

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 Informatiques
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'informations

Devant le jury composé de :

Mme. M.....	Pr. Université Alger 1	Président
Mme. ZIANI Amel	Professeur. Université Alger 1	Encadreur
M. BOUADJENEK Mohamed Reda	Pr. Deakin University	Co-Encadreur
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 encadreurs M. Mohamed reda Bouadjenek et Mme. Amel Ziani qui ont bien voulu accepter de diriger et d'encadrer ce travail, également pour leurs patiences, 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 tous le personnel qui nous ont apporté aide et assistance et donné toutes les informations dont nous avions besoin pour la réalisation de cette thèse.

Un grand merci à nos familles, surtout nos parents qui nous ont soutenus et suivis tout au long de ce projet.

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.

Nos remerciements vont aussi à Asmaa Agabi, Akram Arar, Mounir Grar, Fares Aliliche, Islem Krim, Oussama Hamada et à tous ceux qui ont contribué de près ou de loin à la concrétisation de ce travail. Qu'ils trouvent tous ici l'expression de notre gratitude et notre parfaite considération.

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 binôme Younes qui a partagé avec moi cette expérience et à notre encadrant Dr.BOUADJENEK Mohamed Reda qui a toujours été présent et nous a donnée les meilleurs conseils pour pouvoir compléter notre travail comme nous l'avons toujours souhaité, je le remercie pour sa patience, son aide et la confiance qu'il nous a témoignés.

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

À mon cousin Iheb Tekkour qui est un grand frère pour moi, ma tante Nassira , papa Ahmed, 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 et M. Tali, 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, Ous-sama 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 ne vous souhaite que 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 ci-dessus et qui est apte de répondre aux questions dans plusieurs domaines de façon exacte et précise en utilisant la base de connaissances de Wikipédia. Notre système nommé YouTAQA 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 est capable de faire le pipeline complet, à partir de la collecte des passages pertinents, jusqu'à l'extraction de la réponse finale ne nécessitant que la question comme entrée. Les modules d'apprentissage approfondi de notre système ont été implementés en utilisant le modèle pré-entraîné BERT qui a été conçu pour réaliser différentes tâches de NLP.

Notre moteur de recherche a pu atteindre un score MRR et MAP de 0.20 ce qui veut dire que le document le plus pertinent se trouve dans l'un de cinq premiers résultats produits. Le modèle de classification des passages a atteint un F1 score de xx% tandis que le modèle d'extraction de réponse a un F1 score de xx%. Les expériences 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 de domaine du Question-Answering.

Mots clés : Recherche d'informations, Apprentissage Approfondi, Traitement de langage naturel, BERT.

Table des matières

Introduction générale	8
1 Généralités	13
1.1 Introduction	13
1.2 Recherche d'informations	13
1.2.1 Définition de la recherche d'informations	13
1.2.2 L'indexation	13
1.2.3 Les étapes d'indexation	14
1.2.4 Requêteage d'informations	16
1.2.5 Classement des résultats	17
1.2.6 Les outils d'évaluation	19
1.2.7 Les outils de développement	19
1.3 Natural Language Processing	19
1.3.1 Définition du traitement du langage naturel	19
1.3.2 Word Embedding	20
1.4 Deep Learning	20
1.4.1 Attention	21
1.4.2 Les Transformateurs	22
1.4.3 BERT (Bidirectional Encoder Representations from Transformers)	24
1.5 Les métriques d'évaluation	26
1.6 Conclusion	28

TABLE DES MATIÈRES

2 Etat de l'art	29
2.1 Introduction	29
2.2 Classification selon le domaine d'application	29
2.3 Closed domain QA Systems	30
2.3.1 Open-domain QA Systems	31
2.3.2 Social QA Systems	31
2.4 Classification selon d'autres critères	33
2.5 Conclusion	36
3 Présentation de la solution	37
3.1 Introduction	37
3.2 Moteur de recherche MRI	37
3.2.1 Choix de la base des documents	38
3.2.2 Prétraitement de la base Wikipédia	39
3.2.3 Indexation des articles	40
3.2.4 Méthodes de recherche adoptées	42
3.3 Module de classification	42
3.3.1 Choix du jeu de données	42
3.3.2 Entrainement du classifieur	42
3.4 Module d'extraction des réponses MER	44
3.4.1 Le choix du jeu de données	44
3.4.2 Entrainement de l'extracteur	45
3.5 Conclusion	47
4 Analyse et discussion des résultats	48
4.1 Introduction	48
4.2 Résultats du module de recherche d'informations MRI	48
4.2.1 Fichier Qrels	48
4.2.2 Fichier Résultats	49
4.2.3 Méthodes de recherche employées	49
4.2.4 Discussion des performances des méthodes de recherche	51
4.3 Résultats du module de classification des documents	56
4.4 Résultats du module d'extraction des réponses MER	56

TABLE DES MATIÈRES

4.5 Résultats globaux	56
4.6 Conclusion	56
Conclusion Générale	57
4.7 Perspectives	58
Bibliographie	59

Table des figures

1.1	Les étapes d'indexation	14
1.2	Architecture de l'index inversé[1].	15
1.3	Processus de Requêtage d'informations	16
1.4	Similarité VSM entre deux documents et une requête.	17
1.5	Architecture de base des réseaux neuronaux récurrents.	21
1.6	Architecture de base des transformateurs.	22
1.7	Architecture de l'encodeur du transformateur.	23
1.8	Principe de self-attention.	24
1.9	Architecture de base de BERT.	25
1.10	Représentation des entrées et sorties du modèle BERT.	26
2.1	La taxonomie de l'état de l'art des systèmes QA.	30
2.2	Classification des systèmes QA.	34
3.1	Schema global du systeme YouTaQA	38
3.2	Structure XML d'un article Wikipédia	39
3.3	Arborescence des fichiers XML	40
3.4	Schéma représentatif des sections d'un article Wikipédia	41
3.5	La tache de classification avec BERT [2]	43
3.6	Exemple de tokenisation.	43
3.7	La tache de réponse aux questions avec BERT [2].	46
3.8	Exemple de l'entrée et sortie de la tache Question-Answering avec BERT.	46
3.9	Captures d'écran de l'application web YouTaQA	47

TABLE DES FIGURES

4.1	Graphe qui représente les valeurs MRR des résultats	52
4.2	Graphe qui représente les valeurs MAP des résultats	52
4.3	Graphe qui représente les valeurs R-Précision des résultats	53
4.4	Graphe qui représente les valeurs de précision@K des résultats	54
4.5	Graphe qui représente les valeurs du rappel@K des résultats	54
4.6	Graphe qui représente les valeurs du Précision-Rappel des résultats	55

Liste des tableaux

1.1	Matrice de confusion.	26
2.1	Classification des systèmes selon plusieurs axes	35
3.1	Les différents jeux de données disponibles.	44

Liste des abréviations

QAS Question-Answering System

QA Question-Answering

ODQAS Open-Domain Question-Answering Systems

CDQAS Closed-Domain Question-Answering Systems

KB Knowledge Bases

IR Information Retrieval

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). Le système QA 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'informations basiques.

Dès leur début, les majeurs défis des systèmes QA sont la précision, l'habileté à répondre à toutes les questions complexes correctement avec une performance semblable à celle des humains. Dans ce document, notre objectif est de mettre en œuvre un modèle qui garantit des performances comparables à l'état de l'art actuel. Pour avoir une vision plus claire sur les systèmes QA 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 systèmes QA sont généralement classés en trois grandes catégories : les QAS pour le domaine ouvert (ODQAS), les QAS pour le domaine fermé (CDQAS) et les QAS pour le domaine social (SQAS).

Les systèmes QA du domaine fermé sont des systèmes aptes pour répondre à des questions dans un domaine spécifique (médecine, informatique,...etc), ce qui implique un ensemble de questions assez limité et restreint. Nous citons par exemple Joost [3], AQUA [4], Start [5] qui sont des systèmes Question-Answering du domaine fermé. La mise en oeuvre de ce type de systèmes peut être considérée comme une tâche plus facile, car les systèmes peuvent utiliser des connaissances spécifiques à un domaine fréquemment

formalisées dans des ontologies.

Pour les systèmes Question-Answering du domaine ouvert, de tels systèmes permettent de répondre à des questions qui ne sont pas limitées à des domaines prédéfinis, ainsi, les questions peuvent être sur quasiment n'importe quel sujet. Pour arriver à cette fin, le système devrait être capable de passer au crible une très grande quantité de documents textuels pour trouver la réponse, ce qui est une tâche plus complexe et défiante que les QAS du domaine fermé.

En ce qui concerne la source de connaissances¹ des systèmes QA 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 systèmes QA basés sur le texte, les faits, le Web, la recherche d'information (Information Retrieval ou IR) et les règles [6]. Pour notre modèle, nous allons opter pour l'approche de la recherche d'information dans une collecte de données basée sur les articles de Wikipedia² seulement. La recherche d'information, contrairement aux autres approches, sollicite et fait la recherche l'information dans des sources qui ne sont pas forcément structurées ce qui permet une meilleure flexibilité dans le cas d'ajout et d'extension des sources de recherche.

Pour la sélection des réponses, nous allons mettre en œuvre une architecture pour réaliser des modèles de Deep Learning responsables de la classification et l'extraction des réponses à partir des résultats obtenus à l'aide du moteur 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 une réponse directe à une question pour laquelle les requêtes traditionnelles consistant à spécifier des mots-clés ne sont pas très conviviales, c'est donc devenu l'une des fonctions les plus désirables pour les consommateurs d'information.

La majorité des connaissances humaines qui représentent les besoins d'information détaillés d'un utilisateur sont uniquement représentées par le langage naturel. Ils sont accessibles aux humains, qui peuvent

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

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

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 lointain. Pour obtenir des réponses précises, il faut formuler le besoin d'information de manière exacte et bien exprimée, au-delà d'un petit ensemble de termes vagues, comme c'est généralement le cas pour la recherche de documents, car d'une part, les requêtes en langage naturel sont réduites à des recherches basées sur des mots-clés. D'autre part, les bases de connaissances sont interrogées avec des requêtes structurées ou logiques 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 (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, est traditionnellement traitée par le domaine de la réponse aux questions ou Question Answering (QA). Les QAS reposent d'abord sur une analyse syntaxique de la question et une utilisation des techniques de recherche d'informations (RI) pour traiter le texte. Ensuite extraire des passages courts (snippets) 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 QA 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 pour 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 question-reponses exhaustif.

Il existe des systemes QAS qui ont le même but global, mais qui se contentent d'offrir la partie extraction des donnees à leurs utilisateurs en les obligant à fournir les documents nécessaires, notre systeme 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 épargnerais l'utilisateur de fournir autre chose que la question, et eventuellement leur 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 rôle serait de mettre en œuvre une architecture générale d'un QA en utilisant des collections et des ensembles de données de référence sur lesquels nous comptons baser les reponses du systeme

Plan du mémoire

Ce présent manuscrit de thèse de master est composé d'une introduction générale dans laquelle nous avons abordé la problématique et la motivation de ce modeste oeuvre. De plus, nous avons mis en avant notre contribution. Après cela, quatre chapitres principaux se présentent dans ce travail qui sont :

Chapitre 1 : Ce premier chapitre est divisé en trois parties, dans la première nous introduisons la recherche d'informations, le processus d'indexation et de requêtage d'informations. Nous identifions dans la deuxième partie le traitement du langage naturel (NLP) et ses bases. Nous présenterons dans la troisième partie l'aspect théorique du Deep Learning et l'architecture du modèle utilisé BERT.

Chapitre 2 : Ce chapitre dresse un état de l'art sur les systèmes de questions-réponses QAS existants. Le chapitre se terminera par une comparaison de ces systèmes selon plusieurs axes.

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 le dataset 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 discussion.

LISTE DES TABLEAUX

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.

Généralités

1.1 Introduction

Dans ce chapitre, nous fournissons le contexte théorique nécessaire pour comprendre les méthodes discutées dans les chapitres suivants. Tout d'abord, nous présenterons les bases de la recherche d'informations et du traitement du langage naturel NLP, puis nous présenterons les technologies utilisées dans les solutions d'apprentissage en profondeur.

1.2 Recherche d'informations

1.2.1 Définition de la recherche d'informations

La recherche d'informations (IR ou Information Retrieval en anglais) est une discipline en informatique qui permet le traitement des documents contenant du texte libre, afin qu'ils puissent être rapidement retrouvés en se basant sur de mots-clés spécifiés dans la requête d'un utilisateur [7].

1.2.2 L'indexation

L'indexation est le processus de création de l'index de recherche afin de l'utiliser dans processus de recherche d'information et de documents. Il est souvent appelé “processus d'indexation offline” car l'indexation doit être exécutée en “offline”, c'est à dire avant que le système ne soit prêt à traiter les requêtes.

L’index inversé Un index inversé (ou “inverted index” en anglais) est une structure qui facilite la localisation rapide d’éléments intéressants. Il permet d’enregistrer le nombre de fois qu’un terme particulier apparaît dans chaque document. Compte tenu des documents de la collection, et afin de réduire la taille de l’index et permettre des recherches plus larges, les mots peuvent être transformés avant l’indexation.

