



HAL
open science

Intégration d'un historique de conversation dans un système de dialogue

Louis Mongin

► **To cite this version:**

Louis Mongin. Intégration d'un historique de conversation dans un système de dialogue. Sciences de l'Homme et Société. 2021. dumas-03544109

HAL Id: dumas-03544109

<https://dumas.ccsd.cnrs.fr/dumas-03544109>

Submitted on 26 Jan 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Intégration d'un historique de conversation dans un système de dialogue

**Louis
MONGIN**

Sous la direction de Sahar GHANNAY, Sophie ROSSET et Claude PONTON

Laboratoire : Laboratoire Interdisciplinaire des Sciences du Numérique
(LISN)

UFR LLASIC
Département Sciences du langage

Mémoire de master 2 mention Sciences du langage - 20 crédits

Parcours : Industries de la langue - orientation professionnelle

Année universitaire 2020-2021

Intégration d'un historique de conversation dans un système de dialogue

**Louis
MONGIN**

Sous la direction de Sahar GHANNAY, Sophie ROSSET et Claude PONTON

Laboratoire : Laboratoire Interdisciplinaire des Sciences du Numérique
(LISN)

UFR LLASIC
Département Sciences du langage

Mémoire de master 2 mention Sciences du langage - 20 crédits

Parcours : Industries de la langue - orientation professionnelle

Année universitaire 2020-2021

DÉCLARATION ANTI-PLAGIAT

1. Ce travail est le fruit d'un travail personnel et constitue un document original.
2. Je sais que prétendre être l'auteur d'un travail écrit par une autre personne est une pratique sévèrement sanctionnée par la loi.
3. Personne d'autre que moi n'a le droit de faire valoir ce travail, en totalité ou en partie, comme le sien.
4. Les propos repris mot à mot à d'autres auteurs figurent entre guillemets (citations).
5. Les écrits sur lesquels je m'appuie dans ce mémoire sont systématiquement référencés selon un système de renvoi bibliographique clair et précis.

PRENOM : Louis

NOM : MONGIN

DATE : 31/08/21

Sommaire

PREAMBULE.....	8
INTRODUCTION.....	9
LE PROJET LIHLITH.....	9
MISSIONS ET OBJECTIFS	10
Partie 1 - État de l’art.....	11
CHAPITRE 1. LES SYSTEMES DE DIALOGUE.....	12
1. MODULE DE COMPREHENSION DU LANGAGE	13
2. MODULE DE GESTION DU DIALOGUE	13
3. MODULE DE GENERATION DE LA REPONSE.....	14
CHAPITRE 2. L’HISTORIQUE DE DIALOGUE DANS LES SYSTEMES CONVERSATIONNELS.....	15
1. POURQUOI UN HISTORIQUE ?	15
2. LES MODELES CONVQA.....	15
3. LES DIFFERENTES METHODES.....	17
4. DATASETS UTILISES.....	20
CHAPITRE 3. ÉTUDE DE MODELES CONVQA	22
1. MODELE HAM.....	22
2. RESULTATS DES MODELES LES PLUS PERFORMANTS	24
Partie 2 - Contributions et expérimentations	25
CHAPITRE 4. PARTICULARITES DU SYSTEME DE DIALOGUE.....	26
1. DESCRIPTION DU SYSTEME.....	26
2. DATASET DoQA.....	29
CHAPITRE 5. PROTOCOLE EXPERIMENTAL.....	31
1. CHOIX DES DONNEES ET PARAMETRES DU MODELE	31
2. ENTRAINEMENTS.....	32
CHAPITRE 6. ENRICHIR LE CONTEXTE.....	34
1. INTEGRATION DES SLOTS AU MODELE.....	34
2. INTEGRATION DES SLOTS AU SYSTEME.....	37
CHAPITRE 7. DIFFICULTES ET PROBLEMES.....	39
1. ENTRAINEMENT DU MODELE AVEC SLURM	39
2. PROBLEMES RENCONTRES	39
Partie 3 - Résultats et analyse	41
CHAPITRE 8. INTERPRETATION DES RESULTATS.....	42
1. METRIQUE D’EVALUATION	42
2. RESULTATS DE L’ENTRAINEMENT	43
3. RESULTATS DE L’ENTRAINEMENT AVEC SLOTS.....	44
CHAPITRE 9. PERSPECTIVES D’AMELIORATIONS.....	48
1. FINE-TUNING DU MODELE AVEC LES DONNEES DU COOKBOOK.....	48
2. AMELIORATION DE L’UTILISATION DE DoQA.....	48

3. FINE-TUNING DE BERT SUR DES DONNEES DE SLOTS	49
CONCLUSION	50
1. BILAN DU TRAVAIL ACCOMPLI	50
2. BILAN PERSONNEL	50
BIBLIOGRAPHIE.....	52
SITOGRAPHIE	53
GLOSSAIRE	54
SIGLES ET ABREVIATIONS	56
TABLE DES FIGURES.....	57
TABLE DES TABLEAUX	58
TABLE DES ANNEXES.....	59
RÉSUMÉ	64
ABSTRACT.....	64

Préambule

Ce mémoire a été rédigé en parallèle d'un stage de fin d'études de 5 mois au Laboratoire Interdisciplinaire des Sciences du Numérique (LISN) situé à Orsay. Ce stage s'inscrit dans le cadre de mon cursus de Master Sciences du Langage parcours Industries de la Langue à l'Université Grenoble Alpes. C'est un master pluridisciplinaire spécialisé dans le traitement automatique des langues (TAL).

Le laboratoire LISN est une unité du CNRS regroupant 16 équipes de recherche. Il a été créé en janvier 2021 suite à la fusion du Laboratoire d'Informatique pour la Mécanique et les Sciences de l'Ingénieur (LIMSI) et du Laboratoire de Recherche en Informatique (LRI). Il est composé de cinq départements : Mécanique des Fluides-Energétiques, Interaction avec l'Humain, Sciences des Données, Algorithmes Apprentissage et Calcul et Sciences et Technologies des Langues.

J'ai rejoint pendant ces 5 mois l'équipe ILES (Information, Langue Ecrite et Signée) appartenant au groupe Sciences et Technologies des Langues. L'équipe ILES se focalise sur le traitement des données langagières écrites et signées. Ses thèmes récurrents sont la recherche d'information, le traitement des langues des signes et les agents conversationnels. Elle est actuellement composée d'une quarantaine de membres.

Ce stage m'a été proposé par Sahar Ghannay, membre du laboratoire LISN et Sophie Rosset, directrice du LISN. Pendant ces cinq mois, elles m'ont accompagné dans la réalisation de mon stage. Je l'ai commencé en distanciel avec des rendez-vous réguliers en visio-conférences. J'ai passé les deux derniers mois du stage en semi-télétravail, où j'ai pu me rendre au laboratoire pour rencontrer une partie de l'équipe ILES.

J'ai également été accompagné par mon tuteur universitaire Claude Ponton, membre du laboratoire LIDILEM (Linguistique et Didactique des Langues Étrangères et Maternelles), qui a suivi le déroulement de mon stage et m'a aidé dans la rédaction de ce mémoire.

Introduction

Le projet sur lequel j'ai travaillé pendant ce stage concerne un système de dialogue orienté tâches. Mes objectifs et missions furent tout d'abord d'étudier les pistes d'amélioration du système, en particulier l'intégration d'un historique de conversation. Suite à ça, il m'a fallu faire une proposition d'amélioration et la mettre en pratique dans le code. Pour finir, la dernière étape fut l'évaluation des performances du système.

Ce mémoire est divisé en trois parties principales. Tout d'abord, j'établirai un état de l'art sur l'avancée des travaux de recherche sur le sujet. Ma seconde partie sera consacrée à la méthodologie de mon travail et tout particulièrement à l'établissement de mon protocole expérimental. Je terminerai ce mémoire en examinant les résultats que j'ai pu obtenir suite aux différents tests réalisés et les perspectives d'amélioration que j'ai pu observer.

Le projet LIHLITH

Il est tout d'abord important de préciser le contexte du projet dans lequel s'inscrit le système de dialogue sur lequel j'ai travaillé.

Le projet LIHLITH (*Learning to Interact with Humans by Lifelong Interaction with Humans*), financé par CHIST-ERA, rassemble l'Université du Pays Basque, le laboratoire LISN, l'*Universidad Nacional de Educacion a Distancia*, l'Université des Sciences Appliquées de Zurich et l'entreprise Synapse Développement.

Le projet s'intéresse aux systèmes conversationnels et a pour but d'améliorer de manière générale les performances des systèmes de dialogue, notamment en exploitant les retours utilisateurs. Il se focalise sur le *Lifelong Learning*, processus au cours duquel la machine apprend au fur et à mesure de ses interactions pour être capable de réutiliser des informations communiquées par l'utilisateur.

Missions et objectifs

L'une des applications de ce projet est un système de dialogue orienté tâches sur le domaine de la cuisine. Un système de dialogue est un système complexe qui permet à un utilisateur d'interagir en langue naturelle avec une machine. C'est sur ce système que mon travail s'est dirigé.

Sa particularité est qu'il propose à l'utilisateur deux scénarios différents. L'utilisateur peut soit demander des recettes, soit poser des questions plus générales sur la cuisine. Le système obtient une bonne performance lorsqu'il s'agit du premier scénario. En revanche, il est difficile d'obtenir des réponses adéquates lorsque l'utilisateur pose des questions plus générales. Pour l'améliorer, mes tuteurs m'ont proposé d'étudier l'intégration d'un historique de dialogue.

Avant de revenir plus en détail sur les particularités du système de dialogue sur lequel j'ai travaillé, je vais tout d'abord établir un état de l'art des modèles proposés par la communauté du TALN (Traitement Automatique de la Langue Naturelle).

Partie 1

-

État de l'art

La première étape de mon stage fut un travail de documentation afin de mieux comprendre les problématiques et buts visés de ce domaine de recherche et d'étudier les solutions existantes. Dans cette première partie, je vais par conséquent établir un état de l'art en m'appuyant sur les articles scientifiques que j'ai lus au début et tout au long de mon stage. Je vais tout d'abord revenir sur les enjeux et difficultés d'intégrer un historique à un système de dialogue. Ensuite, j'expliquerai les rôles des différents éléments qui composent un modèle ConvQA (*Conversational Question Answering*). Enfin, je présenterai plus en détail un modèle qui a retenu mon attention.

Chapitre 1. Les systèmes de dialogue

Le but d'un agent conversationnel est de pouvoir communiquer avec un humain à travers un dialogue de questions/réponses. Les systèmes de dialogue, agents conversationnels ou encore *chatbots*, utilisent les technologies de l'apprentissage automatique¹ (aussi appelé apprentissage machine ou *machine learning* en anglais), de l'intelligence artificielle et de la recherche d'information. Alors que les *chatbots* vont avoir un dialogue sous format textuel, les agents conversationnels peuvent également converser avec la parole en utilisant les technologies de reconnaissance et de synthèse de la parole. Nous nous focaliserons ici sur les systèmes de dialogue de type textuel.

Un système de dialogue est composé de différents modules, différents composants ayant leur propre rôle dans le système. Les trois modules que l'on retrouve toujours sont les modules de compréhension du langage, de gestion du dialogue et pour finir, de génération de la réponse. La structure globale d'un système de dialogue peut être visualisée à l'aide de la figure 1.

¹ Cf. Glossaire

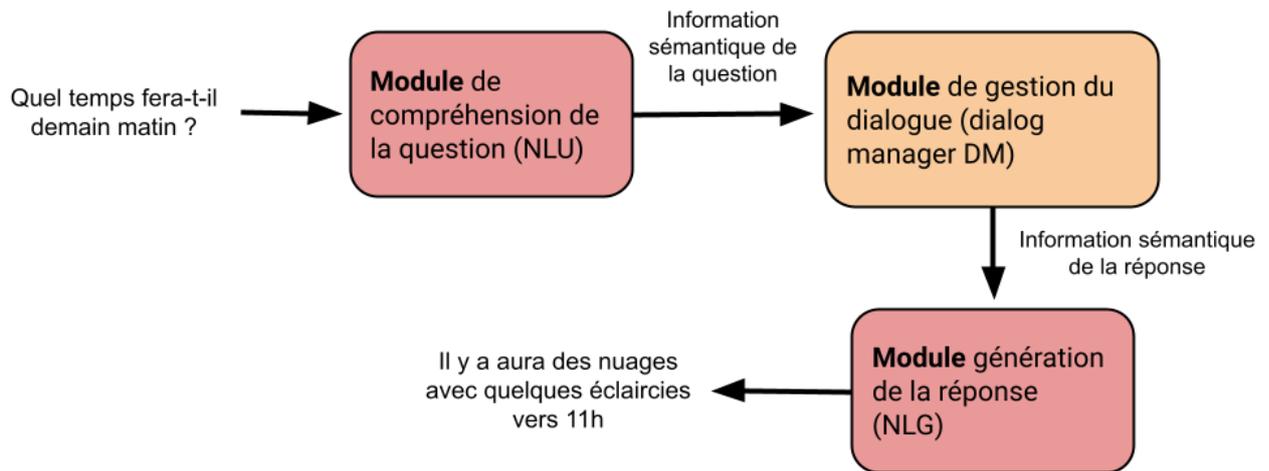


Figure 1 - Schéma de la structure générale d'un système de dialogue

1. Module de compréhension du langage

Le module de compréhension du langage est aussi souvent appelé NLU (en anglais, *natural language understanding*). Le rôle de ce module est de récupérer grâce à un analyseur sémantique les informations principales de la requête. Le domaine de détection des concepts (*slot-filling*) permet de récupérer ce que l'on appelle des slots, aussi appelés concepts. C'est une méthode de TAL qui permet d'étiqueter les mots selon leur sens dans la phrase. Ce module va par conséquent récupérer les slots de la phrase qui nous intéressent ainsi que leur valeur associée et transmettre ces informations au module de gestion du dialogue.

