques : **Loss** (Erreur) et **Accuracy**. Durant l'apprentissage et la validation du modèle, nous avons tracés les courbes pour les deux sous modèles de détection du **début** et de **fin** de réponse.

Nous observons sur les graphiques figures 4.6 que la courbe de **Training Accuracy** pour la détection du début a atteint une valeur de **75%** au bout de 14 Epochs, tandis que la **Validation Accuracy** s'est stabilisé à partir du *9ème* Epoch à un pourcentage de **70%**. Nous enregistrons en parallèle la diminution du **Training Loss** de ce modèle jusqu'à la valeur **0.75** au *15ème* Epoch et le **Validation Loss** qui s'est figé à la valeur **1.12** à partir du *8ème* Epoch.

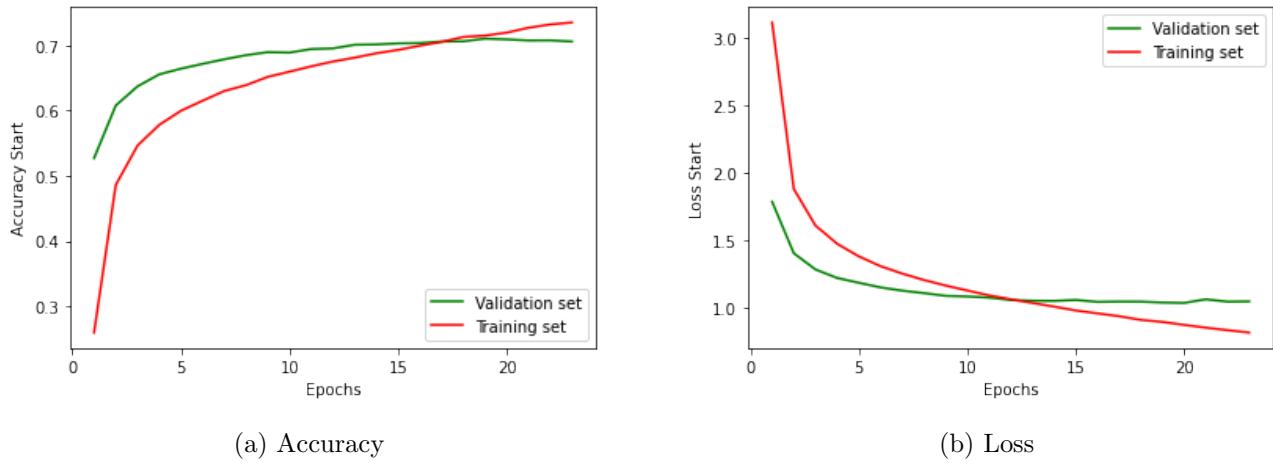


FIGURE 4.6: Graphes d'évaluation du Module d'Extraction de Réponse (Start).

Les courbes de **Accuracy** et **Loss** du deuxième sous-modèle de détection de fin sont présentés dans la figure 4.7. Nous remarquons dans ces dernières que la détection du début sur la **Training Accuracy** a atteint une valeur de **80%** après 14 epochs d'apprentissage. Tandis que la **Validation Accuracy** a accru jusqu'à **73%** au **9ème Epoch**. Nous notons également que le **Training Loss** de ce modèle a diminué jusqu'à la valeur **0.75** au **15ème Epoch** et le **Validation Loss** est **1.12** dès le **8ème Epoch**.