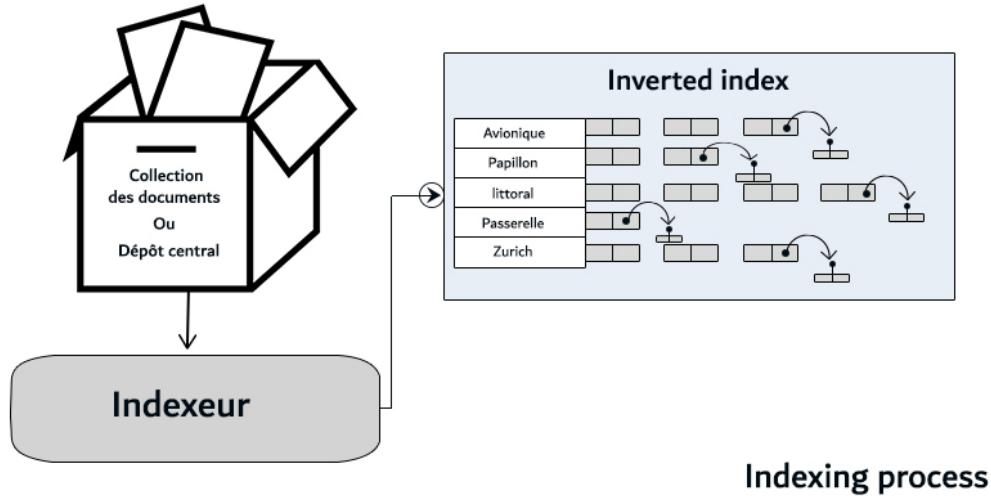


FIGURE 1.1: Les étapes d’indexation

1.2.3 Les étapes d’indexation

1- Transformations des documents

Les mots vides : Les mots vides (ou “stop words” en anglais) sont des mots couramment utilisés qui sont exclus des recherches pour aider à indexer et analyser les pages Web plus rapidement. Quelques exemples de mots vides sont : ”a”, ”et” ”mais” ”comment”, ”ou” et ”quoi”. Bien que la majorité de tous les moteurs de recherche utilisent des mots vides, ils n’empêchent pas un utilisateur de les utiliser, mais ils sont ignorés.

La tokenisation : La tokenisation sert à transformer chaque mot d’un document en un token afin de faciliter son utilisation dans les futures manipulations. En effet, la manipulation des tokens est plus facile que l’utilisation simple des documents.

La normalisation : La normalisation, qui implique la conversion des minuscules et la suppression des variations de la personne ou du temps. Ainsi, ”children” est normalisé en ”child” et ”U.S.A” est normalisé

en “USA”.

Le stemming : Le stemming, qui transforme les mots en supprimant les suffixes ou les préfixes en une forme dite “racine”, qui n’est pas nécessairement un mot réel dans la langue. Ainsi, avec l’algorithme stemming, ”modulation” et ”module” donnent la même racine “modul”.

2- Construction de l’index

Durant cette phase, l’indexeur construit un tableau tel que la première colonne sert à abriter tous les tokens issues de la transformation des mots, la deuxième colonne quant à elle sert à préciser dans quel document le token de la même ligne est apparu. Ensuite, et afin d’optimiser la recherche, les termes dans le tableau sont d’abord triés et organisés selon l’ordre alphabétique.

Enfin, les termes du tableau qui se répètent seront fusionnés tout en précisant les documents dans lesquels ils apparaissent. La figure 1.2 permet de voir à quoi ressemble un index inversé dans sa forme finale.

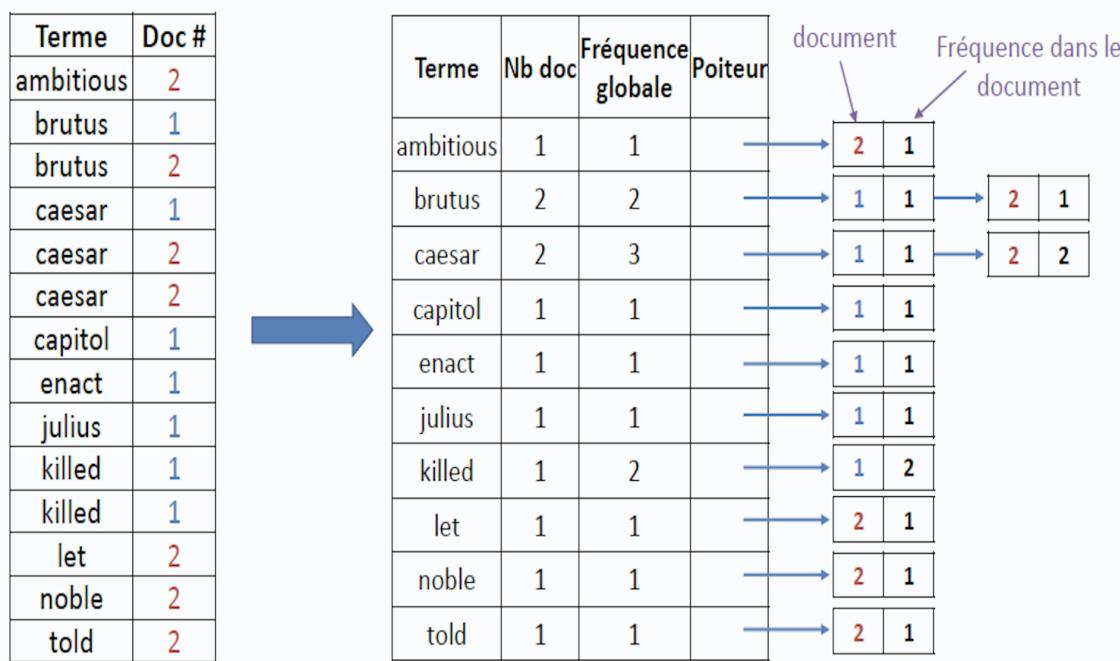


FIGURE 1.2: Architecture de l’index inversé[1].

1.2.4 Requêtage d'informations

Le requêtage d'information consiste à répondre aux requêtes de recherche d'information des utilisateurs du système. Étant donné que la collection de documents est indexée, le processus de recherche peut être lancé. La figure 1.3 permet d'avoir un vue globale sur le processus de requêtage d'information.

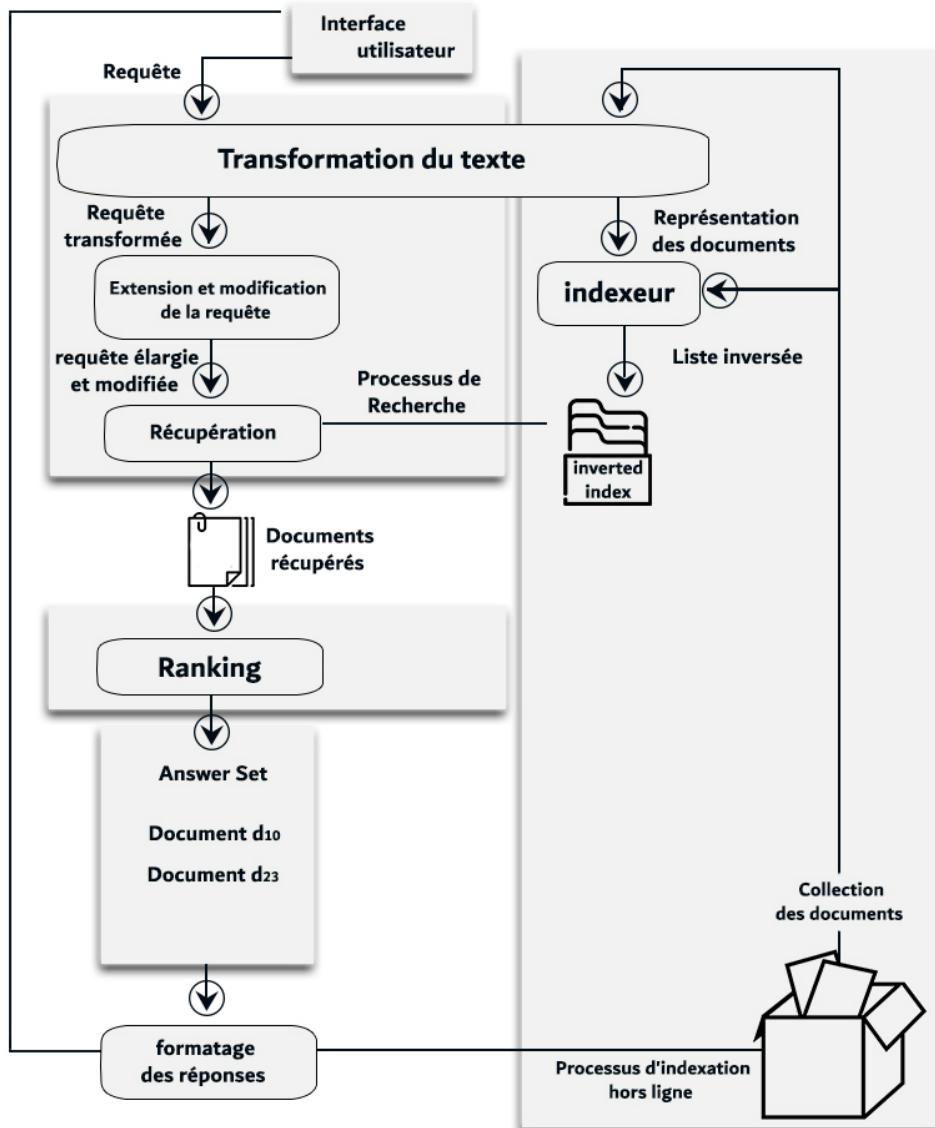


FIGURE 1.3: Processus de Requêtage d'informations

Tout d'abord, l'utilisateur lance une requête dans laquelle il spécifie les informations dont il a besoin. Cette requête est ensuite analysée et modifiée par des opérations qui ressemblent à celles appliquées aux

documents. Les opérations typiques à ce stade consistent à corriger l'orthographe et à éliminer quelques termes tels que les mots vides. Ensuite, la requête étendue et modifiée est traitée pour obtenir l'ensemble des documents récupérés, qui est composé de documents contenant les termes de la requête. Le traitement rapide des requêtes est rendu possible par la structure d'index construite.

Ensuite, les documents retrouvés sont classés en fonction de la probabilité de pertinence pour l'utilisateur. Cette étape est l'une des étapes les plus critiques car la qualité des résultats, telle que perçue par les utilisateurs, dépend fondamentalement du classement. Les documents les mieux classés sont ensuite formatés pour être présentés à l'utilisateur. Le formatage consiste à retrouver le titre des documents et à générer des extraits de ceux-ci, c'est-à-dire des extraits de texte contenant les termes de la requête, qui sont ensuite affichés à l'utilisateur [7].

1.2.5 Classement des résultats

Vector space model VSM : Un modèle d'espace vectoriel est un modèle algébrique qui comporte deux étapes : Dans un premier temps, nous représentons les documents textuels sous forme de vecteurs de mots et, dans un deuxième temps, nous les transformons en format numérique afin de pouvoir appliquer toutes les techniques d'exploration de texte telles que la recherche d'informations, l'extraction d'informations et le filtrage d'informations. Par exemple, pour comparer la pertinences de deux documents par rapport à une requête donnée, nous calculons le cosinus de l'angle theta 1 et de theta 2 dans la figure 1.4 et le document qui présentera la valeur minimale de ces deux résultats sera pris en compte.

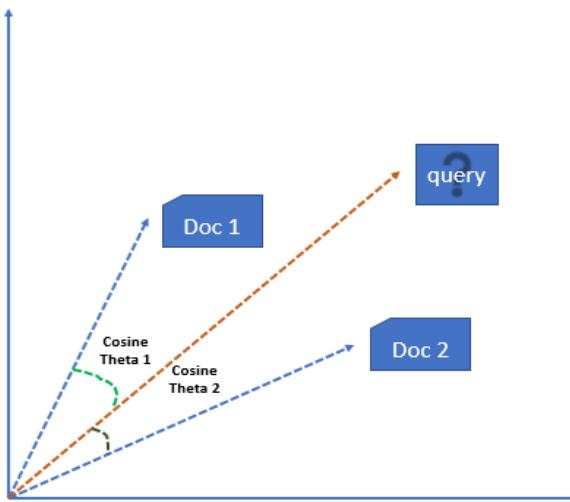


FIGURE 1.4: Similarité VSM entre deux documents et une requête.

Jaccard Similarity Coefficient : La mesure de similarité définit la similitude entre deux ou plusieurs documents. Le coefficient de similarité Jaccard mesure le degré de similarité entre les documents récupérés [8].

L'indice de Jaccard est calculé par le rapport entre le cardinal de l'intersection des ensembles considérés et le cardinal de l'union des ensembles comme suit :

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

où A et B sont deux ensembles (documents dans le cas de la recherche d'information).

Term frequency : La fréquence du terme (Term Frequency ou TF en anglais), qui est souvent utilisée dans la recherche d'informations, indique la fréquence à laquelle le terme apparaît dans le document [9]. Il est calculé suivant la formule suivante :

$$TF(i, j) = \frac{\text{Fréquence du terme } i \text{ dans le document } j}{|\text{Document}_j|}$$

TF-IDF weighting : TF-IDF est l'abréviation de "Frequency-Inverse Document Frequency", et le poids TF-IDF est un poids souvent utilisé dans la recherche d'informations. Les variations du schéma de pondération TF-IDF sont souvent utilisées par les moteurs de recherche comme outil central pour noter et classer la pertinence d'un document en fonction d'une requête de l'utilisateur.

$$tf.idf_{t,d} = (1 + \log(tf_{t,d})) \log_{10}\left(\frac{N}{df_t}\right)$$

où :

- t : Fait référence au terme en question.
- d : Réfère au document en cours d'évaluation.

Ainsi, de façon globale, pour une requête donnée q , le score d'un document sera :

$$\text{Score}_{TF-IDF}(q, d) = \sum_{t \in q \cap d} tf.idf_{t,d}$$

BM-25 weighting : 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 [10]. Le score BM-25 est calculé comme suit :

$$\text{Score}_{BM-25}(D, Q) = \sum_{i=1}^n IDF(q_i) \frac{f(q_i, D)(k_1 + 1)}{f(q_i, D) + k_1(1 - b + b \frac{|D|}{avgdl})}$$

où :

- $f(q_i, D)$: La fréquence du terme q_i dans le $document_D$.
- $|D|$: Le nombre de mots total 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 $b = 0.75$).
- $IDF(q_i)$: La fréquence inverse du document pondérant le terme q_i de la requête.