2. Module de gestion du dialogue

Le module de gestion du dialogue fait le plus gros du travail dans un système conversationnel. C'est lui qui va choisir la réponse à donner à l'utilisateur. Pour cela, il récupère la question et son information sémantique et fait le lien avec les données (base de données, divers documents, etc). Si besoin, un sous module est appelé pour récupérer les données appropriées à l'aide du domaine de la recherche d'information (IR, en anglais, *information retrieval*). C'est aussi lui qui va gérer tout le contexte autour du dialogue, dont l'historique de conversation. Le module renvoie l'information sémantique en sortie au module de génération.

3. Module de génération de la réponse

Le module de génération de la réponse (NLG, en anglais, *natural language generation*) s'occupe tout simplement de transformer l'information sémantique de la réponse en réponse textuelle, compréhensible pour un utilisateur humain.

Chapitre 2. L'historique de dialogue dans les systèmes conversationnels

1. Pourquoi un historique ?

Le défi principal d'un système de dialogue est d'avoir le dialogue le plus interactif possible et de simuler une conversation avec un humain. Pour cela, il est indispensable que le système prenne en compte l'entièreté du dialogue et non pas seulement la dernière question posée. Autrement dit, les tours de dialogue précédents jouent un rôle particulièrement important dans la compréhension de la question courante puisqu'ils participent au contexte autour de celle-ci. La figure 2 met en exergue l'importance de l'historique de conversation puisque pour comprendre la dernière question, la machine doit de se rappeler de la question précédente.

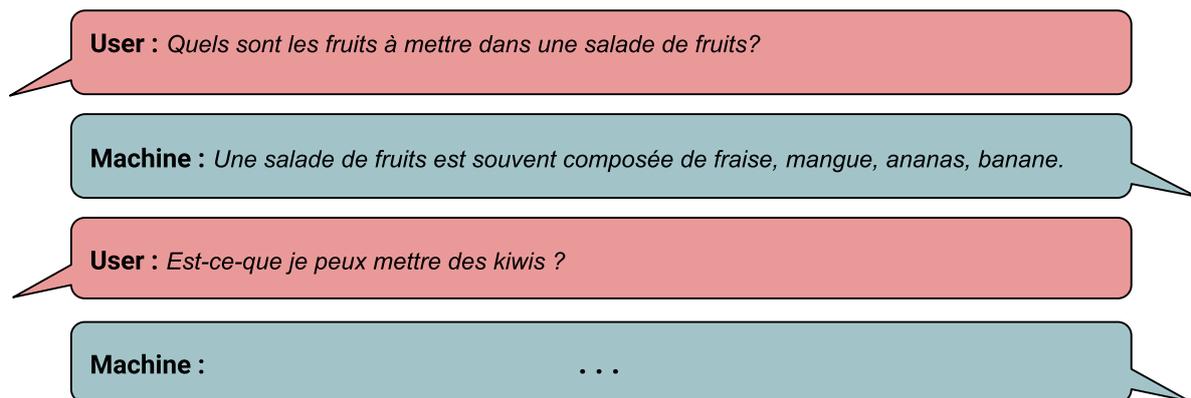
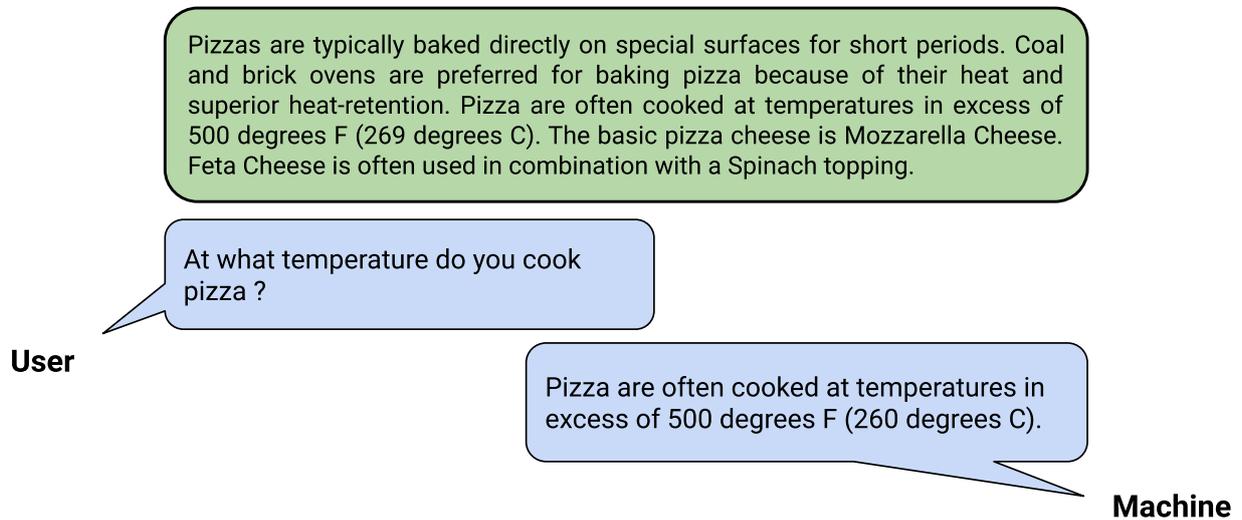


Figure 2 - Exemple d'un dialogue utilisateur/machine nécessitant un historique.

2. Les modèles ConvQA

Les modèles ConvQA sont une extension du domaine de recherche MRC (*machine reading comprehension*) (Zhang et al. 2019). Les modèles MRC sont des systèmes auxquels on fournit un paragraphe, aussi appelé contexte ou passage, et une série de questions en lien avec le paragraphe. Pour chaque question, le système doit trouver la réponse à l'intérieur du

paragraphe. Par conséquent, il est attendu de la machine qu'elle puisse lire le texte, le comprendre et en extraire les réponses.



*Figure 3 - Exemple d'un dialogue utilisant un modèle MRC.
Le système va directement chercher la réponse dans le paragraphe.*

De plus, on dit des modèles ConvQA que ce sont des systèmes à tours multiples (*multi-turn* en anglais). Dans un système de dialogue, un tour de dialogue se réfère à une question et sa réponse parmi un dialogue composé de paires de question-réponse. Par conséquent, la particularité de ces systèmes à tours multiples est qu'ils n'utilisent non pas seulement le tour actuel pour répondre à la question mais également tout ou une partie des tours précédents pour enrichir le contexte. Cette notion de tours multiples revient donc à prendre en compte l'historique de dialogue. Une représentation des ces deux types de systèmes, à tour unique et à tours multiples, peut être visualisée dans la figure 4.

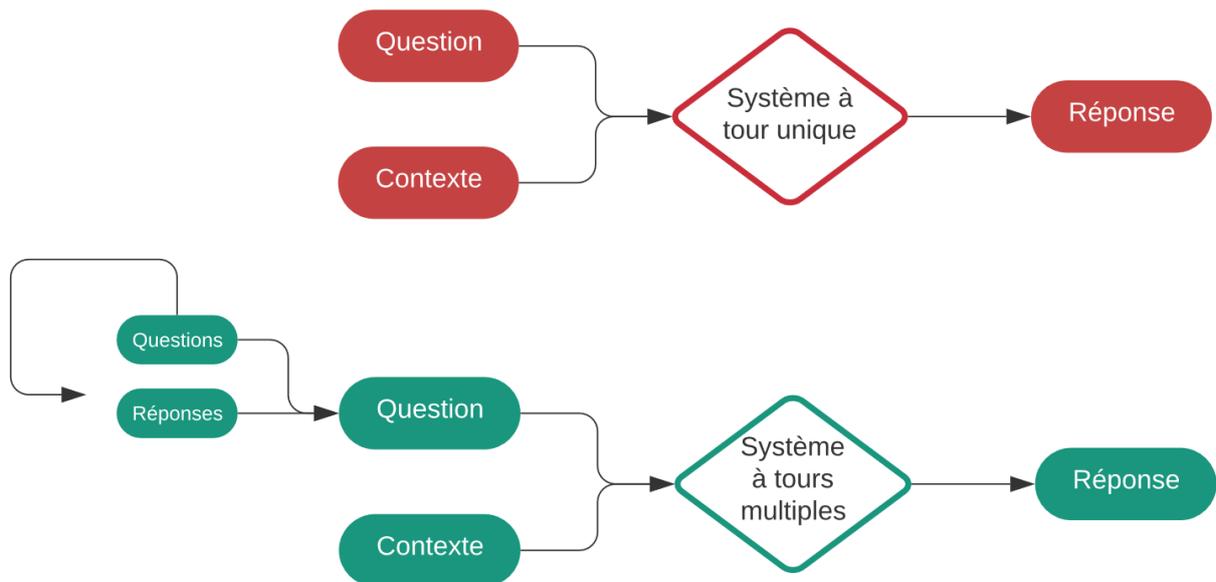


Figure 4 - Visualisation d'un système à tour unique et d'un système à tours-multiples.

3. Les différentes méthodes

Suite aux différents modèles qui ont été proposés par la communauté de recherche sur ce domaine, les auteurs de Gupta, Rawat, et Yu 2020, proposent un modèle générique pour illustrer les différents composants d'un modèle ConvQA. Je souhaite donc revenir sur ces éléments et leurs fonctions dans l'ensemble du modèle.

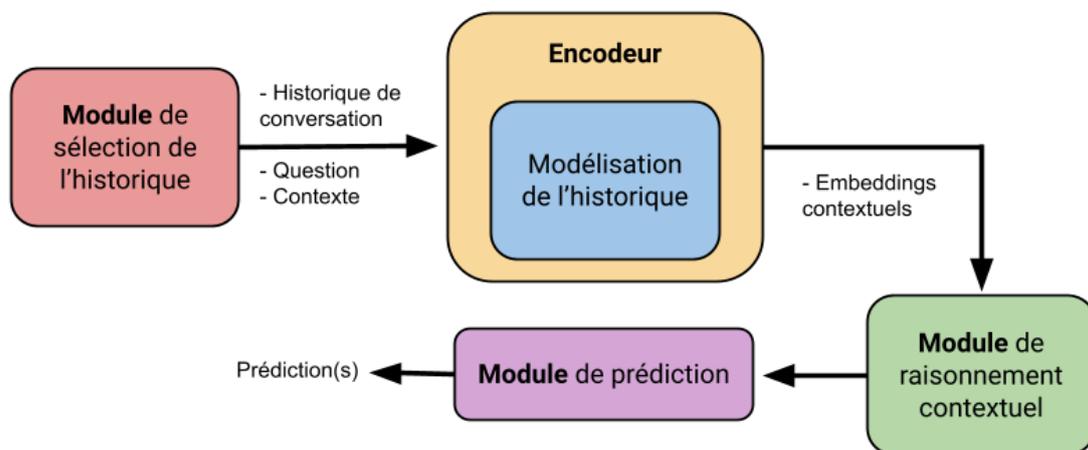


Figure 5 - Représentation générique d'un modèle ConvQA (inspiré de Gupta, Rawat, et Yu 2020)

Un modèle ConvQA est composé de quatre éléments principaux : un module de sélection d'historique, un encodeur, un module de raisonnement et un module de prédiction.