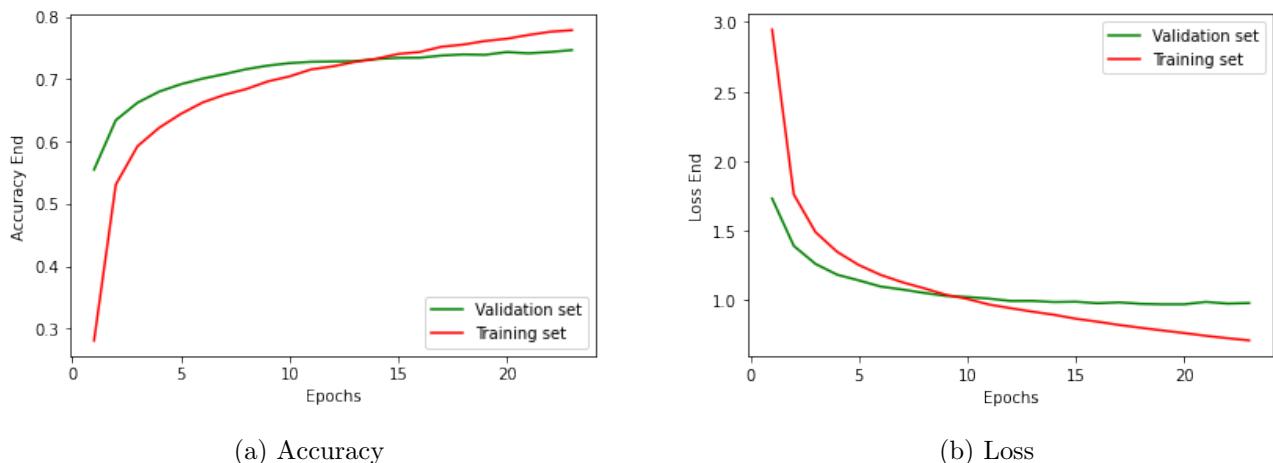


FIGURE 4.7: Graphes d'évaluation du Module d'Extraction de Réponse (End).

Dans le but de tester l'efficacité de notre modèle, nous avons utilisé l'évaluateur proposé par SQuAD¹, qui prend comme entrée un fichier contenant les réponses extraites à partir des contextes de SQuAD pour

1. <https://rajpurkar.github.io/SQuAD-explorer/>

chaque question du Test Set, et donne en sortie les métriques **F1-Score** et **Exact Match**. les résultats enregistrés sont les suivant :

- “exact” : 86.85,
- “f1” : 91.58,
- “total” : 11873,
- “HasAns_total” : 5928.

Les résultats présentés ci-dessus montrent que le **Test Set** contient **5928 questions** ayant une réponse dans leurs contextes d'un total de **11873 questions**. Notre MER a pu atteindre un **Exact Match** de ~ 87% et un **F1-Score** de ~ 92%.

4.6 Résultats globaux

Les résultats présentés de l'évaluateur montrent que sur les **5928 questions** du **Test Set**, le système **YouTaQA** un atteint un **Exact Match** de ~ xx% et un **F1-Score** de ~ xx%.

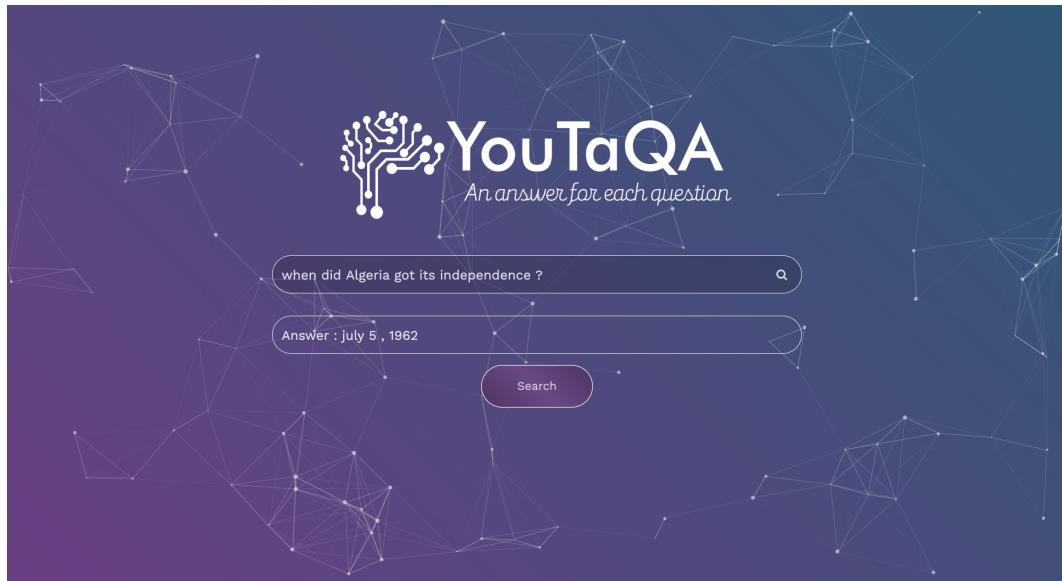


FIGURE 4.8: Exemple d'utilisation du système YouTaQA

Nous estimons que notre système affiche des performances globales acceptables en prenant en compte la marge d'erreur et les limites liées à la source d'information de notre système ainsi qu'un temps d'exécution moyen de 3 ~ 4 secondes seulement.

4.7 Conclusion

Dans ce dernier chapitre, nous avons présenté les résultats donnés par l'implémentation de notre architecture présentée dans le chapitre précédent. Nous avons testé nos modules et avons présenté et discuté les résultats obtenus en calculant des métriques d'évaluation et en présentant des graphes afin de faciliter la comparaison des expérimentations.

Conclusion Générale

Le travail qui nous a été confié dans le cadre de notre projet de fin d'études, consiste à concevoir et réaliser un système complet de questions-réponses basé sur la recherche d'informations.