1.2.6 Les outils d'évaluation

TREC Eval

Le Text REtrieval Conference (TREC) 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 [11].

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 IR.

1.2.7 Les outils de développement

Lucene

Apache Lucene 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 multiplateforme. Apache fournit aussi une interface python de Lucene nommée PyLucene.

1.3 Natural Language Processing

1.3.1 Définition du traitement du langage naturel

Le traitement du langage naturel (Natural language processing ou NLP en anglais) est le processus de compréhension automatique du langage humain. Dans l'intelligence artificielle, la 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 [12].

1.3.2 Word Embedding

L'encapsulation des mots (ou Word embedding en anglais) a été inventée pour résoudre les problèmes du modèle du sac de mots (BOW ou Bag of words en anglais) car ce dernier ne tient compte ni de la sémantique ni de l'ordre des mots. Bien que la méthode n-gram puisse prendre en compte l'ordre des mots dans une certaine mesure, elle n'est pas une solution idéale compte tenu de l'efficacité et des performances [13]. Des termes tels que "aéronef", "avion", "vol" sont utilisés dans le même sens, mais le modèle BOW les considère différemment. Le word embedding permet de faire correspondre chaque terme du vocabulaire à un espace vectoriel à dimension fixe pour les utiliser comme caractéristiques. Word2Vec, GloVe et FastText sont les modèles les plus utilisés pour les tâches de word embedding.

1.4 Deep Learning

L'apprentissage profond (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.

Les RNN (Réseaux de neurones récurrents) 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. Ils ont connu un grand succès dans l'étiquetage des séquences et les tâches de prédiction telles que la reconnaissance de l'écriture manuscrite et la modélisation du langage. Ils contiennent des connexions cycliques qui les rendent un outil puissant [14].

Les RNN traitent les jetons de manière séquentiel, où après l'état est stocké dans un vecteur après chaque jeton, pour qu'il soit ensuite combiné avec tous les vecteurs qui précèdent afin de réaliser les traitements nécessaires, comme le montre la figure 1.5.

Les informations des jetons passent donc par tout le chemin pour se retrouver en fin de séquence. Mais d'un point de vue pratique, ce mécanisme reste imparfait et inefficace, dû au risque de «Gradient vanishing and exploding problems» ou perte d'informations, c-à-dire lors de la mise à jour des poids du réseau (proportionnelle à la dérivée partielle de la fonction d'erreur par rapport au poids actuel dans

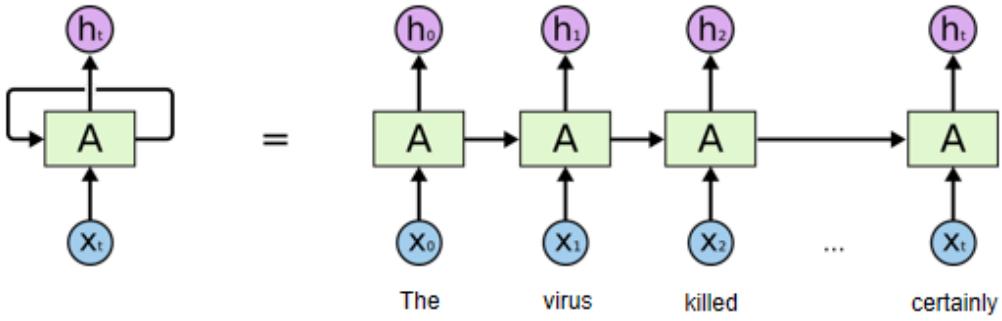


FIGURE 1.5: Architecture de base des réseaux neuronaux récurrents.

chaque itération), dans le cas où le gradient est très faible, ceci empêchera le poids de modifier sa valeur voire empêcher le réseau de poursuivre son entraînement et de perdre l’information pertinente [15].

À ce stade, le mécanisme d’attention est apparu, qui permet d’examiner la phrase avec prise en compte de tous les états précédents et les pondérer en fonction d’une mesure apprise de la pertinence du jeton actuel, fournissant ainsi des informations plus précises sur les jetons pertinents lointains.

1.4.1 Attention

Le mécanisme d’attention peut être expliqué par l’exemple des langues dans lesquelles l’ordre n’a pas vraiment d’importance comme la langue polonaise. Dans ce genre de langues, nous pouvons changer l’ordre des mots en fonction de ce que nous voulons mettre en valeur. De même pour 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. 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 [16].

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.

1.4.2 Les Transformateurs

Le Transformateur est un modèle de DL utilisé principalement dans le domaine du NLP. Comme les RNN, 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 [16].

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 [17].

Les transformateurs sont 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.

Les transformateurs reposent sur une architecture encoder-decoder (figure 1.6). Le composant d'encodage et de décodage sont des piles du même nombre.

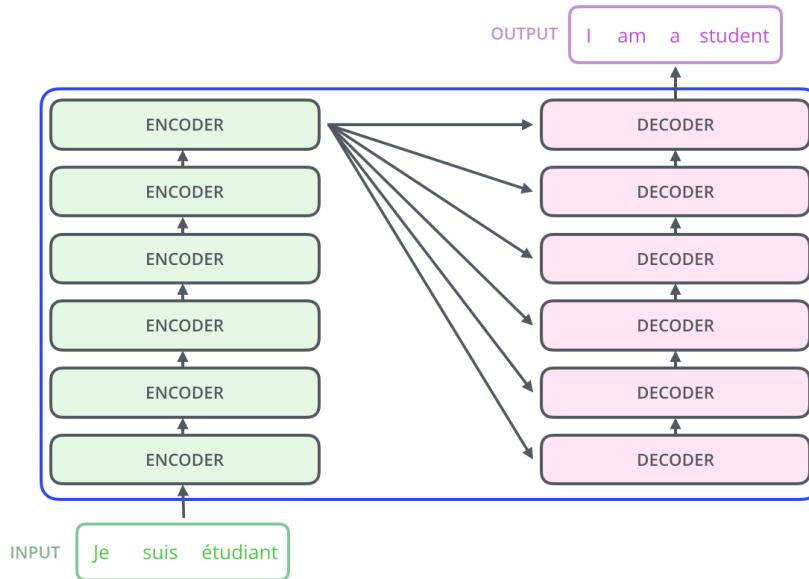


FIGURE 1.6: Architecture de base des transformateurs.

L'encodeur est composé de 6 couches identiques. Chaque couche a deux sous-couches. La première est celle de «Multi-head Self-Attention»[18] qui permet au modèle de s'occuper des informations provenant de différents positions. La seconde est un simple réseau «feed-forward»[] entièrement connecté en fonction

de la position.

Le décodeur à son tour, composé de 6 couches identiques, et 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 le output de l'encodeur. Cette couche aide le décodeur à se concentrer sur les parties pertinentes de la phrase d'entrée.

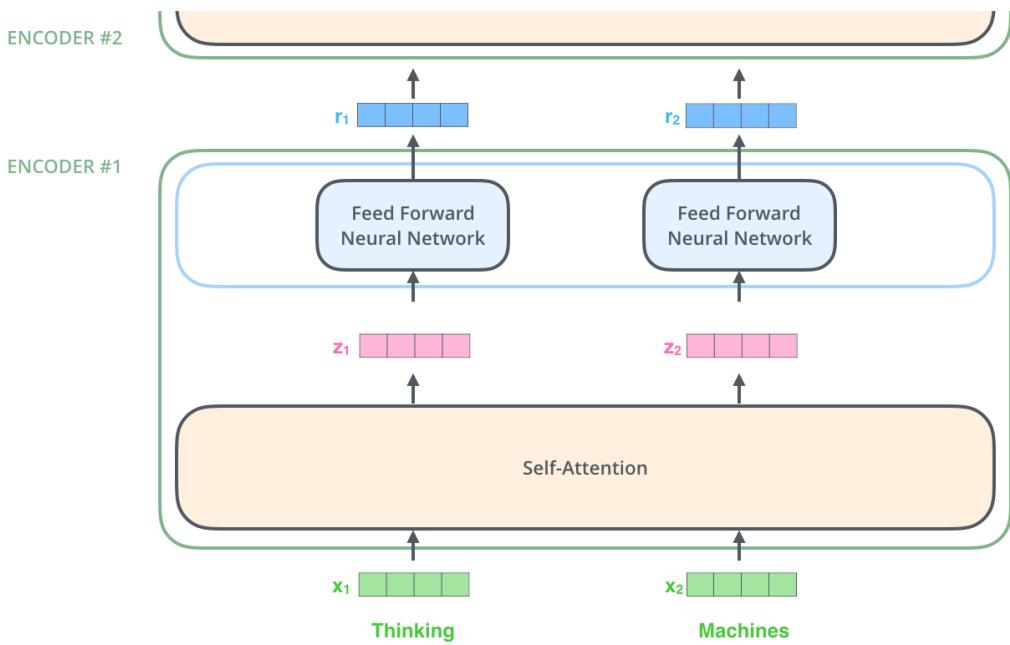


FIGURE 1.7: Architecture de l'encodeur du transformateur.

En phase d'encodage, comme le montre la figure 1.7 :

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 chaque couche, la self-attention est calculée 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 to each others afin d'avoir le sens exact de l'input comme le montre la figure 1.8.

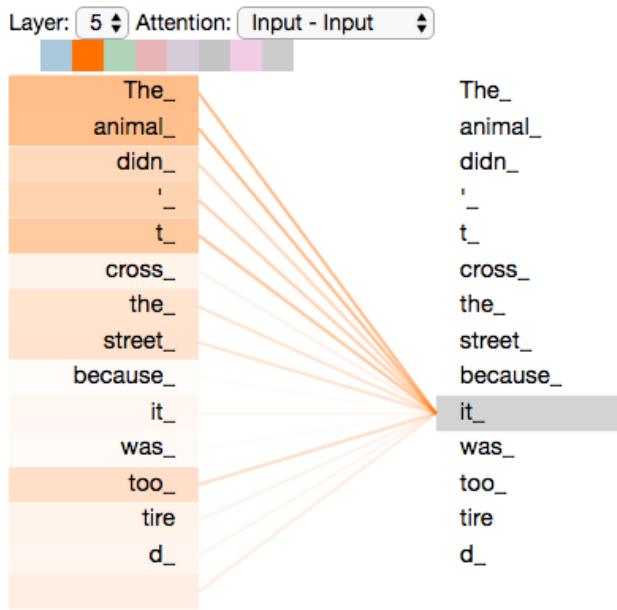


FIGURE 1.8: Principe de self-attention.

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 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) [2] et GPT (Generative Pre-trained Transformer) [19], qui ont été formés avec d'énormes ensembles de données en langage général et peuvent être adaptés à des tâches linguistiques spécifiques.

1.4.3 BERT (Bidirectional Encoder Representations from Transformers)

Les transformateurs est une architecture d'encodeur-décodeur, c-à-dire ils prennent un input et produisent un output. BERT est une architecture qui n'utilise que la partie d'encodeur des transformateurs pour réaliser de multiples tâches. On appelle ça : Transfer Learning.

Transfer Learning : 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 utilisé en Deep Learning avec les modèles pre-trained comme BERT, ils sont conçus pour réaliser une des taches spécifiques (de traitement d’images, traitement de langage naturel...) avec changement de l’input suivant la tâche à réaliser [20].

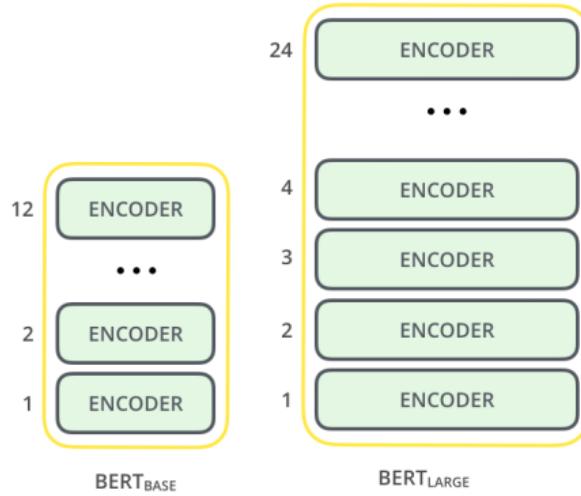


FIGURE 1.9: Architecture de base de BERT.

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** : contient 24 couches, hidden size=1024 et 16 self-attention heads.

«Uncased» réalise une mise en minuscules avant la tokenisation (eq. 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 une méthode 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 1.10, 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].

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

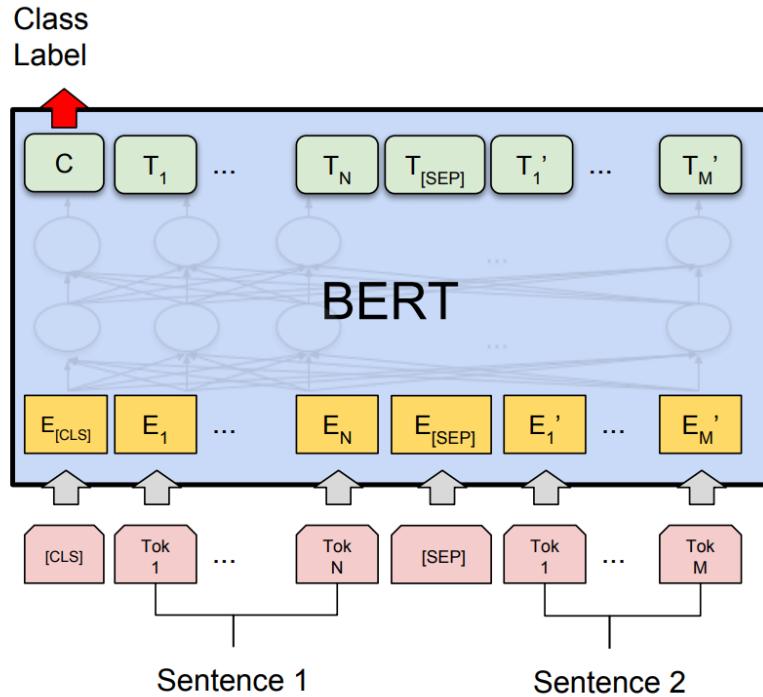


FIGURE 1.10: Représentation des entrées et sorties du modèle BERT.