3. 1. SELECTION DE L'HISTORIQUE

On retrouve dans chaque modèle ConvQA un module de sélection d'historique. C'est lui qui va s'occuper de sélectionner quelle partie de l'historique il est intéressant d'exploiter pour aider le système à répondre à la question. Certains modèles vont par défaut récupérer toutes les questions/réponses des n-tours précédents. Toutefois, prendre une grande partie de l'historique peut faire baisser la performance du système. En effet, en récupérant beaucoup de questions/réponses, on augmente les chances qu'il y ait eu un changement de sujet au fur et à mesure de la conversation. De ce fait, la question actuelle n'a potentiellement plus de lien avec l'historique, ce qui peut induire le système en erreur.

Pour remédier à ce problème, certains modèles, comme BiDAF++ w/2-Context (Ohsugi et al. 2019) et SDNet (Zhu, Zeng, et Huang 2019), ne récupèrent pas plus de trois tours de dialogue avant la question actuelle. D'autres conseillent de ne prendre en compte que les questions et de ne pas utiliser l'historique des réponses (FlowQA). D'autres encore proposent une sélection d'historique plus poussée comme FlowDelta (Yeh et Chen 2020) dans le but de ne récupérer que les parties de l'historique qui aident à répondre à la question actuelle.

3. 2. ENCODEUR

Une fois que le modèle a sélectionné l'historique de dialogue, on encode les tokens² de la question actuelle, du contexte et de l'historique en plongements de mots. Les plongements de mots, aussi couramment nommés embeddings en anglais ou plongement lexical, est une technique de TAL permettant de représenter des mots avec des vecteurs de nombres réels. De cette manière, plus la différence entre les vecteurs de deux mots est moindre, plus on peut en conclure que le sens des deux mots est proche (cf Annexe n°1).

² Cf. Glossaire

La partie de l'encodeur qui nous intéresse tout particulièrement est la modélisation de l'historique, autrement dit l'intégration de l'historique au reste des tokens. On retrouve principalement trois méthodes de modélisation d'historique.

La première consiste simplement à une concaténation des tokens de l'historique, du paragraphe et de la question actuelle en encodant un séparateur pour différencier les éléments les uns des autres. Certains modèles proposent par exemple d'encoder les numéros des tours de dialogue à l'embedding de chaque question (GraphFlow (Chen, Wu, et Zaki 2020)). On peut voir ci-dessous une représentation de ce genre de concaténation.



La seconde méthode consiste à encoder les tokens du contexte avec des marqueurs d'historique. Autrement dit, au lieu de directement encoder les tokens des réponses de l'historique, on va encoder leurs positions à l'intérieur du paragraphe (HAM). Cette méthode est souvent combinée avec une concaténation simple pour les questions.

Il est également possible d'intégrer les représentations intermédiaires générées dans le module de raisonnement pour récupérer les informations sémantiques de l'historique sélectionné (FlowQA, FlowDelta, GraphFlow). Cette méthode évite de travailler directement sur des données textuelles brutes.

3.3. RAISONNEMENT CONTEXTUEL

Le module de raisonnement contextuel est la partie du système qui va transformer les embeddings de tokens en embeddings contextuels.

Pour cette étape, il existe autant de possibilités qu'il existe de modèle. Néanmoins, il est possible de la simplifier en utilisant un modèle de langue pré-entraîné comme BERT (*Bidirectional Encoder Representations from Transformers*). BERT est un modèle de langue développé par Google. Comme son nom l'indique, il a la particularité d'utiliser des

Transformers, un mécanisme d'attention composé d'un encodeur qui lit le texte en entrée et d'un décodeur qui fait une prédiction. Contrairement aux modèles unidirectionnels, l'encoder ne lit pas les mots de gauche à droite ou de droite à gauche mais toute la séquence en même temps. Cela lui permet de prendre en compte tout le contexte d'un mot, c'est-à-dire les autres mots qui l'entourent (Devlin et al. 2019).

Dans un système ConvQA, BERT permet de simplifier l'architecture des modules puisque les étapes d'encodage et de raisonnement se font simultanément. Dans cette approche, on regroupe la question et le contexte en marquant la séparation des deux avec des marqueurs d'embedding. On donne cette entrée à BERT qui retourne un embedding contextualisé pour chaque token de segment donné en entrée. Nous verrons plus en détails ce fonctionnement dans le chapitre 3 en étudiant un modèle en particulier qui utilise BERT.

3. 4. PREDICTION

Le module de prédiction va permettre, comme son nom l'indique, de prédire la réponse à la question. Il va récupérer les embeddings contextuels générés par le module de raisonnement et donner en sortie un extrait de texte provenant du contexte.

4. *Datasets utilisés*

Les modèles ConvQA utilisent en majorité les datasets QuAC (Choi et al. 2018) et/ou CoQA (Reddy, Chen, et Manning 2019). Ce sont des datasets de questions/réponses sous forme de dialogues. Chaque dialogue dispose d'un titre de section principal et d'un contexte sur lequel les questions vont se reposer.

4. 1. QUAC

QuAC est un dataset assez conséquent qui comporte 14 000 dialogues dont 10 000 questions. Les données ont été récupérées à partir de dialogues entre étudiants et professeurs. D'un côté, les étudiants posent des questions concernant des articles Wikipédia³, et de l'autre, les professeurs répondent aux questions à l'aide d'un extrait provenant de l'article.

³ <https://www.wikipedia.org/>

Contrairement à d'autres datasets, la majorité des questions de QuAC sont des questions ouvertes, parfois ambiguës et qui ne peuvent être réellement comprises qu'en prenant en compte le contexte autour de la question.

4.2. CoQA

CoQA quant à lui détient 127 000 questions provenant de 8000 dialogues. Les données ont été produites par des dialogues entre collègues de travail se posant des questions sur un passage textuel.

Puisque certains modèles n'ont été évalués qu'avec QuAC, pour plus de simplicité, j'ai choisi d'utiliser seulement comme référence le dataset QuAC. C'est aussi le dataset qui détient la structure qui se rapproche le plus de DoQA (Campos et al. 2020), un troisième dataset que nous utiliserons avec le système de dialogue du projet LIHLITH et que je présenterai dans le prochain chapitre. De ce fait, je pourrais mettre en miroir les performances du modèle avec d'un côté le dataset QuAC et de l'autre le dataset DoQA.

Chapitre 3. Étude de modèles ConvQA

Parmi les différents articles que j'ai pu lire, certains ont plus particulièrement retenu mon attention. Je vais m'attarder un peu plus longtemps dans cette partie sur le modèle HAM (History Attention Mechanism) sur lequel je me suis par la suite appuyé lors de mes expériences. Je vais par conséquent décrire ses particularités avant de brièvement les comparer à celles des autres modèles qui m'ont intéressé.

1. Modèle HAM

1.1. MODELISATION DE L'HISTORIQUE ET ENCODEUR

Le modèle HAM (Qu et al. 2019) utilise le modèle de langue BERT pour la partie encodage. C'est lui qui va encoder la question, le contexte et l'historique de conversation en représentations contextualisées. Cela revient à incorporer du contexte à l'intérieur des plongements de mots. Le fonctionnement de cet encodeur peut être observé dans la figure 6.

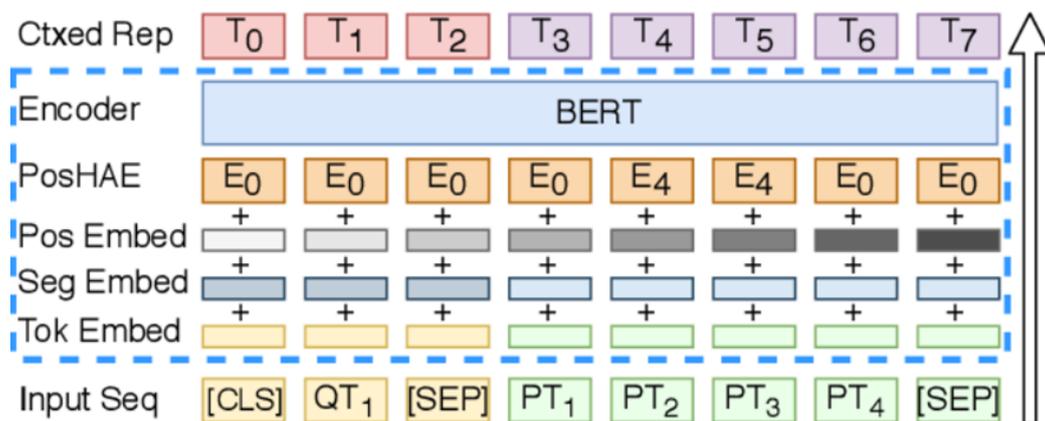


Figure 6 - Schéma des embeddings utilisés par le modèle HAM (figure provenant de Qu et al. 2019)

Les tokens de la question et les tokens du paragraphe sont concaténés et ajoutés en entrée à l'encodeur avec des marqueurs de séparation [SEP] et de début [CLS]. Le processus d'encodage peut ensuite démarrer. On ajoute des informations aux embeddings de tokens,

notamment l’embedding de segment (Seg Embed) qui va permettre de différencier la question du contexte. On ajoute également des embeddings positionnels (Pos Embed) qui vont indiquer la position du token dans la séquence. Pour finir, on retrouve aussi des plongements de mots qui indiquent les positions des tours de dialogue précédents (PosHAE). C’est juste après cette étape que BERT intervient pour transformer la séquence en représentations contextualisées.

1. 2. SELECTION DE L’HISTORIQUE

Comme son nom l’indique, le modèle HAM effectue une sélection de l’historique grâce à un mécanisme d’attention sur l’historique. La sélection de l’historique de conversation se base donc sur des poids qui vont permettre de mettre en avant les tours de dialogue qui sont les plus pertinents face à la question courante. Il est également possible de paramétrer le nombre de tours de dialogue maximum que le modèle prend en compte. C’est-à-dire que s’il est limité à 4 par exemple, le modèle ne prendra pas en compte plus de 4 pairs de questions-réponses précédentes.

1. 3. PREDICTION MULTI-TASKING

Le modèle HAM propose une prédiction que l’on appelle *multi-tasking*, c’est-à-dire que l’on prédit à la fois l’extrait de la réponse et des actes de dialogues. Ces derniers nous permettront de prédire si la question peut être répondue par oui ou non (yesno) et si la question fait suite à la précédente (followup). Ces actes de dialogue pourraient potentiellement enrichir le module qui génère la réponse (NLG).

2. Résultats des modèles les plus performants

Modèle	F1 QuAC	Partie de l'historique pris en compte	Technique utilisée pour la concaténation de l'historique	Prédictions supplémentaires ?
FlowDelta	65.5	Seulement les questions	Intégration des dernières représentations générées dans le module raisonnement	X
HAM	65.4	Toutes les questions et réponses	Concaténation des tokens du contexte avec la position des réponses des tours de dialogue précédents	Prédiction du <i>followup</i> et du <i>yesno</i>
GraphFlow	64.9	Toutes les questions et réponses	Concaténation des questions de l'historique avec l'information du numéro de tour de dialogue. Concaténation des marqueurs d'embedding de réponses au contexte. Des graphs de contexte sont ensuite utilisés	Prédiction du <i>yesno</i>
FlowQA	64.1	Seulement les questions	Intégration des dernières représentations générées dans le module raisonnement	Prédiction de s'il est impossible de répondre à la question
SDNet	N/A	Les deux dernières questions et réponses	Concaténation de l'historique des questions réponses à la question courante	X

Tableau 1 - Tableau reprenant cinq des modèles ConvQA présentant les meilleures caractéristiques (données provenant de (Gupta, Rawat, et Yu 2020))

Avant de choisir de débiter mes expérimentations sur le modèle HAM, je me suis attardé plus longuement sur cinq des modèles obtenant les meilleures performances. Le Tableau 1 recense cinq modèles avec leurs performances sur le dataset QuAC ainsi que leurs avantages et leurs inconvénients. Les modèles FlowDelta et HAM sont les modèles que j'ai trouvés à titre personnel les plus intéressants.

Les deux modèles obtiennent des scores similaires avec le dataset QuAC. En effet, ils établissent respectivement un score F1 de 65.4 pour HAM et de 65.5 pour FlowDelta. Je m'étais penché sur ces deux modèles en particulier pour leurs techniques de modélisation d'historique. Une des raisons majeures pour laquelle mon choix s'est plutôt orienté vers HAM est le fait que FlowDelta ne considère pas du tout les réponses dans l'historique. Autrement dit, il ne prend en compte que l'historique des questions. Au contraire, HAM propose de choisir le fait de prendre en compte ou non les réponses et de cette manière me permet de comparer les performances entre ces deux paramètres. Pour finir, la structure de HAM et son environnement python m'ont paru plus simple et intuitif et certainement plus en adéquation avec le système de dialogue du projet LIHLITH.