Le but principal de notre système était de fournir une solution complète qui, après l'identification des problèmes rencontrés dans les travaux précédents, faciliterais la réponse aux questions des utilisateurs. La tâche principale des systèmes QAS existants étant de répondre aux questions en exigeant tout de même un passage dans lequel il extrairait la réponse, notre système permettrait aux utilisateurs d'automatiser la fourniture du passage en utilisant un moteur de recherche performant et rapide.

Nous nous sommes appuyés sur l'application des méthodes innovantes basées sur le Deep Learning afin de décrire et éliminer avec succès les anomalies et les problèmes que nous avons rencontrés au fil des étapes du développement de la solution.

Pour but de réaliser le pipeline complet de notre système nommé YouTaQA, nous avons tout d'abord implémenté un moteur de recherche en utilisant la base de documents « Wikipédia » comme source d'informations, tout en considérant chaque section d'article comme étant un document en lui-même afin d'affiner les résultats de recherche. Ensuite, en se basant sur le « Deep Learning » et spécialement le modèle BERT qui permet d'atteindre l'état de l'art à l'heure où cette thèse a été rédigée, nous avons bâti un module de classification de documents qui sert à reclasser les passages résultants de notre moteur de recherche en lui fournissant la question et les 5 meilleurs passages, cela nous permet à chaque fois d'identifier le passage le plus probable et d'avoir une réponse exacte à une question donnée.

Étant notre Classifieur basé sur le « Deep Learning », nous avons été amenés à faire un entraînement à ce dernier, ce qui nous a poussé à choisir le DataSet de QNLI qui fournit plus de 66.000 paires de question-passage et qui précise à chaque paire si le passage contient une réponse à la question. Après l'avoir entraîné, notre Classifieur permettait désormais des performances convergentes vers l'état de l'art actuel en affichant un score F1 égale à 80% ce qui est plus qu'acceptable en prenant compte la difficulté des tâches NLP et de la classification du texte en particulier.

Ensuite, et après avoir choisi le meilleur passage susceptible à contenir une réponse à la question posée par l'utilisateur, la tâche la plus importante était d'extraire la réponse exacte à cette question, d'où le module d'extraction des réponses MER a été réalisé afin d'accomplir cette tâche. Le module d'extraction des réponses a été implémenté en utilisant le modèle BERT et a été entraîné en s'appuyant sur le DataSet SQuAD réalisé par l'université de Stanford et qui fournit plus de 100.000 paires de question-réponse. Le module MER affiche un score F1 égale à 84% en le testant sur plus de 50.000 questions, ce score représente une contribution qui pourrait être ajoutée à la littérature.

Afin de tout concrétiser, et de fournir une expérience utilisateur digne d'un système haut niveau, nous avons implémenté une application web moderne qui satisfait les normes internationales de design.

Perspectives

Les travaux rapportés dans cette thèse ne composent qu'un simple morceau dans un puzzle de travaux supplémentaires qui doivent être réalisés pour aboutir à un système parfait. Pour cela, les perspectives envisageables afin d'améliorer ce travail sont multiple, nous citons :

- Améliorer notre moteur de recherche en employant des méthodes basées sur le Deep Learning.
- Implémenter un module de mise à jour automatique qui permet d'actualiser la base des documents Wikipédia régulièrement afin d'envisager de répondre aux questions liées aux nouveaux sujets.
- Mettre en œuvre la version arabe et française de notre système YouTaQA pour but d'atteindre une plus large communauté.
- Implémenter une API afin de faciliter l'utilisation du noyau YouTaQA dans des applications tierces.

Pour finir, *le savoir est la seule matière qui s'accroisse quand on la partage*, comme le dit Socrate, sur ce, nous avons mis à disposition notre code source sur GitHub : <https://github.com/rbouadjenek/YouTaQA>.

Bibliographie

- [RNN, 2017] (2017). Basic architecture of rnn and lstm. <https://pydeeplearning.weebly.com/blog/basic-architecture-of-rnn-and-lstm>. Accessed : 2020-07-25.
- [VSM, 2017] (2017). information retrieval document search using vector space model in r. <http://www.dataperspective.info/2017/11/information-retrieval-document-search-using-vector-space-model-in-r.html>. Accessed : 2020-07-29.
- [Tra, 2018] (2018). The illustrated transformer. <https://developpaper.com/the-calculation-of-self-attention-in-bert-series/>. Accessed : 2020-07-25.
- [Fin, 2019] (2019). Ideas on how to fine-tune a pre-trained model in pytorch. <https://medium.com/udacity-pytorch-challengers/ideas-on-how-to-fine-tune-a-pre-trained-model-in-pytorch-184c47185a20>.
- [Sel, 2020] (2020). The calculation of self attention in bert series. <https://developpaper.com/the-calculation-of-self-attention-in-bert-series/>. Accessed : 2020-07-25.
- [Wik, 2020] (2020). Wikipedia :size comparisons. https://en.wikipedia.org/wiki/Wikipedia:Size_comparisons. Accessed : 2020-07-25.
- [Abadi et al., 2016] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. (2016). Tensorflow : A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*, pages 265–283.
- [Adamic et al., 2008] Adamic, L. A., Zhang, J., Bakshy, E., and Ackerman, M. S. (2008). Knowledge sharing and yahoo answers. In *Proceeding of the 17th international conference on World Wide Web - WWW '08*. ACM Press.