1.5 Les métriques d'évaluation

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 1.1: Matrice de confusion.

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

- **TP** : Nombre d'échantillons correctement prédit appartenant à la catégorie "Oui".
- **FP** : Nombre d'échantillons dans la catégorie "Oui" qui n'ont pas été correctement prédits.
- **TN** : Nombre d'échantillons de la catégorie "Non" correctement prédits.

— **FN** : Nombre d'échantillons de la catégorie “Non” 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 [21]. 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.

$$Precision = \frac{TP}{TP + FP}$$

$$Rappel = \frac{TP}{TP + FN}$$

F1 score : Le score F, également appelé *scoreF1*, 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.

$$F1 = 2 \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

R-précision : La R-précision est une métrique qui exige de connaître tous les documents qui sont pertinents pour une requête. Le nombre de documents pertinents, R , est utilisé comme limite pour le calcul, et cela varie d'une requête à l'autre [22]. Par exemple, s'il y a 10 documents pertinents pour la requête "Algeria" dans un corpus ($R=10$), la précision R pour "Algeria" examine les 10 premiers documents renvoyés, compte le nombre de documents pertinents et transforme ce nombre en fraction de pertinence :

$$\frac{r}{R} = \frac{r}{10}$$

Mean average precision : La précision moyenne (MAP) ou parfois simplement appelée AP est une mesure populaire utilisée pour mesurer la performance des modèles de recherche d'informations [23]. La précision moyenne d'un ensemble de requêtes est définie par :

$$MAP = \frac{\sum_{q=1}^Q \text{AveP}(q)}{Q}$$

où Q est le nombre de requêtes dans l'ensemble et $\text{AveP}(q)$ est la précision moyenne (AP) pour une requête donnée, 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 [24]. Cette métrique est calculée sur la base de la formule suivante :

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

où rank_i fait référence à la position du premier document pertinent pour la i -ième requête et $|Q|$ est le nombre de requêtes total.

1.6 Conclusion

Dans ce chapitre, nous avons abordé la théorie de base du domaine de recherche d'informations 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 exhaustive et claire du traitement du langage naturel et du deep learning afin d'introduire le lecteur pour le prochain chapitre qui présentera le Deep learning.

Chapitre 2

Etat de l'art

2.1 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 caractériser, d'évaluer et de classifier ces systèmes afin de connaître l'état de l'art de ce domaine.

2.2 Classification selon le domaine d'application

Dans le domaine des QAS, nous faisons souvent référence à la classification de ces derniers en fonction du domaine dans lequel ils opèrent et répondent aux questions de ce dernier. Comme le montre la taxonomie illustrées dans la figure 2.1. Les QAS peuvent être classés selon leur domaine d'application. Il existe trois classe qui sont : Closed-domain QAS, Open-domain QAS et Social QAS.

Dans cette section, nous allons discuter les caractéristiques de ces classes en fournissant des exemples des QAS selon certains critères afin de comparer et extraire les avantages et les limites de chaque domaine d'application.

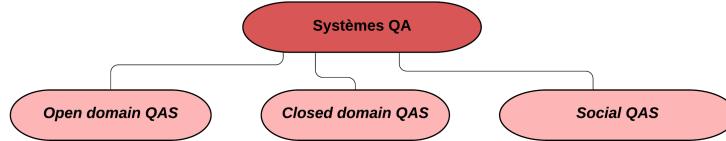


FIGURE 2.1: La taxonomie de l'état de l'art des systèmes QA.

2.3 Closed domain QA Systems

Les systèmes QAS du domaine fermé ou (domaine restreint) 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 [25]. Ce domaine nécessite une disposition linguistique pour comprendre le texte en langue naturelle afin d'apporter une réponse précise aux requêtes [26].

Les premiers QAS créés sont LUNAR [27] en 1961 et BASEBALL [28] en 1972 qui ont été les premiers exemples d'interfaces en langage naturel avec des bases de données (NLIDB), i.e. interroger une base de données structurée à l'aide de questions en langage naturel [29]. LUNAR répond aux questions en langage naturel sur l'analyse géologique des roches renvoyées par les missions lunaires Apollo. Il est capable de répondre à 90% des questions dans son domaine posées par des utilisateurs non formés au système. Tandis que BASEBALL, répond aux questions sur les dates, les lieux et les résultats des matchs de baseball. Ces premiers systèmes encodaient de grandes quantités de connaissances spécifique au domaine dans des bases de données. Les QAS du domaine restreint modernes sont moins dépendants des grandes KB et ne se concentrent pas dans la compréhension des langues comme le faisaient les anciens systèmes. Nous citons dans le domaine de la médecine, le premier système nommé MYCIN [30] qui explique les concepts médicaux. MedQA [31] qui a pour objectif de répondre à tous les types de questions médicales. Ce système intègre des techniques de recherche, d'extraction et de résumé des informations afin de générer automatiquement un texte au niveau des paragraphes pour les questions de définition (eg : "Qu'est-ce que X ?"). Toujours dans le domaine de la médecine, HONqa [32], EAGLi¹ et askHERMES [33] qui ont

1. eagl.unige.ch/EAGLi

utilisé MEDLINE² comme source majeure de réponses. HONqa³ est un système multilingue (anglais, français et italien) qui a été conçu. Il utilise des sites web de santé certifiés qui permettent d'orienter les informations vers des personnes ayant des niveaux différents de connaissances en matière de santé [34]. Tandis que askHERMES⁴ réponds à tous les types de questions. Les réponses sont présentées de trois façons : réponses regroupées par termes, liste de réponses simples classées et réponses regroupées par contenu [34]. D'autre part, loin du domaine médical, le site web KAAS [35] qui a été développé pour être utilisé dans un environnement d'apprentissage (Advanced Interactive Environnement de découverte pour l'ingénierie Education ou AIDE) a pour but de répondre aux questions des étudiants de premier cycle de deux universités spécialisées dans l'aéronautique l'ingénierie. WEBCOOP [36] qui a été conçu et développé pour le domaine de tourisme. Il utilise une représentation logique des données et facilite leur interrogation avec des requêtes en langage naturel. De même, Diekema et al. [] qui ont construit un QAS pour la recherche d'information dans le domaine de l'ingénierie aérospatiale.

Malgré la précision que ces systèmes ont atteinte 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. Mais les QAS du domaine fermé peuvent être combinés pour créer des QAS de domaine ouvert (Indurkhya et al., 2010 ; Lopez et al., 2011) [26].

2.3.1 Open-domain QA Systems

2.3.2 Social QA Systems

Malgré la précision et la complétude que les QAS ont atteint de nos jours, on se trouve toujours face à plusieurs problèmes [37], dont la mauvaise expression de la question et l'absence des réponses pertinentes. Pour cela, un nouveau type de QAS est apparu qui est le Social QAS. Il met en relation les utilisateurs afin collecter le contenu nécessaire pour répondre aux questions. Ce type de QAS permet de fournir un moyen pour répondre à plusieurs types de questions telles que la recommandation (eg. Recherche d'opinions, connaissance factuelle, résolution de problèmes...) en prenant en compte la fréquence de ces questions et les motivations des utilisateurs qui interrogent leur réseau social plutôt que d'utiliser un moteur de recherche traditionnel [38]. D'une part, social QAS incluent un composant de réseautage social (social networking), par lequel les utilisateurs peuvent créer des liens avec d'autres utilisateurs en posant et en

2. <https://www.ncbi.nlm.nih.gov/bsd/medline.html>

3. services.hon.ch/cgi-bin/QA10/qa.pl

4. <https://www.askhermes.org/>

répondant à des questions et en établissant une réputation dans la communauté pour leurs contributions de contenu. D'autre part, ils offrent une vaste base de connaissances évolutive de questions et réponses. Ils constituent une ressource précieuse pour les utilisateurs ayant des besoins d'informations similaires [39].

Le concept des QAS social est apparu dans les débuts des années 2000, lorsque The AnswerBank⁵ a été réalisé qui représente une plateforme où les utilisateurs posent des questions et obtiennent de vraies réponses de vraies personnes et donnent quelques réponses en retour. Ensuite, en 2001, Wikipedia⁶ a créé un nouveau service presque similaire nommé Wikipedia Reference Desk⁷. Dans ce QAS, les utilisateurs posent leurs questions, et des volontaires chez Wikipedia offrent des solutions/réponses. En 2002, des anciens employés chez Samsung (Jeon et al.) ont développé Naver's KiN [40] qui a été lancé la première fois en Corée du Sud par NHN Corporation⁸. Contrairement aux autres QAS Social, sur NKIN les utilisateurs peuvent non seulement poser et répondre à plusieurs questions mais aussi communiquer entre eux via email ou messages. Un an après, la concurrence a pris de l'ampleur, trois social QAS ont pris place sur internet ; AnswerBag⁹, FunAdvice¹⁰ et Ask Me Help Desk¹¹. AnswerBag créé par Joel Downs, donne la possibilité à ses utilisateurs de poser et répondre à des questions dans plus de 4000 catégories. Il ne contient pas la fonctionnalité de sélection de la meilleure réponse mais comporte un système de notation qui permet de trier les réponses selon leurs notes [41]. Sur AnswerBag, on peut attacher une image ou une vidéo avec la question/réponse. En 2005, deux systèmes sont apparus Answers.com¹² et Yahoo! Answers¹³. Ce dernier autorise les utilisateurs à publier une question en détail avec le titre de la question dans une catégorie relative à laquelle la question appartient. D'autres bénévoles de la communauté sont autorisés à répondre. L'utilisateur qui pose la question a la possibilité de sélectionner la meilleure réponse qui a résolu son problème ou a répondu à sa question [42]. En 2006 et 2007, construire un QAS Social est devenu l'intention de tous les chercheurs. On cite ; Blurtit¹⁴, AOL Answers¹⁵, Chacha¹⁶, Fluther¹⁷,

5. <https://www.theanswerbank.co.uk/>

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

7. https://en.wikipedia.org/wiki/Wikipedia:Reference_desk

8. <https://www.nhn.com/>

9. <https://www.answerbag.com/>

10. funadvice.com

11. <https://www.askmehelpdesk.com/>

12. <https://www.answers.com/>

13. <https://answers.yahoo.com/>

14. <https://www.blurtit.com/>

15. <https://www.aol.com/>

16. <http://www.chacha.com/>

17. <https://www.fluther.com/>

LinkedInAnswers, Wiki Answers¹⁸ et AskVille¹⁹. Celui qui a marqué le plus de succès pendant ces deux années est le Wiki Answers. Il a été lancé autant que FAQ Farm en 2002 par Chris Whitten ensuite acheté par Answer Corporation en Novembre 2006. Wiki Answers permet aux utilisateurs de modifier les question et réponses existantes pour améliorer la qualité du contenu. Les questions redondantes sont identifiées, fusionnées et généralisées. Wiki Answers contient plus de 7,000 categories []. Avant la fin de la première décennie, la première version de StackOverflow²⁰ créée par Jeff Atwood et Joel Spolsky a été lancée ; un site de QAS pour les programmeurs professionnels et passionnés. Il propose des questions et réponses sur un large éventail de sujets en programmation informatique. Durant la deuxième décennie, pleins de sites web qui tentent à leur tour contribuer à ce domaine nous citons Ask.fm²¹, Quora²², Zhihu.com²³, Aardvark²⁴, Brilliant.org²⁵ et Spring.me lancé en 2006 sous le nom de formspring²⁶.

Les systèmes cités dans cette section ne sont pas assez complets par rapport au temps de réponse et à la qualité des réponses. Il est donc important de classer correctement des réponses dans les systèmes QA. Pour cela, Dalip et Al. [43] a proposé une méthode d'apprentissage qui consiste à l'utilisation des caractéristiques spécifiques du domaine des questions-réponses. Cette approche a donné des résultats extraordinaires et a produit un nouvel état de l'art.

2.4 Classification selon d'autres critères

A l'origine, on désignait par un système Question-Answering la recherche de réponses dans des assemblages de documents non structurées. L'évolution de la quantité des données ont poussé de nombreuses nouveautés récentes à apparaître telles que : WebQuestions [44] et SimpleQuestions [45] fondées sur le KB Freebase [46], ou sur des bases de connaissances (Knowledge bases ou KB) extraits automatiquement, par exemple, OpenIE triples et NELL [47]. Néanmoins, les bases de connaissances KB exposent des limites attachées (incomplétude, schémas fixes) qui ont fait appel aux chercheurs pour revenir au cadre original de réponses à partir du texte brut.