Partie 2

-

Contributions et expérimentations

Dans cette seconde partie, je vais tout d’abord expliquer plus en détail le fonctionnement du système de dialogue sur lequel j’ai travaillé. C’est ce qui me permettra par la suite d’établir un protocole d’expérimentation.

Chapitre 4. Particularités du système de dialogue

1. Description du système

La deuxième étape de mon stage fut la lecture et compréhension du code du système de dialogue. Pour m’aider, j’ai illustré le système avec des schémas des différents modules pour visualiser leurs structures et les données d’entrée et de sortie. Il me paraît tout d’abord important de décrire de manière générale la manière dont fonctionne ce système de dialogue. La figure 7 permet de visualiser la structure générale du système. On y retrouve les différents modules que je vais décrire plus en détails dans ce chapitre.

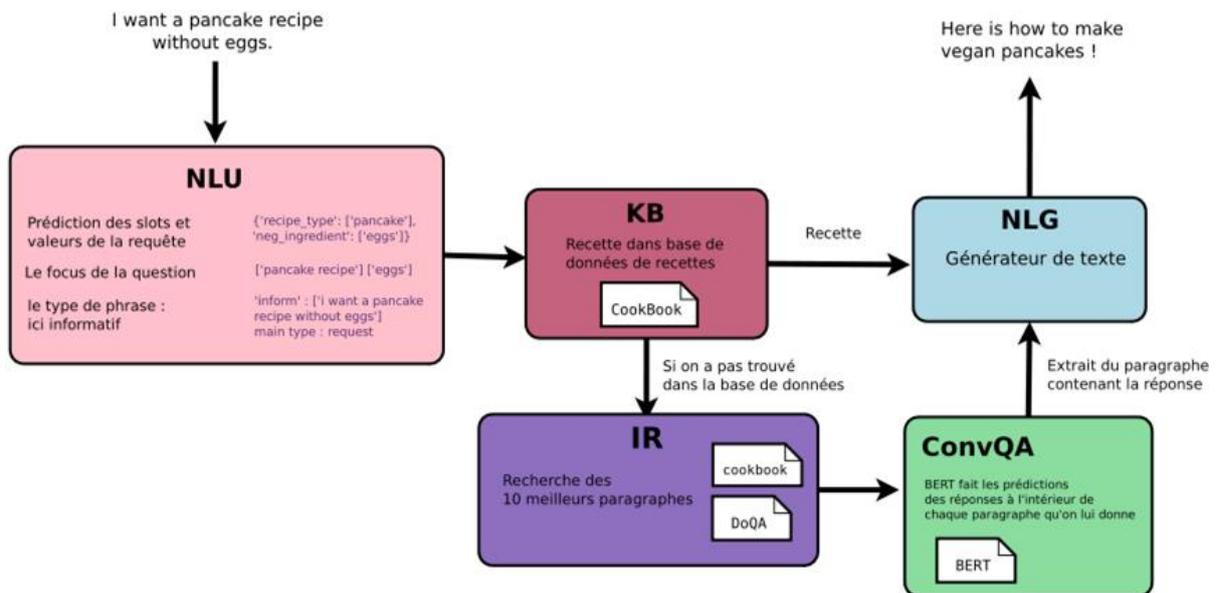


Figure 7 - Schéma global du système de dialogue

Le système de dialogue propose à l'utilisateur deux scénarios différents. Dans un premier temps, il est possible de trouver une recette adaptée à certains critères. Nous appellerons ce module⁴ KB (en anglais *knowledge base*, base de données).

Si la question est plus générale, le système va faire appel à un second module, le module FAQ (en anglais, *frequently asked questions*) qui utilise le dataset DoQA⁵.

1. 1. MODULE NLU

Le système de dialogue est tout d'abord composé du module NLU (en anglais, *Natural Language Understanding*). Comme nous l'avons dans le chapitre 1, ce module permet au système de récupérer grâce à un analyseur sémantique les informations principales de la requête, c'est-à-dire les slots, aussi appelés concepts. C'est une méthode de TAL qui permet d'étiqueter les mots selon leur sens dans la phrase. Ce module va par conséquent récupérer les concepts sémantiques de la phrase qui nous intéressent ainsi que leur valeur associée. Par exemple, respectivement, « *recipe_type* » et « *pancakes* » comme on peut le voir dans la figure 8, qui est une capture d'écran de l'étape de détection des concepts du système de dialogue du projet LIHLITH.

```
# Hello, I'm a cooking chatbot. I can help you find a recipe, answer questions about it or about cooking in general.
> I'm looking for a pancake recipe without eggs
HC: slots: {'recipe_type': ['pancake'], 'neg_ingredient': ['eggs']}
HC: utterance's focus: ['pancake recipe']
HC: utterance's context: ['eggs']
HC: utterance's types: {'inform': ['i am looking for a pancake recipe without eggs']}
HC: utterance's main type: request
context: None
EC: slots: {'recipe_type': ['pancake'], 'neg_ingredient': ['eggs']}
EC: routing: request
routing: Knowledge base accessed
```

Figure 8 - Capture d'écran de la détection des concepts suite à la requête d'un utilisateur.

⁴. Un module est une partie du système. En informatique, il s'agit d'une unité dans une architecture plus globale.

⁵. <https://huggingface.co/datasets/doqa>

1.2. MODULE KB

C'est le module KB qui va vérifier que la requête de l'utilisateur corresponde bien à une recette. Le système de dialogue va chercher dans sa base de données les recettes correspondantes à la requête pour les proposer à l'utilisateur. Ce dernier pourra alors choisir la recette qui l'intéresse puis se renseigner sur les étapes et les ingrédients qui la compose. La figure 9 montre un exemple de requête utilisant le scénario KB et le système indiquant la liste des recettes qu'il détient correspondants à la question.

```
# Hello, I'm a cooking chatbot. I can help you find a recipe, answer questions about it or about cooking in general.
> Do you have recipes with durian ?
HC: slots: {'ingredient': ['durian']}
HC: utterance's focus: ['recipes']
HC: utterance's context: ['durian']
HC: utterance's types: {'inform': ['do you have recipes with durian']}
HC: utterance's main type: request
context: None
EC: slots: {'ingredient': ['durian']}
EC: routing: request
routing: Knowledge base accessed
acts: {'act': 'tell_multiple_recipes', 'recipes': [{'object': "cookbook:'durian_pancake'", 'name': 'Durian Pancake'}, {'object': "cookbook:'pulut_durian'", 'name': 'Pulut Durian'}, {'object': "cookbook:'tempoyak'", 'name': 'Tempoyak'}]} {'act': 'select_one'}
# 3 recipes correspond to your search, including : "Durian Pancake" , "Pulut Durian" , "Tempoyak" . Which one do you want?
```

Figure 9 - Capture d'écran d'une requête au système qui utilise le scénario KB.
L'utilisateur demande une recette et le système propose plusieurs résultats.

1.3. MODULE FAQ

Si on ne trouve pas la réponse dans le module KB, le système va alors appeler le module FAQ. Ce module utilise deux sources principales de données. Il utilise une partie du dataset DoQA et un Cookbook, composé de divers paragraphes contenant des informations sur la cuisine. Dans un premier temps, le système va effectuer une recherche de paragraphes à l'aide de Solr⁶. Solr est un serveur de recherche très utilisé dans le domaine de recherche d'information (IR). Ce dernier va indexer les documents que l'on détient et rechercher les dix meilleurs paragraphes. C'est BERT qui va faire les prédictions des extraits de texte à l'intérieur des meilleurs paragraphes dans le sous module ConvQA. On peut voir un exemple de ces

⁶. Serveur de recherche d'Apache <https://solr.apache.org/>

résultats dans la figure 10. Pour trois paragraphes différents, le module ConvQA propose quatre extraits de texte différents.

```
# Hello, I'm a cooking chatbot. I can help you find a recipe, answer questions about it or about cooking in general.
> What colors are lentils ?
HC: slots: {'ingredient': ['lentils']}
HC: utterance's focus: ['colors', 'lentils']
HC: utterance's context: []
HC: utterance's types: {'description': 'what colors are lentils'}
HC: utterance's main type: description
context: None
EC: slots: {'ingredient': ['lentils']}
EC: routing: description
routing: FAQ base accessed
# Here's what I found in the documents : - colors that range from yellow to red-orange to green, brown or even black.
- A variety of lentils exist with colors that range from yellow to red-orange to green, brown or even black.
- lentils exist with colors that range from yellow to red-orange to green, brown or even black.
- colors that range from yellow to red-orange to green,
- colors
- tradition holds that a bean pot should be two-tone brown, on cream. .
- tradition holds that a bean pot should be two-tone brown,
- tradition holds that a bean pot should be two-tone brown, on cream
- but tradition holds that a bean pot should be two-tone brown, on cream. .
- tradition holds that a bean pot should be two-tone brown, on cream.
- Rice is tinted with different colors
- Rice is tinted with different colors and later sliced to make an image. .
- sushi, which makes a beautiful image. It is quite difficult to make this, and also quite expensive when bought. Rice is tinted with differen
t colors
- Sushi Saikuzushi of sushi is Saikuzushi is an artistic type of sushi, which makes a beautiful image.
- sushi, which makes a beautiful image. Anything else?
>
```

Figure 10 - Capture d'écran d'une requête utilisant le module FAQ

Ces prédictions vont ensuite être envoyées au module NLG (*Natural Language Generation*) qui va générer la réponse.

Lors de mon stage, je me suis concentré principalement sur le module FAQ pour plusieurs raisons. Tout d'abord, c'est ce scénario qui présente la moins bonne performance. C'est également aussi celui qui profiterait le plus à l'ajout d'une gestion de l'historique de conversation. De plus, les données utilisées pour le module FAQ proviennent en partie du dataset DoQa, dataset de questions-réponses potentiellement utilisable pour entraîner un modèle de ConvQA.

2. Dataset DoQA

Comme expliqué précédemment, DoQA (Campos et al. 2020) est un dataset de questions-réponses sur le domaine de la cuisine. Il contient 2 437 dialogues et 10 917 questions-réponses récupérés sur des forums de conversations répertoriant des FAQ (*Stack Exchange*⁷) comme on peut le voir dans la figure 11.

⁷. <https://cooking.stackexchange.com/>

Titre

Why add salt to the water when cooking pasta?

Asked 11 years ago Active 2 years, 4 months ago Viewed 143k times

Question

What is the effect of adding salt to the water when cooking pasta?

171

pasta food-science salt boiling

Réponse

164

The salt adds flavor, but it also helps reduce the gelation of the starch in the pasta. The starch in food is the form of microscopic grains. When these grains come into contact with water, they will trap some of it (think cornstarch in cold water), but when the water is hot they swell up like balloons and merge with each other, and you have starch gelation.



Another thing you may want to add to the pasta water is some acid (lemon or cream of tartar). Tap water in most cities is made alkaline, which increases the starch loss from the pasta to the water, making the pasta stickier.



Figure 11 - Example d'une question sur le forum de questions réponses Stack Exchange.

Chaque dialogue comporte :

- Un titre de section qui reprend le sujet global de la discussion. Il est en général similaire à la première question.
- Un paragraphe (aussi appelé passage ou contexte) contenant des informations sur le sujet du dialogue.
- Une liste de questions et leurs réponses.

Chaque pair de question/réponse contient :

- Une question généralement ouverte.
- Un extrait du contexte désigné comme la réponse et sa position à l'intérieur du paragraphe.
- La mention du *followup*, c'est-à-dire de savoir si la question actuelle fait suite à la question précédente.
- L'indication du *yesno* pour indiquer si l'on peut répondre par oui ou non à la question.

Chapitre 5. Protocole expérimental

Une des premières étapes importantes et techniques de mon stage fut d'intégrer le modèle HAM comme module de remplacement du module ConvQA dans le système de dialogue. Le but final était de pouvoir tester le système avec le modèle entraîné sur différentes données et avec différents paramètres. Pour ça, il a fallu dans un premier temps modifier le code du modèle HAM pour pouvoir l'entraîner avec les données et paramètres choisis.

1. Choix des données et paramètres du modèle

1. 1. CHOIX DES DONNEES

Le modèle de base ConvQA qui s'occupe de prédire l'extrait de réponse dans le paragraphe a été entraîné avec QuAC avant d'effectuer un fine-tuning⁸ avec le dataset DoQA. De plus, une partie des données utilisées pour trouver les paragraphes correspondants aux questions proviennent de DoQA. C'est donc avec ce dataset que j'ai entraîné le modèle.

1. 2. ADAPTATION DU MODELE POUR DOQA

Pour adapter le script du modèle fonctionnant avec le dataset DoQA, j'ai dû modifier le code pour récupérer les différentes informations selon la structure de DoQA. En effet, le système doit parcourir l'ensemble de données et récupérer les différents éléments imbriqués dans des balises dans un fichier .json. Selon les datasets, la structure varie plus ou moins et il est donc nécessaire de s'assurer que les différentes données sont bien récupérées.