- [Ahn et al., 2004] Ahn, D., Jijkoun, V., Mishne, G., Müller, K., Rijke, M., and Schlobach, S. (2004). Using wikipedia at the trec qa track.
- [Baeza-Yates and Ribeiro-Neto, 2011] Baeza-Yates, R. A. and Ribeiro-Neto, B. A. (2011). *Modern Information Retrieval - the concepts and technology behind search, Second edition*.
- [Bahdanau et al., 2015] Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate.
- [Banko et al., 2002] Banko, M., Brill, E., Dumais, S., and Lin, J. (2002). Askmsr : Question answering using the worldwide web. In *Proceedings of 2002 AAAI Spring Symposium on Mining Answers from Texts and Knowledge Bases*, pages 7–9.
- [Baudiš, 2015] Baudiš, P. (2015). Yodaqa : a modular question answering system pipeline. In *POSTER 2015-19th International Student Conference on Electrical Engineering*, pages 1156–1165.
- [Baudiš and Šedivý, 2015] Baudiš, P. and Šedivý, J. (2015). Modeling of the question answering task in the yodaqa system. In *Proceedings of the 6th International Conference on Experimental IR Meets Multilinguality, Multimodality, and Interaction - Volume 9283*, CLEF’15, page 1156–1165, Berlin, Heidelberg. Springer-Verlag.
- [Bauer and Berleant, 2012] Bauer, M. A. and Berleant, D. (2012). Usability survey of biomedical question answering systems. *Human Genomics*, 6(1).
- [Beitzel et al., 2009] Beitzel, S. M., Jensen, E. C., and Frieder, O. (2009). *MAP*, pages 1691–1692. Springer US, Boston, MA.
- [Benamara and Saint-Dizier, 2003] Benamara, F. and Saint-Dizier, P. (2003). Webcoop : A cooperative question-answering system on the web.
- [Berant et al., 2013] Berant, J., Chou, A., Frostig, R., and Liang, P. (2013). Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.
- [Bian et al., 2017] Bian, W., Li, S., Yang, Z., Chen, G., and Lin, Z. (2017). A compare-aggregate model with dynamic-clip attention for answer selection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM ’17, page 1987–1990, New York, NY, USA. Association for Computing Machinery.
- [Blier et al., 2018] Blier, L., Wolinski, P., and Ollivier, Y. (2018). Learning with random learning rates.

- [Bollacker et al., 2008] Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase : A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, page 1247–1250, New York, NY, USA. Association for Computing Machinery.
- [Bordes et al., 2015] Bordes, A., Usunier, N., Chopra, S., and Weston, J. (2015). Large-scale simple question answering with memory networks. *ArXiv*, abs/1506.02075.
- [Bouadjenek et al., 2016] Bouadjenek, M. R., Hacid, H., and Bouzeghoub, M. (2016). Social networks and information retrieval, how are they converging ? a survey, a taxonomy and an analysis of social information retrieval approaches and platforms. *Information Systems*, 56.
- [Bouma G., 2005] Bouma G., Mur J., v. N. G. (2005). Reasoning over dependency relations.
- [Brill et al., 2002] Brill, E., Dumais, S., and Banko, M. (2002). An analysis of the AskMSR question-answering system. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP 2002)*, pages 257–264. Association for Computational Linguistics.
- [Buscaldi and Rosso, 2006] Buscaldi, D. and Rosso, P. (2006). Mining knowledge from wikipedia for the question answering task. In *In Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC'06*, page 727–730.
- [Cabrio et al., 2012] Cabrio, E., Cojan, J., Aprosio, A. P., Magnini, B., Lavelli, A., and Gandon, F. (2012). Qakis : an open domain qa system based on relational patterns.
- [Cao et al., 2011] Cao, Y., Liu, F., Simpson, P., Antieau, L., Bennett, A., Cimino, J. J., Ely, J., and Yu, H. (2011). AskHERMES : An online question answering system for complex clinical questions. *Journal of Biomedical Informatics*, 44(2) :277–288.
- [Chen et al., 2016] Chen, D., Bolton, J., and Manning, C. D. (2016). A thorough examination of the cnn/daily mail reading comprehension task. *CoRR*, abs/1606.02858.
- [Chen et al., 2017] Chen, D., Fisch, A., Weston, J., and Bordes, A. (2017). Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv :1704.00051*.
- [Chen et al., 2001] Chen, K., Kvasnicka, V., Kanen, P., and Haykin, S. (2001). Feedforward neural network methodology. *IEEE Transactions on Neural Networks*, 12(3) :647–648.
- [Chollet et al., 2015] Chollet, F. et al. (2015). Keras.
- [Craswell, 2009a] Craswell, N. (2009a). *Mean Reciprocal Rank*, pages 1703–1703. Springer US, Boston, MA.