-
- 18. <https://wiki-answers.com/>
 - 19. <https://askville.com.cutestat.com/>
 - 20. <https://stackoverflow.com/>
 - 21. <https://ask.fm/>
 - 22. <https://quora.com/>
 - 23. <https://www.zhihu.com/>
 - 24. [https://en.wikipedia.org/wiki/Aardvark\(searchengine\)](https://en.wikipedia.org/wiki/Aardvark(searchengine))
 - 25. <https://brilliant.org/>
 - 26. <https://fr.wikipedia.org/wiki/Spring.me>

Une deuxième motivation qui avait pour but d'attirer le regard à nouveau sur ce problème est celle de la compréhension automatique du texte, autrement dit répondre à des questions après avoir lu un court texte ou une histoire. Ce sous-domaine a progressé considérablement dernièrement grâce à de nouvelles structures d'apprentissage approfondi comme les réseaux de neurones basés sur l'attention (Attention mechanism²⁷) et la mémoire augmentée [48] et à la publication de nouveaux jeux de données d'entraînement et d'évaluation comme QuizBowl [49], CNN/DailyMail [50] basé sur des articles de presse , CBT [51] basé sur des livres pour enfants , ou SQuAD [52] et WikiReading [53], tous deux basés sur Wikipédia.

Aujourd'hui, les systèmes QA sont de plus en plus nombreux et qui offrent une diversité imposante. En conséquence, la classification par domaine d'application n'est plus suffisante pour comparer et définir toutes les caractéristiques des systèmes QA. Pour cela, nous avons établi d'autres critères de classification (classification par type de question, source de données et par source de connaissance) qui permettent de mettre en valeur les différences de ces systèmes et extraire les avantages et les limites de chacun d'eux.

Dans la figure 2.2, nous illustrons les différents axes de classification selon lesquels nous allons classer et comparer les systèmes QA existants.

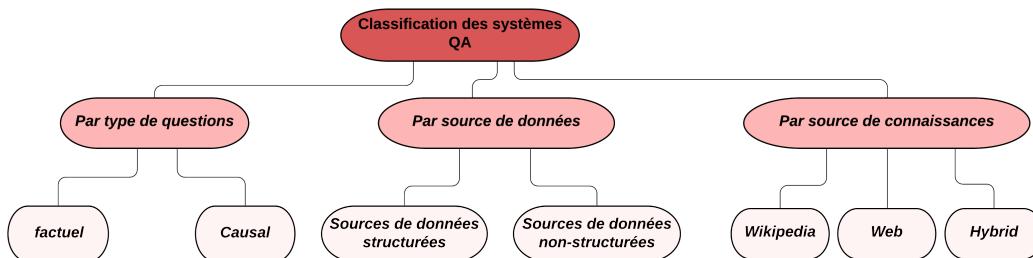


FIGURE 2.2: Classification des systèmes QA.

En outre, le tableau 1 résume les différents axes selon lesquels nous allons classer les systèmes QA tout en expliquant ces axes et des exemples de chaque catégorie de système QA.

Comme nous pouvons le voir dans le tableau 2.1, les systèmes QA peuvent être vus et classés selon plusieurs critères. Parmi ces critères, nous trouvons tout d'abord le domaine d'application du système qui peut être un domaine de type ouvert ou de type fermé. Une autre vision de la classification nous a permis

27. Le mécanisme d'attention : Un mécanisme qui permet au modèle de se concentrer et d'accorder plus d'attention aux parties pertinentes de la séquence d'entrée selon les besoins.

Système	Domaine d'application			Type de question				Source de données	Type de réponse
	Ouvert	Fermé	Social	Factuelle	Causale	Confirmation	Liste		
BASEBALL	X			X		X		X	X
LUNAR	X			X			X X	X	X
MedQA	X						X	X	X
MYCIN	X								
HONqa	X			X			X X X	X	
EAGLi	X						X	X	X
askHERMES	X			X X X X X X X				X	
KAAS	X			X					
WEBCOOP	X				X X		X X		
START	X								
QuALIM	X								
HyperQA	X								
DeepQA	X								
BiDAF	X								
BERT	X								
ALBERT	X								
DrQA	X								
QAnet	X								
TANDA	X								
XLNet	X								

TABLE 2.1: Classification des systèmes selon plusieurs axes

de classer les systèmes selon le type de questions prises en charge par ces derniers. Aussi, nous avons utilisé une autre approche de classification qui est la source de connaissances. Cette classification permet de classer les systèmes selon leurs sources sur lesquelles ils se basent pour extraire les informations afin de répondre à une question. Dans notre étude bibliographique, nous avons constaté différentes approches de recherche d'une réponse, ce qui nous a incité à classer les systèmes selon différentes approches de recherche. Enfin, chaque système fournit une réponse d'une façon différente. Pour cela, nous avons établis une classification selon les types de réponses de ces systèmes.

2.5 Conclusion

L'objectif de ce chapitre était de donner un aperçu global sur les systèmes réalisés jusqu'à présent dans le domaine du Question-Answering, ainsi que les problèmes et les faiblesses de ce que les chercheurs ont réussi à produire de mieux jusque là. De plus, nous avons vu les différentes approches appliquées lors de l'extraction des réponses ainsi que les diverses sources de connaissances des systèmes QA. Par ailleurs, nous avons vu les différents axes de classification selon lesquels nous pouvons classer et évaluer un système QA.

Présentation de la solution

3.1 Introduction

A présent, nous allons détailler l'architecture de notre système YouTaQA et décrire ses différents composants. 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.

3.2 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 personnes qui souhaitent trouver une information ou une donnée saisissent une requête sur ce qu'elles aimeraient trouver et le moteur fournit le contenu qui correspond à ce qu'elles veulent.

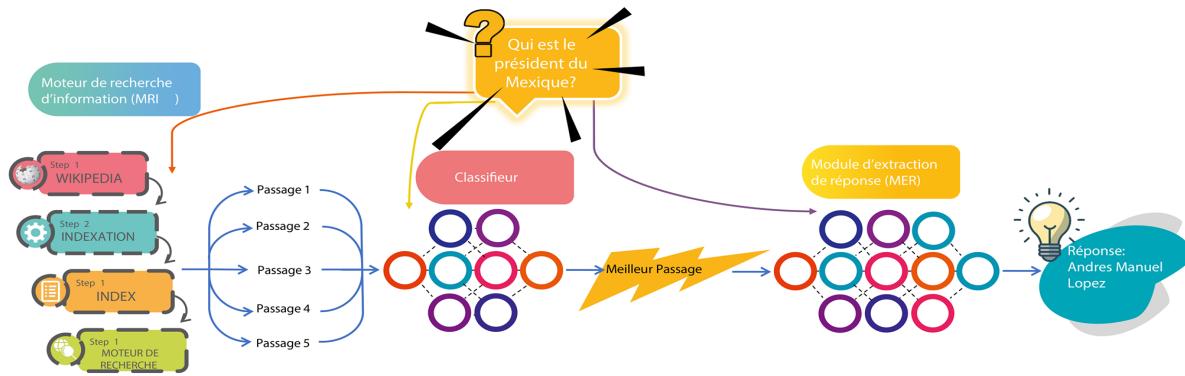


FIGURE 3.1: Schema global du système YouTaQA

Pour notre moteur de recherche, nous avons suivie la même philosophie. Nous nous sommes basé sur la collection exhaustive d'articles fournie par Wikipédia.

3.2.1 Choix de la base des documents

En vue d'établir un moteur de recherche, nous avons pris la peine de chercher et comparer les différentes bases de documents fournie par les grands géants du web (Google, FaceBook, Microsoft, etc) et notre choix ultime est tombé sur la base de wikipedia qui offre l'intégralité de ses articles en plusieurs langues parmi eux l'anglais. Cette base de documents est disponible en ligne gratuitement en format XML¹ et qui comprend plus de 6.1 millions d'articles.

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

3.2.2 Prétraitement 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.

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

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

3. **WikiExtractor** :<https://github.com/attardi/wikiextractor>

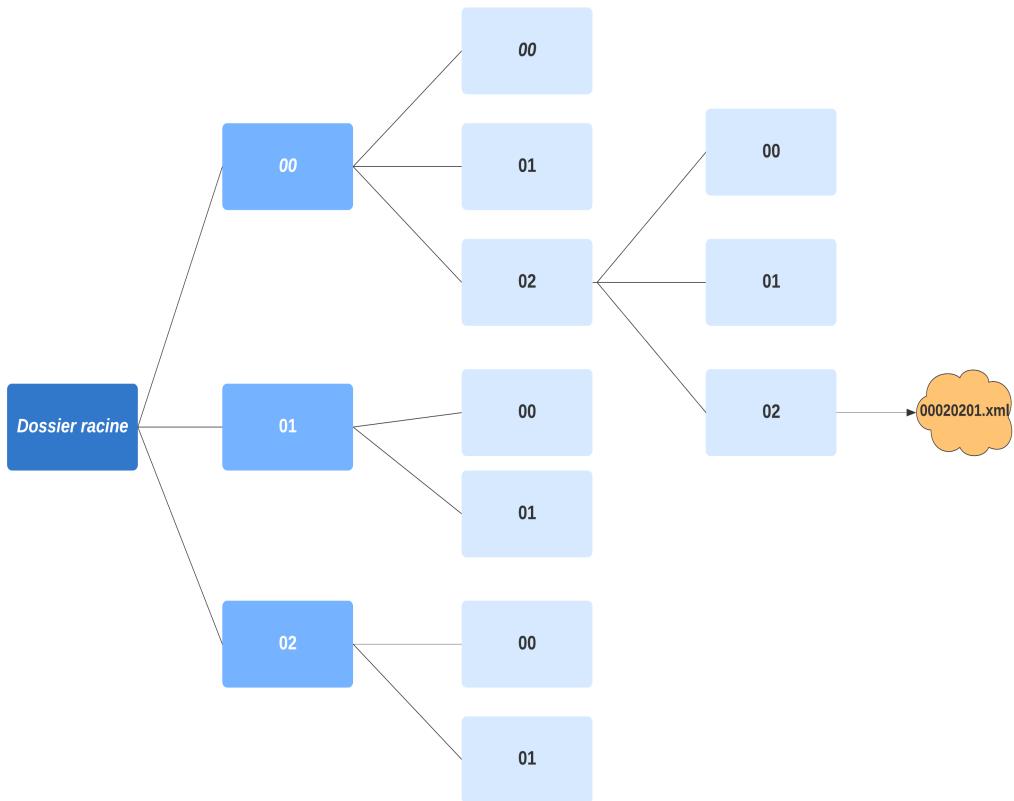


FIGURE 3.3: Arborescence des fichiers XML

permis d'avoir des articles en texte brut sans la mise en page de Wikipédia.

3.2.3 Indexation des articles

Après avoir traité les articles de Wikipédia, nous nous sommes engagés dans l'indexation de ces articles en utilisant la bibliothèque **PyLucene**.

Dans cette étape, nous avons procédé à implémenter 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 stemming des mots de chaque article.

Pour but de faciliter la tâche de notre future module d'extraction des réponses, et afin de booster les performances de notre système en terme de temps d'exécution, au lieu de considérer les articles en eux

CHAPITRE 3. PRÉSENTATION DE LA SOLUTION

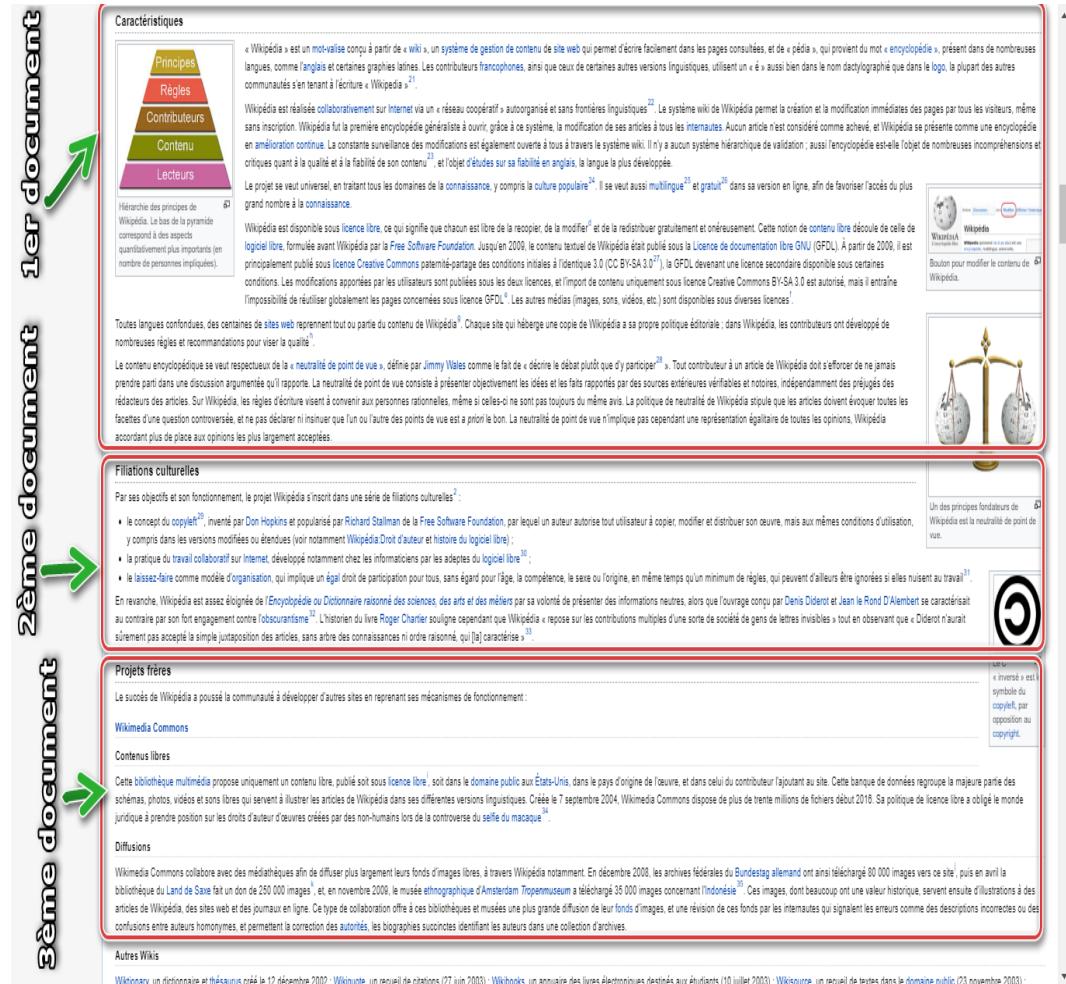


FIGURE 3.4: Schéma représentatif des sections d'un article Wikipédia

mêmes comme des documents, nous avons eu l'idée de considérer les sections des articles comme document 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 Wikipedia et comment l'index va considérer cet article en divisant le même article en trois documents.