1. 3. CHOIX DES PARAMETRES

Concernant les paramètres de l'entraînement je me suis particulièrement posé des questions sur le nombre de tours d'historique maximum à prendre à compte.

⁸ Le terme fine-tuning en anglais, traduisible par la notion d'ajustement fin, désigne le fait d'entraîner un modèle qui a déjà été entraîné sur une tâche, sur une autre similaire avec des paramètres beaucoup plus fins, afin d'ajuster les poids avec plus de précision.

2. Entraînements

Une fois que l'on peut entraîner notre modèle avec nos données et paramètres choisis, on peut utiliser ce modèle entraîné dans notre système de dialogue.

2. 1. ENVIRONNEMENT DU SYSTEME DE DIALOGUE

Le système de dialogue a été développé avec le langage de programmation python (version 3.6) et fonctionne à l'aide de plusieurs bibliothèques. Pour faciliter leurs installations, j'ai utilisé *Conda*⁹, un gestionnaire de package permettant de créer divers environnements python et d'y installer des bibliothèques.

Le système utilise notamment les bibliothèques *torch*¹⁰, pour l'apprentissage profond, et *transformers*¹¹.

2. 2. ENVIRONNEMENT DU MODELE HAM

Le modèle HAM lui aussi fonctionne avec la version 3.6 de python, ce qui a grandement facilité l'intégration du modèle au système de dialogue. De plus, c'est la bibliothèque *TensorFlow*¹² (version 1.8.0) qui va permettre de gérer l'entièreté du modèle HAM. *TensorFlow* permet de créer et gérer des réseaux de neurones.

2. 3. AJOUT DU MODELE AU MODULE FAQ

Pour intégrer le modèle dans le système de dialogue, il a fallu que je modifie principalement trois parties du code, visibles dans les encadrés verts de la figure 12.

⁹ <https://docs.conda.io/en/latest/>

¹⁰ <https://github.com/pytorch/pytorch>

¹¹ <https://github.com/huggingface/transformers>

¹² <https://github.com/tensorflow/tensorflow>

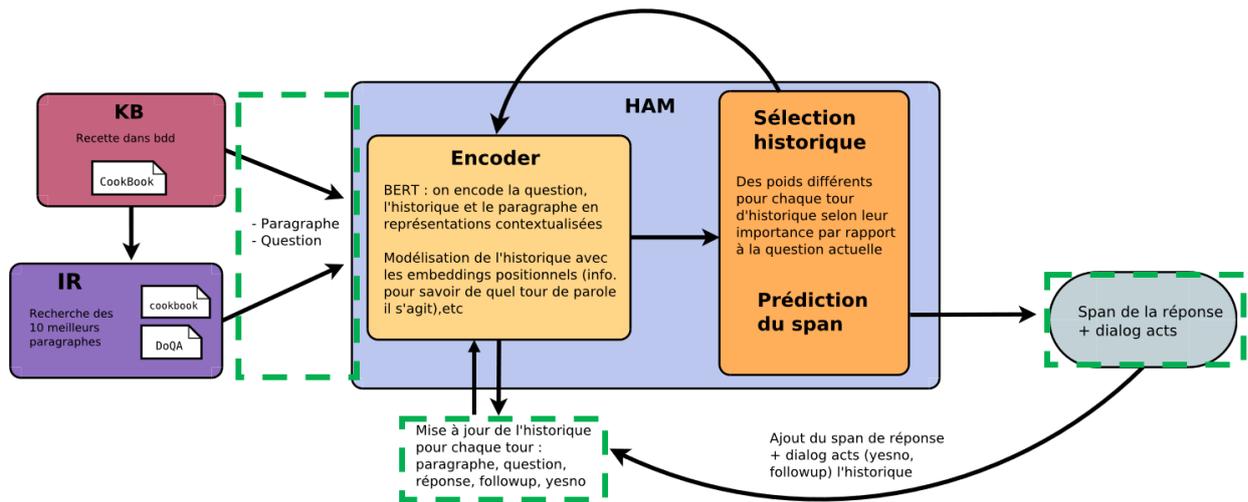


Figure 12 - Représentation des trois parties du code à modifier

Tout d'abord, il a fallu récupérer les informations d'entrée, c'est-à-dire la question, le paragraphe sélectionné et l'historique. Pour faciliter l'envoi et la récupération des données entre les modules, j'ai opté pour utiliser un fichier *pickle*¹³ qui contient l'historique et qui se met à jour au fur et à mesure de la discussion. Un exemple d'un tel fichier se trouve dans la figure 13.

```

** HISTORY FILE ****
'id': 'q1', 'context': "I would say that recipes technique sounds ok, but when I have cooked duck before I have always cooked skin side down
n a medium heat till the fat all runs out and skin is golden but this always takes longer than 3 minutes more in the 10 minute range. Maybe
is is because I use a lower heat. the recipe doesn't say how hot the pan should be. The aim is to render most of the fat out from under the
in of the duck.I usually baste with the fat whilst it is cooking then finish off under the grill (broiler - not underneath the BBQ :)), but
me times finish in the oven too. CANNOTANSWER", 'question': 'how to grill duck legs', 'answer': 'the aim is to render most of the fat out fr
under the skin of the duck', 'answer_start': 0, 'yesno': 'x', 'followup': 'y']]
** HISTORY FILE ****

```

Figure 13 - Exemple d'une sauvegarde d'historique

¹³ <https://docs.python.org/3/library/pickle.html>

Chapitre 6. Enrichir le contexte

Une des pistes d'amélioration du système fut l'enrichissement du contexte à l'aide des concepts sémantiques. Puisque les concepts et valeurs des questions du dataset DoQA n'ont pour le moment pas été annotés manuellement, j'ai pu utiliser la détection automatique des concepts du module NLU.

1. Intégration des slots au modèle

La première étape fut d'enrichir le dataset DoQA en ajoutant les slots pour chaque question. Pour cela, j'ai pu m'aider d'un script existant qui a pour fonction d'enregistrer chaque sortie du module NLU dans des fichiers séparés pour une liste de questions qu'on lui donne en entrée. La création d'un premier script m'a permis d'automatiser la récupération de chaque question du dataset pour les donner en entrée au script existant. Puis j'ai créé un dernier script qui m'a permis de récupérer les informations des concepts et valeurs de sortie du NLU. Le script les inscrit ensuite dans une nouvelle version du dataset dans laquelle les slots sont annotés. Il est possible de visualiser un extrait du dataset avec les concepts et valeurs annotés dans la figure 14.

```

"background": "Most recipes for sponge cake use a quantity of yogurt which sometimes is replaced with milk, or even soja or almond drinks. In those recipes (flour, sugar, eggs, butter/oil, yogurt/milk, some flavouring), what is the role of the dairy product? Is it about protein/fat/chs, about water, about flavour, about colour? Do the substitution for soja or almond drink make sense, or exist better ones?",

"title": "What's the role of milk or yogurt in sponge cake?",

"paragraphs": [ {

    "context": "To add to Stephanie's answer, sometimes dairy is incorporated for a variety of reasons. In some cases, like a coffee cake, sour cream or yogurt can account for a good amount of the moisture and fat in the batter. Rich dairy products like cream and sour cream usually lead to very tender baked goods, due to helping to incorporate more fat (the fact that it's already emulsified helps too). Soured dairy, like yogurt, sour cream, and buttermilk are frequently used so that their acidity will react with baking soda in the recipe for leavening. CANNOTANSWER",

    "id": "C_c824995057b542d79f639ffb0800e3e5",

    "qas": {

        "answers": {

            "answer_start": 38,

            "text": "dairy is incorporated for a variety of reasons. In some cases, like a coffee cake, sour cream or yogurt can account for a good amount of the moisture and fat in the batter.",

            "followup": "y", "id": "C_c824995057b542d79f639ffb0800e3e5_q#0",

            "question": "What's the role of milk or yogurt in sponge cake?",

            "yesno": "x",

            "slots": "ingredient milk yogurt recipe type spongecake"

        }

    }
}

```

Figure 14 - Extrait du dataset DoQA après ajout des concepts et valeurs pour chaque question.

1.1. METHODE DE RECUPERATION DES SLOTS

Je me suis intéressé à deux méthodes d'annotation des slots. La première méthode fut de simplement utiliser chaque question séparément et d'annoter ses concepts et valeurs. Cette méthode porte le nom de « Slots par phrase » dans les exemples des tableaux 2 et 3. Dans un second temps, j'ai fait une détection des concepts et valeurs du titre de la section du dialogue. Ce titre (*title* dans l'extrait du dataset) est souvent similaire à la première question d'un dialogue. J'ai choisi, pour chaque question, de concaténer les slots de la question avec les slots du titre. De cette manière, les questions qui ne reprennent pas les mots clés du dialogue vont quand même avoir les mots clés correspondants. Deux exemples de dialogue sont visibles dans le tableau 2 et le tableau 3. On peut bien visualiser que la deuxième méthode (« slots du titre + par phrase ») nous permet d'avoir plus de concepts et valeurs lorsque la question ne reprend pas explicitement les mots clés du contexte. La possibilité d'ajouter au fur et à mesure les slots de chaque question en reprenant les précédents a aussi été envisagée. Mais les slots devenaient

parfois trop nombreux et il y avait peu de différence avec la méthode 2. J'ai donc décidé de faire des entrainements avec seulement les deux premières méthodes de récupération de slots.

	Titre + Questions	Slots par phrase	Slots du titre + par phrase
T	How do you prepare fresh fennel for cooking?	category other fresh ingredient fennel	category other fresh ingredient fennel
Q	How do you prepare fresh fennel for cooking?	category other fresh ingredient fennel	category other fresh ingredient fennel
Q	What does roughly chop mean?		category other fresh ingredient fennel
Q	Is it safe to eat the bulb?		category other fresh ingredient fennel

Tableau 2 - Exemple des slots dans le dataset selon les deux techniques de récupération de slots. La mention T se réfère au "titre" du dialogue, et la mention Q renvoie aux questions des dialogues.

	Titre + Questions	Slots par phrase	Slots du titre + par phrase
T	Cracking open a pistachio	recipe type pistachio	recipe type pistachio
Q	Is there a special too available for crack- ing open a pistachio?	recipe type pistachio	recipe type pistachio
Q	I have severe arthritis and joint pains, so I don't think I could open them this way.		recipe type pistachio
Q	I have tried using regular nut crackers but they do not work well. Are there no other tools you know off to help open them?	recipe type nutcrackers	recipe type nutcrackers
Q	Do you have any other tips for opening pistachios?	recipe type pistachios event opening	recipe type pistachios event opening pistachio

Tableau 3 - Exemple des slots dans le dataset selon les deux techniques de récupération de slots. La mention T se réfère au "titre" du dialogue, et la mention Q renvoie aux questions des dialogues.

1.2. DEUX POSSIBILITES POUR ENTRAINER AVEC LES SLOTS

Une fois le nouveau dataset créé et contenant les slots pour chaque question, j'ai pu entrainer le modèle HAM avec les slots. Pour ça, j'ai dû modifier plusieurs parties du code du modèle, en particulier le module d'encodage.

Pour ça, j'ai essayé deux méthodes différentes. La première a été de simplement concaténer les embeddings de la question avec les embeddings de slots. L'autre a été de rajouter un marqueur de séparation [SEP] entre les embeddings de la question et les embeddings de slots de manière similaire à ce qui est fait entre le contexte et la question.

2. Intégration des slots au système

La dernière étape fut d'intégrer les slots dans la version du système de dialogue qui utilise le modèle HAM modifié. Pour cela, le système récupère les slots de la question détectés

par le module NLU, puis le module ConvQA les récupère pour les ajouter à la séquence d'encodage. Le système enregistre également les concepts et valeurs de chaque question dans le fichier d'historique pour pouvoir les réutiliser dans les prochains tours de dialogue.

Chapitre 7. Difficultés et problèmes

Afin de pouvoir tester les performances du modèle HAM dans le système de dialogue, il a d’abord fallu l’entraîner. Pour commencer, et également dans le but d’être sûr que le modèle fonctionne bien dans mon environnement, j’ai entraîné le modèle tel quel avec les données recommandées (dataset QuAC) et les paramètres de base.

1. *Entraînement du modèle avec Slurm*

Le modèle HAM fonctionne sur GPU (Graphic Processing Units), à la différence de ceux que l’on fait fonctionner seulement avec des CPU (Central Processing Units). J’ai pu utiliser le cluster de calcul lab-ia¹⁴ qui permet de lancer des jobs à l’aide de slurm¹⁵. Le modèle nécessite des ressources assez élevées et obtenir des résultats prend en général 90h.