- [Craswell, 2009b] Craswell, N. (2009b). *R-Precision*, pages 2453–2453. Springer US, Boston, MA.
- [Craswell, 2009c] Craswell, N. (2009c). *R-Precision*, pages 2453–2453. Springer US, Boston, MA.
- [Dalip et al., 2013] Dalip, D. H., Gonçalves, M. A., Cristo, M., and Calado, P. (2013). Exploiting user feedback to learn to rank answers in q&a forums. In *Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval - SIGIR '13*. ACM Press.
- [Devlin et al., 2018] Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). Bert : Pre-training of deep bidirectional transformers for language understanding.
- [Diekema et al., 2004] Diekema, A. R., Yilmazel, O., and Liddy, E. D. (2004). Evaluation of restricted domain question-answering systems. In *Proceedings of the Conference on Question Answering in Restricted Domains*, pages 2–7, Barcelona, Spain. Association for Computational Linguistics.
- [ElKafrawy et al., 2018] ElKafrawy, P. M., Sauber, A. M., and Sabry, N. A. (2018). Semantic question answering system using dbpedia. In *Lecture Notes in Computer Science*, pages 821–832. Springer International Publishing.
- [Fader et al., 2014] Fader, A., Zettlemoyer, L., and Etzioni, O. (2014). Open question answering over curated and extracted knowledge bases. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, page 1156–1165, New York, NY, USA. Association for Computing Machinery.
- [Ferrucci et al., 2010] Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A., Lally, A., Murdock, J. W., Nyberg, E., Prager, J., Schlaefer, N., and Welty, C. (2010). Building watson : An overview of the deepqa project. *AI Magazine*, 31 :59–79.
- [Garg et al., 2019] Garg, S., Vu, T., and Moschitti, A. (2019). Tanda : Transfer and adapt pre-trained transformer models for answer sentence selection.
- [Gazan, 2016] Gazan, R. (2016). Seven words you can't say on answerbag. In *Proceedings of the 27th ACM Conference on Hypertext and Social Media - HT '16*. ACM Press.
- [Green et al., 1961] Green, B. F., Wolf, A. K., Chomsky, C., and Laughery, K. (1961). Baseball. In *Papers presented at the May 9-11, 1961, western joint IRE-AIEE-ACM computer conference on - IRE-AIEE-ACM '61 (Western)*. ACM Press.
- [Greff et al., 2016] Greff, K., Srivastava, R. K., Koutník, J., Steunebrink, B. R., and Schmidhuber, J. (2016). Lstm : A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10) :2222–2232.

- [Harabagiu et al., 2001] Harabagiu, S., Moldovan, D., Pasca, M., Mihailea, R., Surdeanu, M., Bunescu, R., Girju, R., Rus, V., and Morarescu, P. (2001). Falcon : Boosting knowledge for question answering.
- [Hermann et al., 2015] Hermann, K. M., Kociský, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching machines to read and comprehend. *CoRR*, abs/1506.03340.
- [Hewlett et al., 2016] Hewlett, D., Lacoste, A., Jones, L., Polosukhin, I., Fandrianto, A., Han, J., Kelcey, M., and Berthelot, D. (2016). WikiReading : A novel large-scale language understanding task over Wikipedia. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, pages 1535–1545, Berlin, Germany. Association for Computational Linguistics.
- [Hill et al., 2015] Hill, F., Bordes, A., Chopra, S., and Weston, J. (2015). The goldilocks principle : Reading children’s books with explicit memory representations. *CoRR*, abs/1511.02301.
- [Hochreiter, 1998] Hochreiter, S. (1998). The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 06(02) :107–116.
- [Iyyer et al., 2014] Iyyer, M., Boyd-Graber, J., Claudino, L., Socher, R., and Daumé III, H. (2014). A neural network for factoid question answering over paragraphs. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 633–644, Doha, Qatar. Association for Computational Linguistics.
- [Jain et al., 2018] Jain, A., Kulkarni, G., and Shah, V. (2018). Natural language processing. *International Journal of Computer Sciences and Engineering*, 6 :161–167.
- [Kaisser, 2005] Kaisser, M. (2005). Qualim at trec 2005 : Web-question answering with framenet. In *TREC*.
- [Katz et al., 2002] Katz, B., Felshin, S., Yuret, D., Lin, J., Marton, G., McFarland, A., and Temelkuran, B. (2002). Omnibase : Uniform access to heterogeneous data for question answering. pages 230–234.
- [Kolomiyets and Moens, 2011a] Kolomiyets, O. and Moens, M.-F. (2011a). A survey on question answering technology from an information retrieval perspective. *Information Sciences*, 181(24) :5412–5434.
- [Kolomiyets and Moens, 2011b] Kolomiyets, O. and Moens, M.-F. (2011b). A survey on question answering technology from an information retrieval perspective. *Information Sciences*, 181(24) :5412–5434.
- [Kowsari et al., 2019] Kowsari, K., Meimandi, K. J., Heidarysafa, M., Mendu, S., Barnes, L. E., and Brown, D. E. (2019). Text classification algorithms : A survey. *CoRR*, abs/1904.08067.