Notre index est constitué de 5 champs essentiels :

- L'identifiant de l'article qui est unique à chaque article.
- Le titre de l'article.
- L'identifiant de la section qui est considéré comme l'identifiant du document par lucene.
- Le titre de la section.
- Le contenu de la section, c'est le contenu principal de chaque document.

3.2.4 Méthodes de recherche adoptées

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

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.

Méthode de recherche MultifieldsSearch

Cette méthode consiste, comme la première méthode, à extraire les mots clés, la seule différence consiste à faire une recherche des mots clés non pas seulement sur le contenu du document mais aussi en prenant en compte son titre.

3.3 Module de classification

3.3.1 Choix du jeu de données

La deuxième étape de notre solution est la classification des résultats obtenus dans la partie précédente. Afin d'entrainer notre modèle durant cette partie nous avons choisi d'utiliser le jeu de données QNLI (Question-answering Natural Language Inference) proposé par GLUE [54]. QNLI est un ensemble de données constitué de paires question-paragraphe, où l'une des phrases du paragraphe (tiré de Wikipedia) contient la réponse à la question correspondante. La tâche consiste à déterminer si la phrase de contexte contient la réponse à la question.

3.3.2 Entrainement du classifieur

BERT est un modèle pré-entraîné basé sur le Transfer Learning capable de réaliser plusieurs tâches NLP. Parmi ces tâches NLP, nous retrouvons la classification de texte.

Comme le montre la figure 3.5, le classifieur prend comme entrée la question et le paragraphe et produit comme sortie une class de label (1 si le paragraphe contient la réponse ; 0 sinon).

Comme première étape, nous avons commencé par la tokenisation de notre jeu de données pour qu'il

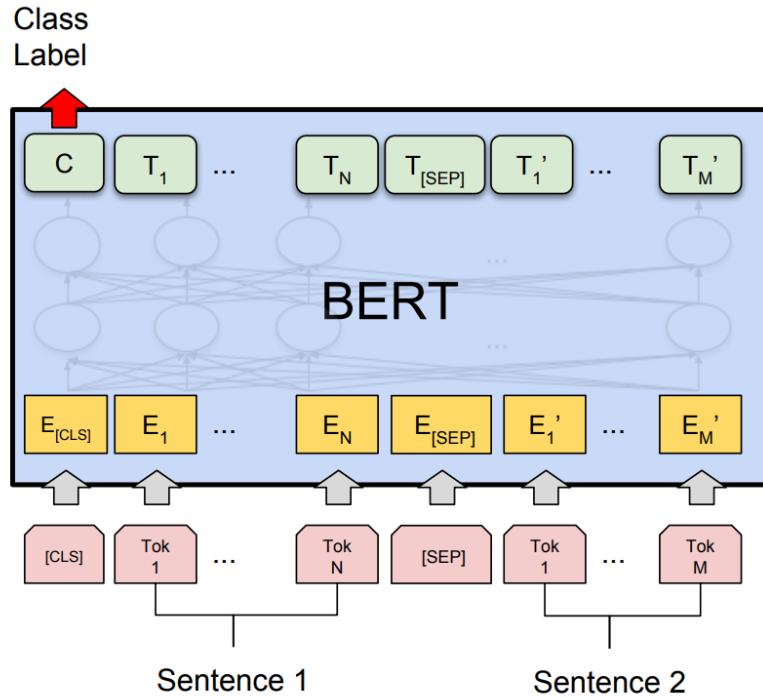


FIGURE 3.5: La tache de classification avec BERT [2]

denvient codé suivant un seul format. La fonction de tokenisation parcours le dataset et concatène chaque paire de Question - Paragraphe. Comme nous l'avons expliqué précédemment, chaque paire commence par le jeton spécial [CLS] et un jeton [SEP] qui sépare de deux parties de l'entrée. Afin que tous les tokens soient de la même taille, nous avons définit une taille maximale, complété avec le mot clé [PAD] comme le montre l'exemple de la figure 3.6.

FIGURE 3.6: Exemple de tokenisation.

Le classifieur retourne en sortie un tableau de probabilités pour l'ensemble des paragraphes pouvant contenir la réponse et ensuite sélectionne le plus pertinent qui est celui avec la plus grande probabilité.

3.4 Module d'extraction des réponses MER

En ce qui concerne le module d'extraction de la réponse, nous allons mettre en oeuvre un modèle qui, en se basant sur le passages reçu par le module de classification, va extraire une réponse précise à la question donnée.

Comme c'est le cas pour le classifieur, la tâche de réponse aux questions de BERT est une tâche de prédiction. Étant donné une question et un paragraphe de contexte, le modèle prédit un début et une fin à partir du paragraphe qui répond le plus probablement à la question.

Pour le 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 le jeu de données de SQuAD.

3.4.1 Le choix du jeu de données

Nombreux sont les jeux de données destinés aux systèmes QA, dans le tableau 3.1, nous allons voir les jeux de données fréquemment utilisés dans les systèmes QA.

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

TABLE 3.1: Les différents jeux de données disponibles.

Pour notre cas, nous allons travailler avec le jeu de données SQuAD (Rajpurkar et al., 2016) proposé par l'université de Stanford et qui contient un nombre impressionnant de questions (100.000 questions posées par des gens sur plus de 500 articles de différents domaines sur wikipedia), de plus, les questions sans réponses étaient le talon d'Achille des jeux de données des systèmes QA, 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.

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. Un passage est un paragraphe d'un article, et sa longueur est 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, nous avons 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 allons voir 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 "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 identifiant "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.

3.4.2 Entrainement de l'extracteur

Après avoir sélectionné le passage le plus pertinent parmi ceux sélectionnés par le MRI, le module d'extraction le prend comme entrée, accompagné par la question posée. Comme c'était le cas pour le module de classification, nous commençons par la tokenisation de la paire (question - réponse choisie par le MC) qui est l'entrée de notre extracteur.

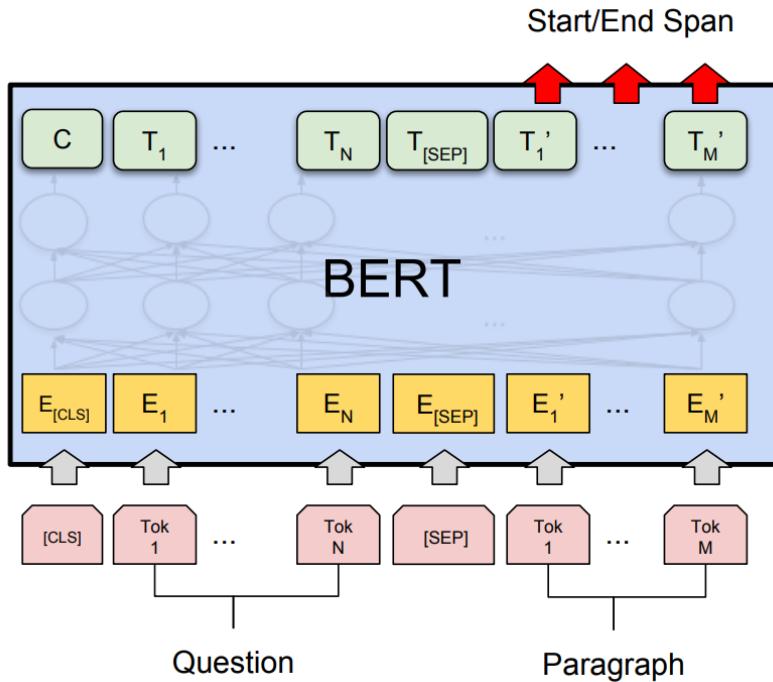


FIGURE 3.7: La tache de réponse aux questions avec BERT [2].

En sortie le modèle pré-entraîné de BERT produit

Comme le montre la figure 3.7, tout comme les tâches de paires de phrases, la question devient la première partie et le paragraphe la deuxième partie de la séquence de saisie. Seuls deux nouveaux paramètres sont appris lors du réglage fin : Un vecteur de début et un vecteur de fin dont la taille est égale à la taille de la forme cachée. La probabilité que le jeton i soit le début de l'intervalle de réponse est calculée par $\text{softmax}(S, K)$ où :

- S : Le vecteur de départ.
- K : La sortie finale du transformateur du jeton i .

Nous illustrons cette tache par un simple exemple dans la figure 3.8 :

```

ENTREE :
> Question : "In what country is Normandy located?"
> Paragraphe : "The Normans (Norman: Normands; French: Normands; Latin: Normanni)
   were the people who in the 10th and 11th centuries gave their name to Normandy, a region in France."

```

```

SORTIE :
> Réponse : "France"

```

FIGURE 3.8: Exemple de l'entrée et sortie de la tache Question-Answering avec BERT.

Afin d'interagir avec les utilisateurs de notre système, nous avons mis en oeuvre une application web SPA⁴ complète qui fournit une expérience utilisateur facile en appliquant les dernières normes UI⁵ et l'utilisation du matériel design fourni par Google comme nous pouvons le voir dans les captures d'écran de notre interface dans le figure 3.9. Pour cela, nous avons utilisés pour le front-end un template Colorlib⁶ bâti en utilisant VueJs⁷, Bootstrap⁸ pour le front-end et le framework python Django⁹ pour le back-end.

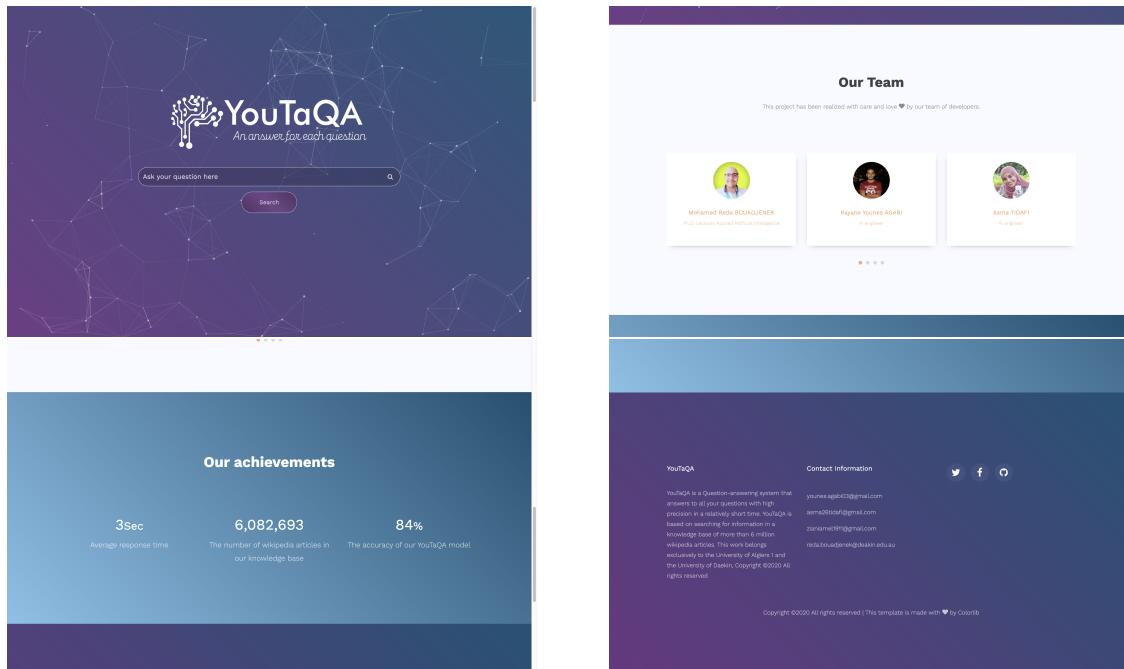


FIGURE 3.9: Captures d'écran de l'application web YouTaQA

3.5 Conclusion

- 4. **SPA (Single page application)** : c'est une application qui n'a pas besoin de recharger la page dans le navigateur pendant son utilisation.
- 5. **UI (user interface)** : signifie "interface utilisateur", c'est la présentation graphique d'une application.
- 6. <https://colorlib.com/wp/templates/>
- 7. **VueJs** : c'est un framework pour construire des interfaces utilisateur à base de javascript.
- 8. **Bootstrap** : c'est un framework CSS pour construire des interfaces utilisateur.
- 9. **Django** : c'est un framework web gratuit écrit en Python qui sert à développer des applications web.

Chapitre 4

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 Résultats du module de recherche d'informations 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 avons considéré comme requêtes pour notre moteur de recherche et comparer ses résultats avec les passages fournis par SQuAD. Pour cela, il nous a fallu construire un fichier résultat et un autre fichier nommé Qrels.