2. *Problèmes rencontrés*

Outre le temps de s’habituer à utiliser Slurm et de parvenir à correctement installer mes différents environnements sur la machine distante, j’ai également passé beaucoup de temps pour parvenir à reprendre mes entraînements après un checkpoint. Pour ça, il a fallu que je me familiarise avec la version 1.8 de *TensorFlow* et que je comprenne bien comment les entraînements du modèle fonctionnent.

1. 2. PROBLEMES AVEC LES DONNEES

Le problème majeur que j’ai pu observer lors de mes tests utilisant le module FAQ du système de dialogue concerne les données utilisées. Tout d’abord, une grande partie des paragraphes récupérés par le module IR provient des données du Cookbook. Ces données ne sont pas ordonnées et ont des défauts qui influencent la performance du système. Par exemple,

¹⁴ <https://doc.lab-ia.fr/>

¹⁵ <https://slurm.schedmd.com/documentation.html>

pratiquement tous les passages commencent par « information about » suivi du mot clé qui caractérise le sujet du paragraphe. Par conséquent, le module IR a tendance à prioriser ces paragraphes et la détection de l'extrait dans le passage se situe souvent au début.

Partie 3

-

Résultats et analyse

Chapitre 8. Interprétation des résultats

L'établissement du protocole expérimental m'a permis d'aboutir à divers résultats que je vais présenter dans ce chapitre.

1. Métrique d'évaluation

Le script d'évaluation que j'ai utilisé pour évaluer l'entraînement des modèles est le script d'évaluation du dataset QuAC. Les scores calculés sont le score F1, le score HEQ (Human Equivalence for Question), le score *yesno* et le score *followup*.

Le score F1 est la métrique qui le plus d'importance lors de l'évaluation de notre système. En effet, il va correspondre à la moyenne harmonique du taux de la précision¹⁶ et du rappel¹⁷. Le score F1 humain calculé sur le dataset QuAC est de 81.1%.

$$F1 = 2 * \frac{\text{Précision} * \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

Comme le score F1 peut moins bien fonctionner pour les questions qui possèdent plusieurs réponses correctes, le score HEQ permet de directement comparer la performance du système avec celle de l'humain. Pour cela, on calcule le pourcentage auquel le score F1 du système surpasse celui du score F1 humain. Autrement dit, si pour une question donnée, la machine obtient le même F1 ou un F1 plus élevé que le F1 humain alors on rajoute 1 point. Si la machine y parvient pour un dialogue entier alors on rajoute 1 point à la sous mesure HEQ-D (d pour *dialog*).

Les *yesno* score et *followup* score permettent de calculer le pourcentage qu'a le système de prédire correctement si la question peut être répondue par oui ou non et si la question fait suite à la précédente.

¹⁶ $\text{Précision} = \frac{\text{nombre d'éléments corrects trouvés}}{\text{nombre d'éléments trouvés}}$

cf. Glossaire pour plus de détails

¹⁷ $\text{Rappel} = \frac{\text{nombre d'éléments correct trouvés}}{\text{nombre d'éléments total de la référence}}$

cf. Glossaire pour plus de détails

2. Résultats de l'entraînement

Les premiers entraînements que j'ai faits ont été produit avec le modèle HAM sans modification et avec le dataset QuAC. L'évaluation du modèle se fait à partir des données de test de QuAC et à l'aide de son script d'évaluation. J'ai pu par la suite comparer ces résultats avec les résultats des entraînements fait avec le dataset DoQA. Le Tableau 4 affiche un résumé de ces premiers résultats. Pour évaluer la performance globale du modèle, on regarde principalement le score F1. Il correspond à la moyenne harmonique du taux de la précision¹⁸ et du rappel¹⁹. Le score F1 humain calculé sur le dataset QuAC est de 81.1%. Comme l'on peut le voir, la grandeur du dataset QuAC fait que l'on obtient de bien meilleurs résultats, dont un F1 de 56.01 face à un F1 de 47 en moyenne pour DoQA. J'ai aussi comparé un entraînement en regardant les 11 derniers tours de dialogue et un autre en ne prenant en compte que les 6 derniers tours de dialogue. On remarque une légère hausse de la performance lorsqu'on prend en compte un historique de conversation plus réduit. Par défaut, le modèle HAM ne prend en compte que l'historique de réponse et non pas les questions précédentes car, comme on peut le voir dans le tableau, la performance est moins bonne. Néanmoins, nous avons tout de même trouvé plus intéressant de prendre en compte réponses et questions de l'historique.

Dataset	F1	followup	yesno	heq	dheq	Max tours d'H pris en compte	particularité
QuAC	56.01	58.77	86.47	50.54	2.6	11	Seulement réponse de l'historique
DOQA	46.53	64.75	74.21	40.0	5.75	6	x
DOQA	47.26	65.98	72.78	40.39	5.0	6	Seulement réponse de l'historique
DOQA	47.01	65.36	75.04	39.62	5.5	11	Seulement réponse de l'historique

Tableau 4 - Résultats de l'entraînement du modèle HAM avec QuAC et DoQA

¹⁸ $Précision = \frac{\text{nombre d'éléments corrects trouvés}}{\text{nombre d'éléments trouvés}}$

cf. Glossaire pour plus de détails

¹⁹ $Rappel = \frac{\text{nombre d'éléments correct trouvés}}{\text{nombre d'éléments total de la référence}}$

cf. Glossaire pour plus de détails

3. Résultats de l'entraînement avec slots

J'ai d'abord effectué les entraînements avec les deux datasets et avec des paramètres différents pour établir lesquels détiennent la meilleure performance et conviennent le mieux à notre tâche. Puis, j'ai comparé les résultats en prenant en compte les slots et valeurs des questions. Comme nous l'avons vu dans le Chapitre 5, j'ai pris en compte deux manières d'annoter les slots dans le dataset. On peut trouver un résumé des meilleurs résultats que j'ai pu obtenir pour chaque méthode dans le tableau 6. Comme anticipé, diminuer le nombre de tours maximums à étudier pour répondre à la question améliore nettement les performances du modèle. En plus de ça, la méthode qui se démarque plus est celle utilisant un embedding de séparation entre les plongements lexicaux de la question et ceux des concepts et valeurs. En effet, rajouter ces séparateurs permet au système de séparer la question de ses slots. Néanmoins, les résultats entre les deux méthodes sont peu différents, certainement dû en partie au fait que BERT est un modèle de langue entraîné avec des données de textes et non pas des données de slots.

	F1	Tours d'historique maximum pris en compte	HEQ	DHEQ
Normal sans slot	46.53	6	40.00	5.75
Slots Q (concaténation)	46.83	6	40.78	5.75
Slots Q ([sep])	46.28	6	39.79	5.0
Slots title + Slots Q (concaténation)	46.30	6	39.27	4.75
Slots title + Slots Q ([SEP])	46.92	6	39.86	4.5
Slots title + Slots Q (concaténation)	46.75	4	42.01	5.0
Slots title + Slots Q ([SEP])	48.29	4	41.53	4.25

Tableau 5 - Tableau récapitulatif des entraînements faits sur modèle HAM avec DoQA et les slots. La mention « concaténation » signifie qu'une simple concaténation des embeddings de la question et des slots a été faite. Au contraire, « [sep] » indique que l'on a utilisé la méthode de concaténation avec le marqueur de séparation entre la question et les slots. « Slots Q » se réfère aux concepts et valeurs de la question seule, « Slots title + Slots Q » renvoie aux slots de la question et de ceux du titre de la section du dialogue.

Après avoir récupéré les modèles entraînés ayant obtenus les meilleurs résultats, j'ai pu les tester dans le système de dialogue modifié utilisant le modèle HAM.

Exemple n°1

Question :

How to grill duck legs ?

Contexte :

I would say that recipes technique sounds ok, but when I have cooked duck before I have always cooked skin side down on a medium heat till the fat all runs out and skin is golden but this always takes longer than 3 minutes more in the 10 minute range.

*Maybe this is because I use a lower heat. The recipe doesn't say how hot the pan should be. **The aim is to render most of the fat out from under the skin of the duck.***

I usually baste with the fat whilst it is cooking then finish off under the grill (broiler - not underneath the BBQ :)), but sometimes finish in the oven too.

Réponse avec enrichissement du contexte à l'aide des slots

Réponse sans prendre en compte les slots

Dans l'exemple ci-dessus, j'ai dans un premier temps utilisé une version du modèle entraîné sans les slots et j'ai pu obtenir la réponse en rouge dans le contexte. Puis, j'ai posé la même question à une version du modèle et du système prenant en compte les concepts et valeurs de la question. La réponse peut-elle être vue en vert dans le passage.

Exemple n°2

Question :

How to grill duck legs ?

Contexte :

Rare duck meat is safe to eat because it does NOT contain the same risk of Salmonella as does chicken meat.

Primarily because ducks, as mentioned above, have not traditionally been raised in the same squalid conditions as "factory raised" chickens - salmonella is a disease that is primarily transmitted through dirt/dirty unclean conditions.

Now, on the other hand, as more and more ducks are being raised in industrial conditions, they are also becoming more likely to contain strains of Salmonella.

Réponses :

ducks, as mentioned above, have not traditionally been raised in the same squalid conditions as "factory raised" chickens - salmonella is a disease that is primarily transmitted through dirt/dirty unclean conditions.

Rare duck meat is safe to eat because it does NOT contain the same risk of Salmonella as does chicken meat. Primarily because ducks, as mentioned above, have not traditionally been raised in the same squalid conditions as "factory raised"

Réponse avec enrichissement du contexte à l'aide des slots

Réponse sans prendre en compte les slots

Comme dans l'exemple n°1, la réponse en rouge ne prenant pas en compte les slots semble être plus appropriée. Néanmoins, les réponses du système même sans prendre en compte les slots ne sont que rarement intéressantes et bien souvent l'extrait de la réponse se situe seulement au début du paragraphe, au dépit de la réponse correcte. On peut donc en conclure que le système de dialogue en l'état ne fonctionne pas bien avec le modèle ConvQA.

Par ailleurs, comme le système de dialogue avec HAM ne fonctionne pas très bien pour le moment, il est difficile de réellement le tester sur plusieurs tours, ce qui a fortement limité mes expérimentations. La raison pour laquelle le système de dialogue avec HAM ne fonctionne

pas bien est en grande partie liée au fait que le modèle est entraîné sur des données différentes de celles utilisées dans le système. Nous verrons dans la prochaine partie quelles pistes d'améliorations peuvent être envisagées pour permettre au module de correctement fonctionner.

Chapitre 9. Perspectives d'améliorations

Suite à l'analyse des résultats que j'ai pu obtenir, plusieurs perspectives d'améliorations sont apparues.

1. Fine-tuning du modèle avec les données du Cookbook

Alors que les résultats des entraînements semblent indiquer que la méthode est validée, les tests sur le système de dialogue indiquent un réel problème concernant les données. De plus, le module ConvQA du système de dialogue n'est pas entraîné avec les données du Cookbook. Idéalement, il faudrait donc pouvoir fine-tuner le modèle avec les données du Cookbook. Ce n'est actuellement pas possible car nous ne disposons pas d'un corpus sur ces données. Il faudrait donc en créer un, potentiellement en utilisant une simulation d'utilisateur. A ce moment, si un corpus d'apprentissage du Cookbook est disponible, il est possible d'envisager que faire un fine-tuning du modèle avec ce corpus améliorerait fortement les résultats proposés par le module FAQ du système de dialogue.

2. Amélioration de l'utilisation de DoQA

Pour améliorer les performances du système, les données nécessiteraient également quelques ajustements. En effet, seulement une petite partie du dataset Doqa est utilisée par le système de dialogue. Il paraîtrait plus intéressant d'utiliser l'entièreté des données de DoQA particulièrement parce que le dataset entier n'est pas très volumineux comparé aux autres datasets de ConvQA.

Un travail d'annotation manuel des slots et valeurs a été fait sur une petite partie du dataset. Il serait plus intéressant d'utiliser des annotations manuelles pour entraîner le modèle avec les concepts au lieu d'utiliser le module NLU pour les prédire. En effet, comme il est possible de le voir dans le tableau 7, le module NLU a une performance assez basse pour détecter correctement les slots du dataset DoQA. Ceci s'explique par le fait que beaucoup de questions présentes dans le dataset sont des questions ouvertes. Il faudrait donc idéalement pouvoir obtenir des slots annotés manuellement sur l'entièreté du dataset et utiliser ces slots pour entraîner le modèle.