- [Kumaran and Carvalho, 2009] Kumaran, G. and Carvalho, V. R. (2009). Reducing long queries using query quality predictors. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '09*, page 564–571, New York, NY, USA. Association for Computing Machinery.
- [Kwok et al., 2001] Kwok, C., Etzioni, O., and Weld, D. S. (2001). Scaling question answering to the web. *ACM Trans. Inf. Syst.*, 19(3) :242–262.
- [Lan et al., 2019] Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., and Soricut, R. (2019). Albert : A lite bert for self-supervised learning of language representations.
- [Lee et al., 2006] Lee, M., Cimino, J., Zhu, H., Sable, C., Shanker, V., Ely, J., and Yu, H. (2006). Beyond information retrieval—medical question answering. *AMIA Annu Symp Proc*, pages 469–473.
- [Leskovec et al., 2014] Leskovec, J., Rajaraman, A., and Ullman, J. D. (2014). *Finding Similar Items*, page 68–122. Cambridge University Press, 2 edition.
- [Lopez et al.,] Lopez, V., Motta, E., Sabou, M., and Fernandez, M. Poweraqua : A multi-ontology based question answering system—v1.
- [Loshchilov and Hutter, 2017] Loshchilov, I. and Hutter, F. (2017). Decoupled weight decay regularization. *arXiv preprint arXiv :1711.05101*.
- [Mervin, 2013] Mervin, R. (2013). An overview of question answering system. 1.
- [Moldovan et al., 1999] Moldovan, D., Harabagiu, S., Pasca, M., Mihalcea, R., Goodrum, R., Girju, R., and Rus, V. (1999). Lasso : A tool for surfing the answer net.
- [Moldovan et al., 2003] Moldovan, D., Pașca, M., Harabagiu, S., and Surdeanu, M. (2003). Performance issues and error analysis in an open-domain question answering system. *ACM Transactions on Information Systems (TOIS)*, 21(2) :133–154.
- [Molla Aliod and González, 2007] Molla Aliod, D. and González, J. (2007). Question answering in restricted domains : An overview. *Computational Linguistics*, 33 :41–61.
- [Nam et al., 2009] Nam, K. K., Ackerman, M. S., and Adamic, L. A. (2009). Questions in, knowledge in ? In *Proceedings of the 27th international conference on Human factors in computing systems - CHI '09*. ACM Press.
- [Ojokoh and Adebisi, 2019] Ojokoh, B. and Adebisi, E. (2019). A review of question answering systems. *Journal of Web Engineering*, 17 :717–758.