4.2.1 Fichier Qrels

Pour lancer l'évaluation d'un moteur de recherche avec **Trec-Eval**, ce dernier à besoin d'un fichier nommé Qrels qui permet de relier les passages de SQuAD (qui sont extraits de Wikipédia) et les vrais passages dans Wikipédia extraits par notre moteur de recherche. Cette étape est dû au fait que les articles et leurs passages dans wikipédia sont souvent mis à jours et modifiés, cela nous permet de définir la nouvelle forme de chaque passage de SQuAD (qui sont extract de Wikipédia version 2016) par rapport à

la dernière version de Wikipédia que nous utilisant comme base de documents pour notre système (version Mars 2020). Pour établir ce fichier, nous avons considéré les passages de SQuAD comme requêtes pour notre système. La structure finale de ce fichier sera comme suit :

PassageSquadId *0 DocumentId* *Revelance*

Tel que :

- ***PassageSquadId*** : L'identifiant du passage considéré comme requête dans SQuAD.
- ***DocumentId*** : L'identifiant du document retourné par notre système.
- ***0*** : Un champs ignoré par TREC-Eval mais qui doit être dans la structure.
- ***Revelance*** : La pertinence du document retrouvé (toujours 1 dans notre cas).

4.2.2 Fichier Résultats

Le fichier de résultats contient un classement des documents pour chaque requête générée automatiquement par notre système. C'est le fichier qui sera évalué par TREC-Eval en fonction de la "bonne réponse" fournie par le premier fichier Qrels. Contrairement au fichier Qrels qui prend comme requête, cette fois-ci nous allons utiliser les questions de SQuAD comme requêtes et à chaque requête, le moteur de recherche est configuré à retourner les 1000 meilleurs passages. Le fichier résultat aura un format comme suit :

QueryId *Q0 DocumentId* *Rank Score STANDARD*

Tel que :

- ***QueryId*** : L'identifiant de la requête (les identifiants des questions dans notre cas).
- ***Q0*** : Un champs ignoré par TREC-Eval mais qui doit être dans la structure.
- ***DocumentId*** : L'identifiant du passage retourné par notre MRI.
- ***Rank*** : Le classement du passage.
- ***Score*** : Le score attribué au document.
- ***STANDARD*** : Une chaîne de caractère ignorée par TREC-Eval mais qui doit elle aussi apparaître dans le fichier.

4.2.3 Méthodes de recherche employées

Postérieurement, nous avons lancé l'évaluateur TREC-Eval en lui fournissions les fichiers résultats et Qrels de 8 méthodes de recherche exactement afin de comparer les performances de chaque méthode et

choisir à la fin la meilleure méthode à utiliser.

Méthode de recherche VSM_SS_QQ¹

Dans cette méthode de requêtage, nous avons utilisé la métrique de classement des résultats VSM tout en utilisant une recherche simple des mots clés de la question de SQuAD dans le contenu des articles seulement.

Méthode de recherche VSM_SS_QTQ²

Pour cette méthode, ça sera la même chose que la méthode précédente sauf que la requête sera formée de la question de SQuAD en la concaténant avec le titre du passage fourni par SQuAD (comme nous avons vu dans la structure du dataset SQuAD, chaque passage à un titre et des questions dont ce passage contient les réponses de ces questions). Ce choix est justifié par le cas où nombreux sont les questions d'un même passage qui, au lieu de contenir le sujet en question, nous trouvons ce sujet référencé par un pronom personnel. Par exemple, nous avons le passage suivant de SQuAD :

Passage : “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”

Question N1 : ”When did Beyonce start becoming popular ?”

Question N2 : ”What did she want to be when she was a kid ?”

Dans cet exemple, nous remarquons que la deuxième question ne comporte pas le sujet principal de la question mais au lieu de ça il est remplacé par le pronom personnel “she” ce qui cause un problème durant l'évaluation de notre moteur de recherche et en conséquence fausser les résultats de l'évaluation.

Méthode de recherche VSM_MFS_QQ³

Cette méthode de recherche est basée sur la recherche ”multi-fields” c'est à dire la recherche des mots d'une requête (la question comme requête dans ce cas) est effectuée non seulement sur le contenu de l'article mais aussi sur le titre de ce dernier.

1. Acronyme de **VSM_SimpleSearch_Question as Query**

2. Acronyme de **VSM_SimpleSearch_Question+Title as Query**

3. Acronyme de **VSM_MultifieldsSearch_Question as Query**

Méthode de recherche VSM_MFS_QTQ⁴

Cette méthode utilise le même principe de la précédente méthode sauf que là, la requête est composée de la question concaténée au titre du passage de SQuAD pour éviter de fausser les résultats de notre système à cause des pronoms personnels.

Méthode de recherche BM25_SS_QQ⁵

La présente méthode est basé sur une simple recherche dans le contenu des articles en utilisant la question seule comme requête. Durant l'étape de classement des résultats, ici nous utilisons l'approche BM25 au lieu du VSM.

Méthode de recherche BM25_SS_QTQ⁶

Quant à cette méthode, ça sera une recherche simple dans le contenu des articles sauf que la requête sera constituée de la question et du titre du passage du dataset SQuAD.

Méthode de recherche BM25_MFS_QQ⁷

Là, nous avons implémenté une méthode de recherche qui emploie la recherche multifields (recherche appliquée dans le contenu et le titre des articles au même temps) et qui à comme requête la question de SQuAD seulement.

Méthode de recherche BM25_MFS_QTQ⁸

La dernière méthode de recherche est basée sur la recherche multifields comme la précédente sauf que la requête sera la question concaténé avec le titre du passage de SQuAD.

4.2.4 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.

4. Acronyme de **VSM_MultifieldsSearch_Question+title as Query**

5. Acronyme de **BM25_SimpleSearch_Question as Query**

6. Acronyme de **BM25_SimpleSearch_Question+title as Query**

7. Acronyme de **BM25_MultifieldsSearch_Question as Query**

8. Acronyme de **BM25_MultifieldsSearch_Question+title as Query**

Histogrammes MAP, MRR et R-Précision

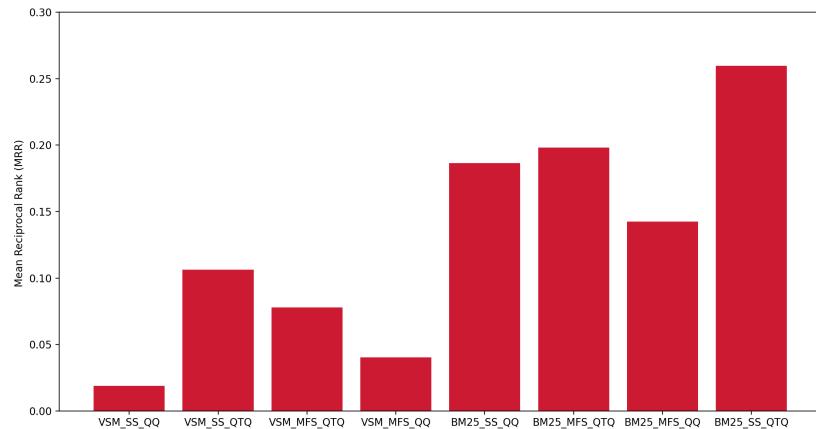


FIGURE 4.1: Graphe qui représente les valeurs MRR des résultats

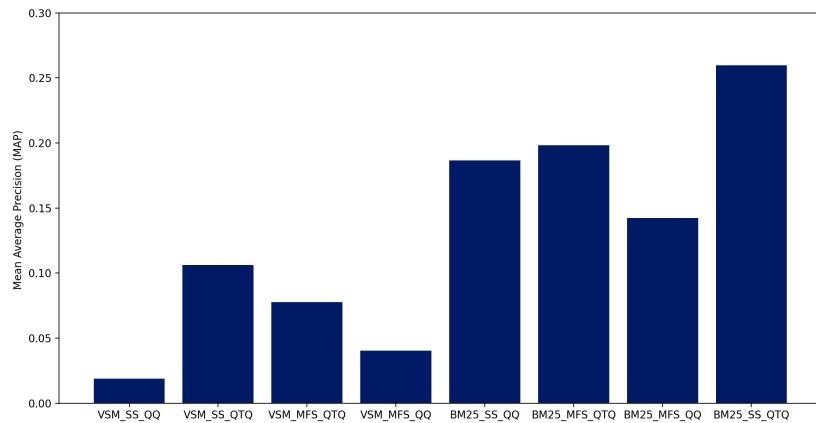


FIGURE 4.2: Graphe qui représente les valeurs MAP des résultats

Les histogrammes dans la figure 4.1 et 4.2 représentent respectivement la précision moyenne et le rang moyen de réciprocité 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 terme de précision moyenne. Cela dit, la métrique BM25 étant plus neuve et étant plus précise, permet une recherche plus performante et plus assurée.

Maintenant, Après avoir comparé les deux métriques de classement, nous passons directement à la comparaison des méthodes qui utilisent BM25, la encore, nous remarquons la supériorité de la méthode de recherche BM25_SS_QTQ, et comme nous pouvons pas utiliser les méthodes qui utilisent la question+titre comme requête durant le stade opérationnel de notre système (en production, le système n'aura que la question comme requête), la meilleure méthode utilisable durant le stade opérationnel de notre système YouTaQA sera celle qui est basée sur la recherche simple BM25_SS_QQ.

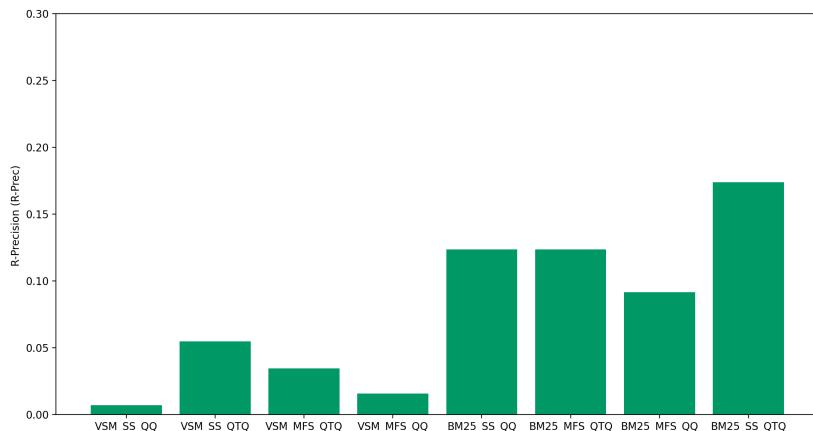


FIGURE 4.3: Graphe qui représente les valeurs R-Précision des résultats

Nous remarquons aussi dans l'histogramme du R-Précision 4.3, qui est une métrique hautement corrélée avec MAP, que les deux méthodes BM25_SS_QQ et BM25_MFS_QTQ ont quasiment les mêmes performances en terme de R-Précision.

Une chose de plus à remarquer dans l'histogramme du MRR, la méthode de recherche BM25_SS_QQ 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 documents 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. Par exemple, pour les graphes de la précision@K et rappel@K présentés dans la figure 4.4 et 4.5 respectivement, où K représente le nombre de résultats retournés suite à une recherche, permettent d'affirmer les performances supérieures réalisées par la méthode BM25_SS_QQ.

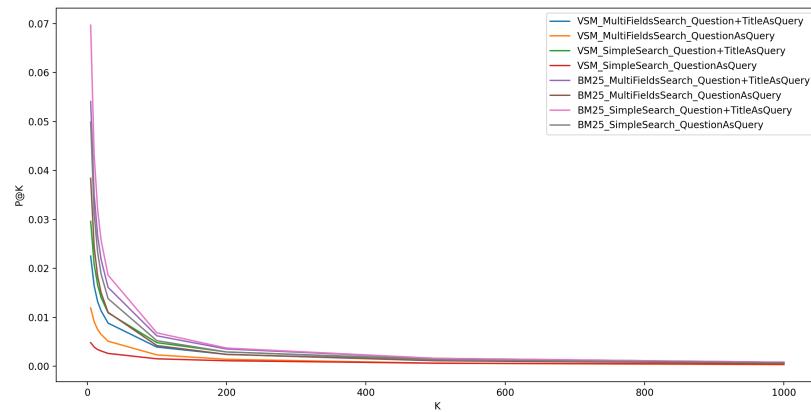


FIGURE 4.4: Graphe qui représente les valeurs de précision@K des résultats

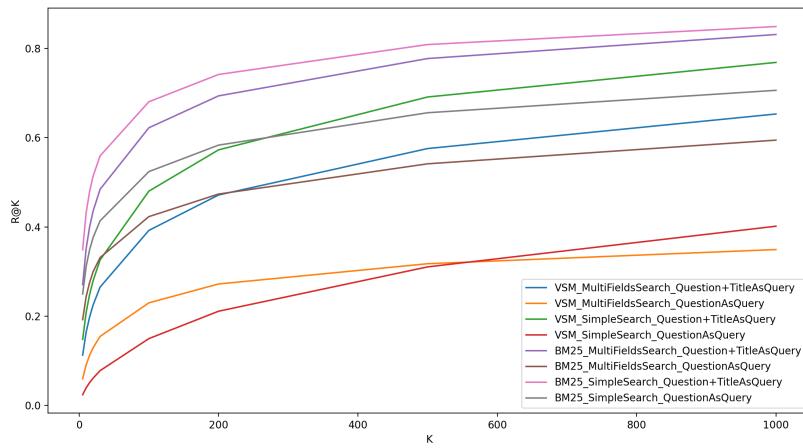


FIGURE 4.5: Graphe qui représente les valeurs du rappel@K des résultats

Par ailleurs, nous constatons des graphes de la précision@K (ou P@K) et du rappel@K que la précision représente le pourcentage de documents prédit correctement par rapport au nombre de documents erronés

retournés ce qui implique une baisse considérable de la précision en augmentant le nombre K de résultats retournés. Quant au rappel, qui donne le pourcentage des documents corrects qui sont donnés sans se préoccuper du nombre de documents erronés retournés, augmente en augmentant le nombre K de résultats retournés. En conséquence, nous avons établis un nouveau graphe présent dans la figure 4.6 qui mets en valeur la relation précision-rappel à chaque valeur de K. Encore une fois, la méthode BM25_SS_QQ figure comme étant la meilleure méthode de recherche en obtenant un rapport précision-rappel toujours au dessus des autres méthodes.