Corpus	SER	Precision	Recall	F-measure
All	71.8 %	42.4 %	39.9 %	41.1 %
Database	43.6 %	61.8 %	57.7 %	59.7 %
Web	57.7 %	59.1 %	53.6 %	56.2 %
DOQA	88.7 %	28.8 %	27.5 %	28.1 %

Tableau 6 - Tableau des résultats de la détection des concepts du module NLU

3. Fine-tuning de BERT sur des données de slots

Comme on a pu le voir dans les chapitres précédents, enrichir le contexte avec les slots améliore la performance du modèle. Néanmoins, les embeddings de slots sont calculés de la même façon que les embeddings des questions. En effet, le modèle utilise un BERT pré entraîné sur des données provenant de Wikipédia, du corpus Brown²⁰ et de pairs de questions-réponses. Il y a donc une grande différence entre ce type de textes et des données de slots. Il serait par conséquent intéressant d'envisager la possibilité de fine-tuner BERT sur des données de slots.

Pour cela, il faudrait tout d'abord récupérer, en grande quantité, des slots de questions liées à la cuisine grâce au système NLU puis fine-tuner BERT avec ces slots. Cela reviendrait donc à utiliser deux modèles de langues : le modèle de langue de base de BERT et un modèle de langue de slots.

²⁰ Un des premiers et plus gros corpus en anglais (https://en.wikipedia.org/wiki/Brown_Corpus)

Conclusion

1. Bilan du travail accompli

Le domaine des systèmes ConvQA est un domaine complexe et qui pose un certain nombre de problématiques. Il reste encore beaucoup de pistes à explorer pour améliorer le système de dialogue du projet LIHLITH, certaines que je n'ai pas pu approfondir autant que je l'aurais voulu, par manque de temps ou de ressources.

J'ai pu observer que les performances du modèle HAM modifié et entraîné avec les données de DoQA sont correctes et semblent pouvoir s'améliorer avec les bons paramètres. De la même manière, nous pouvons anticiper une hausse des performances du modèle avec l'amélioration du dataset DoQA, notamment en prenant en compte des slots annotés mais aussi en agrandissant la taille générale du dataset.

2. Bilan personnel

En premier lieu, ce stage a été pour moi une manière de découvrir un milieu que je ne connaissais pas, celui de la recherche. J'ai trouvé très enrichissant de faire face aux manières de procéder et méthodes utilisées et de travailler sur un projet déjà bien abouti sur lequel plusieurs personnes ont travaillé.

Effectuer un état de l'art pendant la première étape de mon stage m'a permis de doucement m'acclimater à tous les aspects importants des systèmes de dialogue et particulièrement à la problématique de l'historique dans un système ConvQA. Je suis satisfait d'être parvenu à obtenir divers résultats suite à mes différents tests et entraînements de modèles.

Malgré les quelques difficultés que j'ai ressenties au début, particulièrement face au domaine de l'apprentissage machine, je pense avoir réussi à approfondir un certain nombre de connaissances qui me seront assurément utiles dans le futur. De la même manière, travailler sur la modification d'un système de dialogue déjà existant n'a pas été une tâche aisée, cependant je pense que c'est un exercice qui m'a été très enrichissant.

J'ai également pu approfondir des connaissances techniques, notamment au niveau de la programmation python, de l'utilisation d'un outil d'apprentissage automatique comme *TensorFlow* et de l'utilisation du système d'exploitation linux et des machines distantes. Même

s'il m'a fallu du temps pour parvenir à correctement lancer et optimiser les entraînements des modèles, ainsi qu'à me familiariser avec la gestion des *logs*²¹, j'ai énormément appris en ayant accès au service de lab-ia pour l'entraînement des modèles.

²¹ Le terme *log* en anglais se réfère à un type de fichier contenant l'historique des informations de sortie d'un système. Ici, il s'agit des sorties concernant l'entraînement des modèles.

Bibliographie

- Campos, Jon Ander, Arantxa Otegi, Aitor Soroa, Jan Deriu, Mark Cieliebak, et Eneko Agirre. 2020. « DoQA -- Accessing Domain-Specific FAQs via Conversational QA ». *arXiv:2005.01328 [cs]*, mai. <http://arxiv.org/abs/2005.01328>.
- Chen, Yu, Lingfei Wu, et Mohammed J. Zaki. 2020. « GraphFlow: Exploiting Conversation Flow with Graph Neural Networks for Conversational Machine Comprehension ». *arXiv:1908.00059 [cs]*, juillet. <http://arxiv.org/abs/1908.00059>.
- Choi, Eunsol, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, et Luke Zettlemoyer. 2018. « QuAC : Question Answering in Context ». *arXiv:1808.07036 [cs]*, août. <http://arxiv.org/abs/1808.07036>.
- Devlin, Jacob, Ming-Wei Chang, Kenton Lee, et Kristina Toutanova. 2019. « BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding ». *arXiv:1810.04805 [cs]*, mai. <http://arxiv.org/abs/1810.04805>.
- Gupta, Somil, Bhanu Pratap Singh Rawat, et Hong Yu. 2020. « Conversational Machine Comprehension: a Literature Review ». *Proceedings of the 28th International Conference on Computational Linguistics*, 2739-53. <https://doi.org/10.18653/v1/2020.coling-main.247>.
- Ohsugi, Yasuhito, Itsumi Saito, Kyosuke Nishida, Hisako Asano, et Junji Tomita. 2019. « A Simple but Effective Method to Incorporate Multi-turn Context with BERT for Conversational Machine Comprehension ». *arXiv:1905.12848 [cs]*, mai. <http://arxiv.org/abs/1905.12848>.
- Qu, Chen, Liu Yang, Minghui Qiu, Yongfeng Zhang, Cen Chen, W. Bruce Croft, et Mohit Iyyer. 2019. « Attentive History Selection for Conversational Question Answering ». *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, novembre, 1391-1400. <https://doi.org/10.1145/3357384.3357905>.
- Reddy, Siva, Danqi Chen, et Christopher D. Manning. 2019. « CoQA: A Conversational Question Answering Challenge ». *arXiv:1808.07042 [cs]*, mars. <http://arxiv.org/abs/1808.07042>.
- Yeh, Yi-Ting, et Yun-Nung Chen. 2020. « FlowDelta: Modeling Flow Information Gain in Reasoning for Conversational Machine Comprehension ». *arXiv:1908.05117 [cs]*, janvier. <http://arxiv.org/abs/1908.05117>.
- Zhang, Xin, An Yang, Sujian Li, et Yizhong Wang. 2019. « Machine Reading Comprehension: a Literature Review ». *arXiv:1907.01686 [cs]*, juin. <http://arxiv.org/abs/1907.01686>.
- Zhu, Chenguang, Michael Zeng, et Xuedong Huang. 2019. « SDNet: Contextualized Attention-based Deep Network for Conversational Question Answering ». *arXiv:1812.03593 [cs]*, janvier. <http://arxiv.org/abs/1812.03593>.

Sitographie

Anaconda - Repéré à l'adresse <https://docs.conda.io/en/latest/> (Août 2021)

Pytorch - Repéré à l'adresse <https://github.com/pytorch/pytorch> (Août 2021)

Transformers - Repéré à l'adresse <https://github.com/huggingface/transformers> (Août 2021)

Tensorflow - Repéré à l'adresse <https://github.com/tensorflow/tensorflow> (Août 2021)

Brown Corpus - Repéré à l'adresse https://en.wikipedia.org/wiki/Brown_Corpus (Août 2021)

FlowDelta - Repéré à l'adresse <https://github.com/MiuLab/FlowDelta> (Août 2021)

HAM - Repéré à l'adresse https://github.com/prdwb/attentive_history_selection (Août 2021)

Glossaire

Apprentissage machine : L'apprentissage automatique, apprentissage machine ou encore machine learning est un sous domaine de l'intelligence artificielle. Il se réfère à l'étude de machines capables d'apprendre à l'aide de données.

BERT : BERT (Bidirectional Encoder Representations from Transformers) est un modèle de langue développé par Google. Comme son nom l'indique, il a la particularité d'utiliser des Transformers, un mécanisme d'attention composé d'un encodeur qui lit le texte en entrée et d'un décodeur qui fait une prédiction. Contrairement aux modèles unidirectionnels, l'encoder ne lit pas les mots de gauche à droite ou de droite à gauche mais toute la séquence en même temps, ce qui lui permet de prendre en compte tout le contexte d'un mot, c'est-à-dire les autres mots qui l'entourent. (Devlin et al. 2019)

Fine-tuning : Le terme fine-tuning en anglais, traduisible par la notion d'ajustement fin, désigne le fait d'entraîner un modèle qui a déjà été entraîné sur une tâche, sur une autre similaire avec des paramètres beaucoup plus fins, afin d'ajuster les poids avec plus de précision.

Module : Dans le contexte d'un système de dialogue, un module est une partie du système. En informatique, il s'agit d'une unité dans une architecture plus globale.

Précision : La précision, ou plutôt exactitude, donne la proportion de réponses correctes parmi l'ensemble des réponses trouvées. C'est-à-dire que l'on regarde le nombre d'éléments corrects sur le nombre de total d'éléments.

$$\text{Précision} = \frac{\text{nombre d'éléments corrects trouvés}}{\text{nombre d'éléments trouvés}}$$

Rappel : Le rappel, ou plutôt exhaustivité, se réfère à la proportion de réponses correctes parmi l'ensemble des éléments de la référence.

$$\text{Rappel} = \frac{\text{nombre d'éléments corrects trouvés}}{\text{nombre d'éléments total de la référence}}$$

Système de dialogue : Système visant à faire communiquer un humain avec une machine par un biais textuel ou par le biais de la parole.

Token : On appelle token une unité de sens dans un texte. Cela résulte de la tokenisation, procédé qui vise à découper un texte en plus petites unités de sens, par exemple un mot ou une ponctuation.

Sigles et abréviations

BERT : Bidirectional Encoder Representations from Transformers

CNRS : Centre National de la Recherche Scientifique

ConvQA : Conversational Question Answering

CoQA : Conversational Question Answering

CPU : Central Processing Units

DoQA : Domain Question Answering

FAQ : Frequently Asked Questions

GPU : Graphic Processing Units

HAM : History Attention Mechanism

HEQ : Human Equivalence for Question

HEQ-D : Dialog Human Equivalence for Dialog

ILES : Information, Langue Ecrite et Signée

IR : Information Retrieval

LIDILEM : Linguistique et Didactique des Langues Étrangères et Maternelles

LIHLITH : Learning to Interact with Humans by Lifelong Interaction with Humans

LIMSI : Laboratoire d'Informatique pour la Mécanique et les Sciences de l'Ingénieur

LISN : Laboratoire Interdisciplinaire des Sciences du Numérique

LRI : Laboratoire de Recherche en Informatique

NLG : Natural Language Generation

NLU : Natural Language Understanding

QuAC : Question Answering in Context

SLURM : Simple Linux Utility for Resource Management

TAL : Traitement Automatique des Langues

Table des Figures

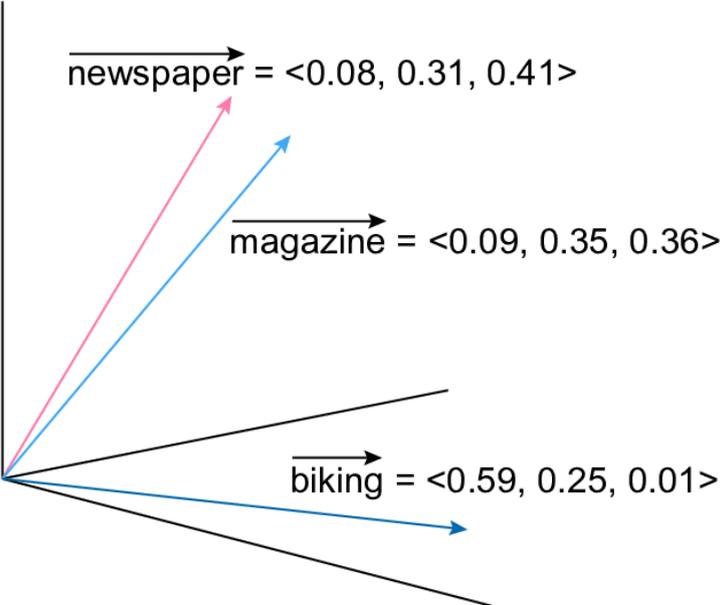
Figure 1 - Schéma de la structure générale d'un système de dialogue	13
Figure 2 - Exemple d'un dialogue utilisateur/machine nécessitant un historique.....	15
Figure 3 - Exemple d'un dialogue utilisant un modèle MRC.	16
Figure 4 - Visualisation d'un système à tour unique et d'un système à tours-multiples.....	17
Figure 5 - Représentation générique d'un modèle ConvQA (inspiré de Gupta, Rawat, et Yu 2020).....	17
Figure 6 - Schéma des embeddings utilisés par le modèle HAM (figure provenant de Qu et al. 2019).....	22
Figure 7 - Schéma global du système de dialogue	26
Figure 8 - Capture d'écran de la détection des concepts suite à la requête d'un utilisateur.....	27
Figure 9 - Capture d'écran d'une requête au système qui utilise le scénario KB.	28
Figure 10 - Capture d'écran d'une requête utilisant le module FAQ	29
Figure 11 - Exemple d'une question sur le forum de questions réponses Stack Exchange.	30
Figure 12 - Représentation des trois parties du code à modifier	33
Figure 13 - Exemple d'une sauvegarde d'historique	33
Figure 14 - Extrait du dataset DoQA après ajout des concepts et valeurs pour chaque question.	35