- [Olvera-Lobo and Gutiérrez-Artacho, 2011] Olvera-Lobo, M.-D. and Gutiérrez-Artacho, J. (2011). Multilingual question-answering system in biomedical domain on the web : An evaluation. In *Multilingual and Multimodal Information Access Evaluation*, pages 83–88. Springer Berlin Heidelberg.
- [Olvera-Lobo and Gutiérrez-Artacho, 2011] Olvera-Lobo, M. D. and Gutiérrez-Artacho, J. (2011). Open- vs. restricted-domain qa systems in the biomedical field. *Journal of Information Science*, 37 :152–162.
- [Pan and Yang, 2010] Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *ieee transactions on knowledge and data engineering*. 22 (10) : 1345, 1359.
- [Paszke et al., 2019] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. (2019). Pytorch : An imperative style, high-performance deep learning library. In *Advances in neural information processing systems*, pages 8026–8037.
- [Qaiser and Ali, 2018] Qaiser, S. and Ali, R. (2018). Text mining : Use of tf-idf to examine the relevance of words to documents. *International Journal of Computer Applications*, 181.
- [Radford et al., 2018] Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding by generative pre-training.
- [Rajpurkar et al., 2016] Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. (2016). Squad : 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250.
- [Reddy and Madhavi, 2017] Reddy, A. C. O. and Madhavi, K. (2017). A survey on types of question answering system. *IOSR-JCE*, 19(6) :19–23.
- [Richardson et al., 2013] Richardson, M., Burges, C. J., and Renshaw, E. (2013). MCTest : A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 193–203, Seattle, Washington, USA. Association for Computational Linguistics.
- [Robertson and Zaragoza, 2009] Robertson, S. and Zaragoza, H. (2009). The probabilistic relevance framework : Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3 :333–389.
- [Rodrigues and Milic-Frayling, 2009] Rodrigues, E. M. and Milic-Frayling, N. (2009). Socializing or knowledge sharing ? In *Proceeding of the 18th ACM conference on Information and knowledge management - CIKM '09*. ACM Press.
- [Ryu et al., 2014] Ryu, P.-M., Jang, M.-G., and Kim, H.-K. (2014). Open domain question answering using wikipedia-based knowledge model. *Inf. Process. Manage.*, 50(5) :683–692.
- [Sak et al., 2014] Sak, H., Senior, A. W., and Beaufays, F. (2014). Long short-term memory recurrent neural network architectures for large scale acoustic modeling.

- [Seo et al., 2016] Seo, M., Kembhavi, A., Farhadi, A., and Hajishirzi, H. (2016). Bidirectional attention flow for machine comprehension.
- [Shortliffe, 1977] Shortliffe, E. (1977). Mycin : A knowledge-based computer program applied to infectious diseases*. *Proceedings / the ... Annual Symposium on Computer Application [sic] in Medical Care. Symposium on Computer Applications in Medical Care*.
- [Sun et al., 2015] Sun, H., Ma, H., Yih, W.-t., Tsai, C.-T., Liu, J., and Chang, M.-W. (2015). Open domain question answering via semantic enrichment. In *Proceedings of the 24th International Conference on World Wide Web, WWW '15*, page 1045–1055, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.
- [Tay et al., 2018] Tay, Y., Tuan, L. A., and Hui, S. C. (2018). Hyperbolic representation learning for fast and efficient neural question answering. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining - WSDM '18*. ACM Press.
- [Teufel, 2007] Teufel, S. (2007). *An Overview of Evaluation Methods in TREC Ad Hoc Information Retrieval and TREC Question Answering*, volume 37, pages 163–186.
- [Ting, 2010] Ting, K. M. (2010). *Precision and Recall*, pages 781–781. Springer US, Boston, MA.
- [Vargas-Vera and Motta, 2004] Vargas-Vera, M. and Motta, E. (2004). Aqua – ontology-based question answering system. In Monroy, R., Arroyo-Figueroa, G., Sucar, L. E., and Sossa, H., editors, *MICAI 2004 : Advances in Artificial Intelligence*, pages 468–477, Berlin, Heidelberg. Springer Berlin Heidelberg.
- [Vaswani et al., 2017] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need.
- [Voita et al., 2019] Voita, E., Talbot, D., Moiseev, F., Sennrich, R., and Titov, I. (2019). Analyzing multi-head self-attention : Specialized heads do the heavy lifting, the rest can be pruned. *arXiv preprint arXiv :1905.09418*.
- [Voorhees and Tice, 2000] Voorhees, E. and Tice, D. (2000). The trec-8 question answering track evaluation. *Proceedings of the 8th Text Retrieval Conference*.
- [Wang et al., 2018] Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. (2018). GLUE : A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP : Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.
- [Wong, 2007] Wong, W. (2007). Practical approach to knowledge-based question answering with natural language understanding and advanced reasoning.