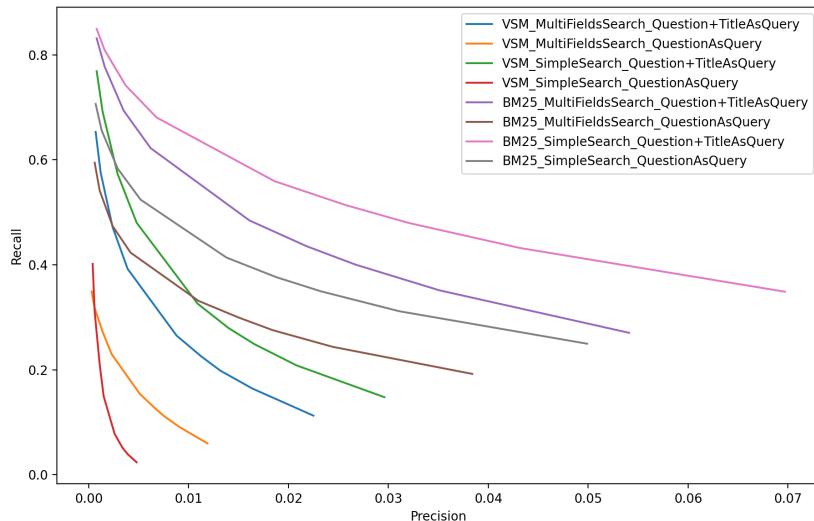


FIGURE 4.6: Graphe qui représente les valeurs du Précision-Rappel des résultats

Au final, 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_SS_QQ, qui en moyenne, permet d'avoir le bon document parmi les 5 premiers documents retournés par notre moteur de recherche ($MRR = 0.20$).

4.3 Résultats du module de classification des documents

4.4 Résultats du module d'extraction des réponses MER

4.5 Résultats globaux

4.6 Conclusion

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 précédent la recherche, 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 dû 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 amené 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.

4.7 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 d'exploitation complet. Pour cela, les perspectives envisageables afin d'améliorer ce travail sont multiples, 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'accroît quand on la partage*, comme le dit Socrate, sur ce, nous avons mis à disposition notre code source sur GitHub ici.

Bibliographie

- [1] Derias yacine. Recherche d'information : Prétraitement et indexation. 2018.
- [2] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert : Pre-training of deep bidirectional transformers for language understanding, 2018.
- [3] van Noord G. Bouma G., Mur J. Reasoning over dependency relations. 2005.
- [4] Maria Vargas-Vera and Enrico Motta. Aqua – ontology-based question answering system. In Raúl Monroy, Gustavo Arroyo-Figueroa, Luis Enrique Sucar, and Humberto Sossa, editors, *MICAI 2004 : Advances in Artificial Intelligence*, pages 468–477, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.
- [5] Boris Katz, Sue Felshin, Deniz Yuret, Jimmy Lin, Gregory Marton, Alton McFarland, and Baris Temelkuran. Omnibase : Uniform access to heterogeneous data for question answering. pages 230–234, 06 2002.
- [6] R. Mervin. An overview of question answering system. 1, 10 2013.
- [7] Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. *Modern Information Retrieval - the concepts and technology behind search, Second edition*. 2011.
- [8] Jure Leskovec, Anand Rajaraman, and Jeffrey David Ullman. *Finding Similar Items*, page 68–122. Cambridge University Press, 2 edition, 2014.
- [9] Shahzad Qaiser and Ramsha Ali. Text mining : Use of tf-idf to examine the relevance of words to documents. *International Journal of Computer Applications*, 181, 07 2018.
- [10] Stephen Robertson and Hugo Zaragoza. The probabilistic relevance framework : Bm25 and beyond. *Foundations and Trends in Information Retrieval*, 3 :333–389, 01 2009.

-
- [11] Simone Teufel. *An Overview of Evaluation Methods in TREC Ad Hoc Information Retrieval and TREC Question Answering*, volume 37, pages 163–186. 04 2007.
 - [12] Aditya Jain, Gandhar Kulkarni, and Vraj Shah. Natural language processing. *International Journal of Computer Sciences and Engineering*, 6 :161–167, 01 2018.
 - [13] Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura E. Barnes, and Donald E. Brown. Text classification algorithms : A survey. *CoRR*, abs/1904.08067, 2019.
 - [14] Hasim Sak, Andrew W Senior, and Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling. 2014.
 - [15] Sepp Hochreiter. 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, April 1998.
 - [16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.
 - [17] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. Lstm : A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10) :2222–2232, 2016.
 - [18] Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention : Specialized heads do the heavy lifting, the rest can be pruned. *arXiv preprint arXiv :1905.09418*, 2019.
 - [19] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training, 2018.
 - [20] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *ieee transactions on knowledge and data engineering*. 22 (10) : 1345, 1359, 2010.
 - [21] Kai Ming Ting. *Precision and Recall*, pages 781–781. Springer US, Boston, MA, 2010.
 - [22] Nick Craswell. *R-Precision*, pages 2453–2453. Springer US, Boston, MA, 2009.
 - [23] Steven M. Beitzel, Eric C. Jensen, and Ophir Frieder. *MAP*, pages 1691–1692. Springer US, Boston, MA, 2009.
 - [24] Nick Craswell. *Mean Reciprocal Rank*, pages 1703–1703. Springer US, Boston, MA, 2009.
 - [25] Passent M. ElKafrawy, Amr M. Sauber, and Nada A. Sabry. Semantic question answering system using dbpedia. In *Lecture Notes in Computer Science*, pages 821–832. Springer International Publishing, 2018.

-
- [26] Bolanle Ojokoh and Emmanuel Adebisi. A review of question answering systems. *Journal of Web Engineering*, 17 :717–758, 01 2019.
 - [27] W. A. Woods. Progress in natural language understanding. In *Proceedings of the June 4-8, 1973, national computer conference and exposition on - AFIPS 73*. ACM Press, 1973.
 - [28] Bert F. Green, Alice K. Wolf, Carol Chomsky, and Kenneth Laughery. 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, 1961.
 - [29] Oleksandr Kolomyets and Marie-Francine Moens. A survey on question answering technology from an information retrieval perspective. *Information Sciences*, 181(24) :5412–5434, dec 2011.
 - [30] Edward Shortliffe. 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*, 10 1977.
 - [31] Minsuk Lee, James Cimino, Hai Zhu, Carl Sable, Vijay Shanker, John Ely, and Hong Yu. Beyond information retrieval—medical question answering. *AMIA Annu Symp Proc*, pages 469–473, 02 2006.
 - [32] María-Dolores Olvera-Lobo and Juncal Gutiérrez-Artacho. 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, 2011.
 - [33] YongGang Cao, Feifan Liu, Pippa Simpson, Lamont Antieau, Andrew Bennett, James J. Cimino, John Ely, and Hong Yu. AskHERMES : An online question answering system for complex clinical questions. *Journal of Biomedical Informatics*, 44(2) :277–288, apr 2011.
 - [34] Michael A Bauer and Daniel Berleant. Usability survey of biomedical question answering systems. *Human Genomics*, 6(1), sep 2012.
 - [35] Anne R. Diekema, Ozgur Yilmazel, and Elizabeth D. Liddy. Evaluation of restricted domain question-answering systems. In *Proceedings of the Conference on Question Answering in Restricted Domains*, pages 2–7, Barcelona, Spain, July 2004. Association for Computational Linguistics.
 - [36] Farah Benamara and Patrick Saint-Dizier. Webcoop : A cooperative question-answering system on the web. 04 2003.
 - [37] Dan Moldovan, Marius Paşa, Sanda Harabagiu, and Mihai Surdeanu. Performance issues and error analysis in an open-domain question answering system. *ACM Transactions on Information Systems (TOIS)*, 21(2) :133–154, apr 2003.

-
- [38] Mohamed Reda Bouadjenek, Hakim Hacid, and Mokrane Bouzeghoub. 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, 03 2016.
 - [39] Eduarda Mendes Rodrigues and Natasa Milic-Frayling. Socializing or knowledge sharing ? In *Proceeding of the 18th ACM conference on Information and knowledge management - CIKM 09*. ACM Press, 2009.
 - [40] Kevin Kyung Nam, Mark S. Ackerman, and Lada A. Adamic. Questions in, knowledge in ? In *Proceedings of the 27th international conference on Human factors in computing systems - CHI 09*. ACM Press, 2009.
 - [41] Rich Gazan. Seven words you cant say on answerbag. In *Proceedings of the 27th ACM Conference on Hypertext and Social Media - HT 16*. ACM Press, 2016.
 - [42] Lada A. Adamic, Jun Zhang, Eytan Bakshy, and Mark S. Ackerman. Knowledge sharing and yahoo answers. In *Proceeding of the 17th international conference on World Wide Web - WWW 08*. ACM Press, 2008.
 - [43] Daniel Hasan Dalip, Marcos André Gonçalves, Marco Cristo, and Pavel Calado. 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, 2013.
 - [44] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 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, October 2013. Association for Computational Linguistics.
 - [45] Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. Large-scale simple question answering with memory networks. *ArXiv*, abs/1506.02075, 2015.
 - [46] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 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, 2008. Association for Computing Machinery.
 - [47] Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 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, 2014. Association for Computing Machinery.

- [48] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate, 2015.
- [49] Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher, and Hal Daumé III. 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, October 2014. Association for Computational Linguistics.
- [50] Karl Moritz Hermann, Tomás Kocišký, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. *CoRR*, abs/1506.03340, 2015.
- [51] Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. The goldilocks principle : Reading children’s books with explicit memory representations. *CoRR*, abs/1511.02301, 2015.
- [52] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad : 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250, 2016.
- [53] Daniel Hewlett, Alexandre Lacoste, Llion Jones, Illia Polosukhin, Andrew Fandrianto, Jay Han, Matthew Kelcey, and David Berthelot. 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, August 2016. Association for Computational Linguistics.
- [54] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 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, November 2018. Association for Computational Linguistics.
- [55] Pum-Mo Ryu, Myung-Gil Jang, and Hyun-Ki Kim. Open domain question answering using wikipedia-based knowledge model. *Inf. Process. Manage.*, 50(5) :683–692, September 2014.
- [56] David Ahn, Valentin Jijkoun, Gilad Mishne, Karin Müller, Maarten Rijke, and Stefan Schlobach. Using wikipedia at the trec qa track. 01 2004.
- [57] Davide Buscaldi and Paolo Rosso. 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, 2006.

-
- [58] Huan Sun, Hao Ma, Wen-tau Yih, Chen-Tse Tsai, Jingjing Liu, and Ming-Wei Chang. 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, 2015. International World Wide Web Conferences Steering Committee.
 - [59] Eric Brill, Susan Dumais, and Michele Banko. 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, July 2002.
 - [60] Petr Baudiš and Jan Šedivý. 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, 2015. Springer-Verlag.
 - [61] Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv :1704.00051*, 2017.
 - [62] Seunghyun Yoon, Franck Dernoncourt, Doo Soon Kim, Trung Bui, and Kyomin Jung. 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, 2019. Association for Computing Machinery.
 - [63] Weijie Bian, Si Li, Zhao Yang, Guang Chen, and Zhiqing Lin. 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, 2017. Association for Computing Machinery.
 - [64] Cody Kwok, Oren Etzioni, and Daniel S. Weld. Scaling question answering to the web. *ACM Trans. Inf. Syst.*, 19(3) :242–262, July 2001.
 - [65] Diego Molla Aliod and José González. Question answering in restricted domains : An overview. *Computational Linguistics*, 33 :41–61, 03 2007.
 - [66] Vanessa Lopez, Enrico Motta, Marta Sabou, and Miriam Fernandez. Poweraqua : A multi-ontology based question answering system-v1.
 - [67] Zhiping Zheng. 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, 2003. Association for Computational Linguistics.
 - [68] Wilson Wong. Practical approach to knowledge-based question answering with natural language understanding and advanced reasoning, 2007.

-
- [69] A Chandra Obula Reddy and K Madhavi. A survey on types of question answering system. *IOSR-JCE*, 19(6) :19–23, 2017.
 - [70] Michael Kaisser. Qualim at trec 2005 : Web-question answering with framenet. In *TREC*, 2005.
 - [71] María Dolores Olvera-Lobo and Juncal Gutiérrez-Artacho. Open- vs. restricted-domain qa systems in the biomedical field. *Journal of Information Science*, 37 :152–162, 04 2011.
 - [72] Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 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, 2018.
 - [73] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Sooricut. Albert : A lite bert for self-supervised learning of language representations, 2019.
 - [74] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet : Generalized autoregressive pretraining for language understanding, 2019.
 - [75] Zhuosheng Zhang, Junjie Yang, and Hai Zhao. Retrospective reader for machine reading comprehension, 2020.
 - [76] Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension, 2016.
 - [77] Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. Qanet : Combining local convolution with global self-attention for reading comprehension, 2018.
 - [78] Siddhant Garg, Thuy Vu, and Alessandro Moschitti. Tanda : Transfer and adapt pre-trained transformer models for answer sentence selection, 2019.
 - [79] Seunghyun Yoon, Franck Dernoncourt, Doo Kim, Trung Bui, and Kyomin Jung. A compare-aggregate model with latent clustering for answer selection. pages 2093–2096, 11 2019.

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 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 in-depth learning modules of our system have been 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 an 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 xx% while the response extraction model reached an F1 score of xx%.

Experiments on the dataset demonstrate the effectiveness of the proposed method, and the results of the comparison show that our architecture gives a more Question-Answering domain. **Keywords :** Information Retrieval, Deep Learning, Natural Language Processing, Transfer Learning.

ملخص

الكلمات المفتاحية