- [Woods, 1973] Woods, W. A. (1973). Progress in natural language understanding. In *Proceedings of the June 4-8, 1973, national computer conference and exposition on - AFIPS '73*. ACM Press.
- [yacine, 2018] yacine, D. (2018). Recherche d'information : Prétraitement et indexation.
- [Yang et al., 2015] Yang, Y., Yih, W.-t., and Meek, C. (2015). WikiQA : A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018, Lisbon, Portugal. Association for Computational Linguistics.
- [Yang et al., 2019] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., and Le, Q. V. (2019). Xlnet : Generalized autoregressive pretraining for language understanding.
- [Yoon et al., 2019a] Yoon, S., Dernoncourt, F., Kim, D., Bui, T., and Jung, K. (2019a). A compare-aggregate model with latent clustering for answer selection. pages 2093–2096.
- [Yoon et al., 2019b] Yoon, S., Dernoncourt, F., Kim, D. S., Bui, T., and Jung, K. (2019b). A compare-aggregate model with latent clustering for answer selection. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, CIKM '19, page 2093–2096, New York, NY, USA. Association for Computing Machinery.
- [Yu et al., 2018] Yu, A. W., Dohan, D., Luong, M.-T., Zhao, R., Chen, K., Norouzi, M., and Le, Q. V. (2018). Qanet : Combining local convolution with global self-attention for reading comprehension.
- [Zhang et al., 2020] Zhang, Z., Yang, J., and Zhao, H. (2020). Retrospective reader for machine reading comprehension.
- [Zheng, 2003] Zheng, Z. (2003). Question answering using web news as knowledge base. In *Proceedings of the Tenth Conference on European Chapter of the Association for Computational Linguistics - Volume 2*, EACL '03, page 251–254, USA. Association for Computational Linguistics.

Abstract

Users' need for comfort and the demand to have accurate answers to their questions are present nowadays which has given a new purpose to artificial intelligence. The best known search engines such as Google tend to offer brief answers to so-called "factoid" questions. This task is considered difficult in terms of the complexity of the queries and even their answers which can be the combination of several passages.

For this, in this thesis, our goal is based on the design and implementation of a Question-Answering system that can overcome the difficulties mentioned above and that is able to answer questions in several areas accurately and precisely using the Wikipedia knowledge base. Our system named YouTAQA starts by collecting the passages that can answer the query entered by the user and ends by extracting the start and end of the exact answer using Deep Learning. That being said, our system is capable of doing the complete pipeline, from collecting the relevant passages, to extracting the final answer requiring only the question as input. The Deep Learning modules of our system were implemented using the pre-trained BERT model which has been designed to perform various NLP tasks.

Our search engine, using the Information Retrieval techniques was able to achieve a MRR and MAP score of 0.20 which means that the most relevant document is found in one of the first five results produced. The passage classification model reached an F1 score of 80% while the response extraction model reached an F1 score of 84%.

Experiments on the dataset have demonstrated the effectiveness of the proposed method and the results of the comparison have shown that our architecture improved the Question-Answering domain.

Keywords : Information Retrieval, Deep Learning, Natural Language Processing, Transfer Learning.

ملخص

حاجة المستخدمين للراحة والمطالبة بالحصول على إجابات دقيقة لأسئلتهم موجودة في الوقت الحاضر والتي أعطت غرضاً جديداً للذكاء الاصطناعي. تمثل أفضل محركات البحث المعروفة مثل جوجل إلى تقديم إجابات موجزة. تعتبر هذه المهمة صعبة من حيث تعقيد الأسئلة وحتى إجاباتها التي يمكن أن تكون مزيجاً من عدة أجوبة.

لهذا ، في هذه المذكورة ، يعتمد هدفنا على تصميم وتنفيذ نظام للإجابة على الأسئلة يمكنه التغلب على الصعوبات المذكورة أعلاه والقادر على الإجابة على الأسئلة في العديد من المجالات بدقة باستخدام موسوعة ويكيبيديا. يبدأ نظامنا المسمى YouTAQA بتجمیع المقاطع التي يمكنها الإجابة على السؤال الذي أدخله المستخدم وينتهي باستخراج بداية ونهاية الإجابة الدقيقة باستخدام تقنية الـ Deep Learning. ومع ذلك ، فإن نظامنا قادر على القيام بكل خطوات البحث ، من جمع الأجوبة ذات صلة ، إلى استخراج الإجابة النهائية التي تتطلب فقط السؤال كمدخل. تم تدريب نظامنا باستخدام نموذج BERT مسبق التدريب الذي تم تصميمه لأداء مهام البرمجة اللغوية المختلفة.

كان محرك البحث الخاص بنا ، باستخدام تقنيات استرجاع المعلومات ، قادراً على تحقيق درجة MRR و MAP بنسبة 0.20 مما يعني أنه تم العثور على الإجابة الأكثر صلة في إحدى النتائج الخمس الأولى التي تم إنتاجها. وصل نظام تصنيف الأجوبة إلى نسبة F1 تبلغ 0.80% بينما وصل نظام استخراج الاستجابة إلى درجة F1 تبلغ 0.84%.

توضح التجارب على مجموعة البيانات فعالية الطريقة المقترنة ، وتظهر نتائج المقارنة أن بنيتنا تعطي مجالاً أكثر للإجابة على الأسئلة.

الكلمات المفتاحية: البحث على المعلومات، تعلم عميق، معالجة اللغة الطبيعية، نقل التعلم.