Table des Tableaux

Tableau 1 - Tableau reprenant cinq des modèles ConvQA présentant les meilleures caractéristiques (données provenant de (Gupta, Rawat, et Yu 2020)).....	24
Tableau 2 - Exemple des slots dans le dataset selon les deux techniques de récupération de slots. La mention T se réfère au "titre" du dialogue, et la mention Q renvoie aux questions des dialogues.....	36
Tableau 3 - Exemple des slots dans le dataset selon les deux techniques de récupération de slots. La mention T se réfère au "titre" du dialogue, et la mention Q renvoie aux questions des dialogues.....	37
Tableau 4 - Résultats de l'entraînement du modèle HAM avec QuAC et DoQA	43
Tableau 6 - Tableau récapitulatif des entraînements faits sur modèle HAM avec DoQA et les slots. La mention « concaténation » signifie qu'une simple concaténation des embeddings de la question et des slots a été faite. Au contraire, « [sep] » indique que l'on a utilisé la méthode de concaténation avec le marqueur de séparation entre la question et les slots. « Slots Q » se réfère aux concepts et valeurs de la question seule, « Slots title + Slots Q » renvoie aux slots de la question et de ceux du titre de la section du dialogue.	44
Tableau 7 - Tableau des résultats de la détection des concepts du module NLU.....	49

Table des Annexes

Annexe 1 - Visualisation en trois dimensions de trois embeddings de mots (figure provenant de (Kenter 2017))



Annexe 2 - Tableau des modèles ConvQA ayant les meilleures performances sur les datasets QuAC et DoQA. (Tableau provenant de Gupta, Rawat, et Yu 2020)

Model	Overall test F1 scores		Approaches in				Repository
	CoQA	QuAC	History Selection	History Modelling	Contextual Reasoning	Additional Training methods used (if any)	
Human Performance	88.8	81.1	N/A	N/A	N/A	N/A	N/A
RoBERTa + AT + KD (Ju et al., 2019)	90.7 (ensemble) 90.4 (single)	74.0 (ensemble) 73.5 (single)	All history Q&A (until truncated)	Appending history Q&A to the current question (truncate query if question tokens exceed)	Contextual Integration using single-turn, pre-trained LM - RoBERTa (Liu et al., 2019a)	Rationale tagging multi-task (token prediction in CoQA evidence) along with Adversarial Training (Goodfellow et al., 2015) and Knowledge Distillation (Furlanello et al., 2018).	No official code repo is given.
XLNet + Augmentation	89.0	71.2	All history Q&A (until truncated)	Appending history Q&A to the current question (truncate query if question tokens exceed)	Contextual Integration using single-turn XLNet-large (Yang et al., 2019) (24L, 1024H, 16A)	None	https://github.com/stevezheng23/xlnet_extension_tf
BERT + Answer Verification	82.8	N/A	All history Q&A (until truncated)	Appending history Q&A to the current question	Contextual Integration using single-turn BERT-Base (Devlin et al., 2019) (12L, 768H, 12A)	Additional answer verification multi-task as in (Hu et al., 2019).	https://github.com/sogou/SMRCToolkit
SDNet	79.3 (ensemble) 76.6 (single)	N/A	Last 2 Q&A	Appending history Q&A to the current question	Attention-based Reasoning with sequential model-SDNet (Zhu et al., 2018)	None	https://github.com/Microsoft/SDNet
BERT w/2-context (Ohsang et al., 2019)	78.7	64.9	Last 2 Q&A	Append each Q (or A) to the current question for every BERT model.	Concatenates contextual outputs from BERT-Base models for every history question and answer (2N+1 sequences for N turns) and runs Bi-GRU over the aggregated sequence.	Answer type (Yes/No) prediction multi-task.	No official code repo is given.
HAM (Qu et al., 2019b)	N/A	65.4	Dynamic selection policy by attending over all the previous history turns and deriving weights based on their contextual correlation with the current turn.	Concatenate context tokens with positional history answer marker embeddings corresponding to each dialog turn at a relative position from the current turn e.g. marker embedding 2 for turn 5 when the current turn is 7. Ignores history questions.	Aggregates contextual outputs from BERT-Base models for every answer (N sequences for N turns) using attention-based selection weights to obtain word-level and seq-level representations.	Dialog act (continuation and affirmation) prediction multi-task	https://github.com/prdwb/attentive_history_selection
Bert-FlowDelta (Yeh and Chen, 2019)	77.7	65.5	All history questions only	Integrate latent representations generated via contextual reasoning on the history turns (FLOW)	Each history turn is first passed through BERT whose last and second last layer outputs are input to separate FlowQA models. Reasoning is done using the Integration-Flow approach with information gain as the propagated FLOW representation.	None	https://github.com/MiuLab/FlowDelta
GraphFlow (Chen et al., 2019)	77.3	64.9	All the history Q&A	Uses a mix of all modeling techniques- encodes relative dialog turn numbers within each question embedding and appends all to the current question, then concatenates answer marker embeddings to context tokens and finally encodes the turn into N context graphs which are used for reasoning.	The encoded context graphs per turn are used as the propagated FLOW representation in the Integration-GraphFlow architecture.	Answer type (Yes/No) prediction multi-task.	https://github.com/hugochan/GraphFlow
FlowQA (Huang et al., 2018a)	75.0	64.1	All history questions only	Integrate latent representations generated via contextual reasoning on the history turns (FLOW)	Reasoning is done using the Integration-Flow approach with latent history turn representation directly as the propagated FLOW representation (instead of information gain).	Unanswerability multi-task.	https://github.com/momohuang/FlowQA
BERT+HAE (Qu et al., 2019a)	N/A	62.4	Last K history answers (max K=11)	Concatenate context tokens with history answer marker embeddings to identify if the token is present at any dialog turn. Ignores history questions.	Contextual integration using a single-turn BERT-base	None	https://github.com/prdwb/bert_hae
BIDAF++ w/2-Context (QuAC baseline) (Choi et al., 2018)	67.8	60.1	Last 2 Q&A	Encodes relative dialog turn numbers within each question embedding and appends all to the current question. Concatenates answer marker embeddings to context tokens.	Combines self-attention with BiDAF (Seo et al., 2017) followed by multi-layered BiLSTM to obtain contextualized representation.	Dialog act multi-task in QuAC.	Use implementation on http://allenai.jp.org
DrQA+PGNet (CoQA baseline) (Reddy et al., 2019)	65.1	N/A	All history Q&A	Appending history Q&A to the current question	BiLSTM based contextual integration over encoded tokens for extracting evidence (DrQA), followed by PGNet (Vinayals et al., 2015) which uses attention-based neural machine translation for abstractive answer reasoning (selecting relevant tokens).	None	https://github.com/stanfordnlp/coqa-baselines
Vamila DrQA (Chen et al., 2017)	52.6	N/A	All history Q&A	Appending history Q&A to the current question	BiLSTM based contextual integration over encoded tokens for extractive span.	None	https://github.com/facebookresearch/DrQA

Annexe 3 - Tableau des résultats de l'entraînement du modèle HAM sur 20 000 itérations. Entraînement effectué sur DoQA et ses slots (méthode avec séparateur [SEP] et slots récupérés sur le titre de la section du dialogue et sur la question. Voir Chapitre 6 pour plus de détails sur les méthodes utilisées.)

Training steps	Total loss	F1	followup	yesno	heq	dheq	Max tours d'H pris en compte
1000	2.09	38.97	54.26	78.40	36.35	2.75	4
2000	2.05	42.97	65.03	77.27	39.72	4.5	4
3000	2.32	44.35	62.64	74.23	37.77	4.75	4
4000	2.59	45.86	65.07	74.00	39.86	4.25	4
5000	3.07	43.75	64.29	75.01	37.44	3.5	4
6000	3.35	42.10	62.91	74.11	34.19	3.25	4
7000	4.07	40.06	61.87	71.58	32.07	2.75	4
8000	4.47	45.56	64.37	73.35	39.66	4.5	4
9000	4.70	43.73	64.25	73.94	36.40	4.0	4
10000	5.16	45.07	64.99	72.48	38.14	6.25	4
11000	5.15	45.81	63.88	76.63	39.25	6.0	4
12000	5.64	46.96	64.25	75.90	40.09	5.5	4
13000	5.51	44.58	65.05	69.04	37.17	4.75	4
14000	5.54	48.29	65.11	78.31	41.53	4.25	4
15000	5.64	46.74	65.80	72.07	39.56	4.5	4
16000	5.60	45.04	64.41	73.28	38.23	4.0	4
17000	5.76	44.76	63.46	72.34	38.28	5.0	4
18000	5.68	41.95	63.29	68.71	34.53	4.0	4
19000	5.92	45.06	64.71	70.86	38.60	5.25	4
20000	5.93	46.30	66.52	72.42	38.89	4.75	4

Table des matières

PREAMBULE	8
INTRODUCTION	9
LE PROJET LIHLITH.....	9
MISSIONS ET OBJECTIFS.....	10
Partie 1 - État de l'art.....	11
CHAPITRE 1. LES SYSTEMES DE DIALOGUE	12
1. MODULE DE COMPREHENSION DU LANGAGE	13
2. MODULE DE GESTION DU DIALOGUE.....	13
3. MODULE DE GENERATION DE LA REPONSE.....	14
CHAPITRE 2. L'HISTORIQUE DE DIALOGUE DANS LES SYSTEMES CONVERSATIONNELS	15
1. POURQUOI UN HISTORIQUE ?	15
2. LES MODELES CONVQA	15
3. LES DIFFERENTES METHODES	17
4. DATASETS UTILISES	20
CHAPITRE 3. ÉTUDE DE MODELES CONVQA	22
1. MODELE HAM.....	22
2. RESULTATS DES MODELES LES PLUS PERFORMANTS.....	24
Partie 2 - Contributions et expérimentations.....	25
CHAPITRE 4. PARTICULARITES DU SYSTEME DE DIALOGUE	26
1. DESCRIPTION DU SYSTEME	26
2. DATASET DOQA	29
CHAPITRE 5. PROTOCOLE EXPERIMENTAL	31
1. CHOIX DES DONNEES ET PARAMETRES DU MODELE.....	31
2. ENTRAINEMENTS.....	32
CHAPITRE 6. ENRICHIR LE CONTEXTE	34
1. INTEGRATION DES SLOTS AU MODELE	34
2. INTEGRATION DES SLOTS AU SYSTEME.....	37
CHAPITRE 7. DIFFICULTES ET PROBLEMES	39
1. ENTRAINEMENT DU MODELE AVEC SLURM	39
2. PROBLEMES RENCONTRES	39
Partie 3 - Résultats et analyse.....	41
CHAPITRE 8. INTERPRETATION DES RESULTATS	42
1. METRIQUE D'EVALUATION.....	42
2. RESULTATS DE L'ENTRAINEMENT	43
3. RESULTATS DE L'ENTRAINEMENT AVEC SLOTS	44
CHAPITRE 9. PERSPECTIVES D'AMELIORATIONS	48

1. FINE-TUNING DU MODELE AVEC LES DONNEES DU COOKBOOK	48
2. AMELIORATION DE L'UTILISATION DE DOQA	48
3. FINE-TUNING DE BERT SUR DES DONNEES DE SLOTS	49
CONCLUSION.....	50
1. BILAN DU TRAVAIL ACCOMPLI.....	50
2. BILAN PERSONNEL.....	50
BIBLIOGRAPHIE	52
SITOGRAFIE.....	53
GLOSSAIRE.....	54
SIGLES ET ABREVIATIONS	56
TABLE DES FIGURES	57
TABLE DES TABLEAUX	58
TABLE DES ANNEXES.....	59
RÉSUMÉ.....	64
ABSTRACT	64

MOTS-CLÉS : apprentissage automatique, système conversationnel, système de dialogue, plongement de mots

RÉSUMÉ

Le défi principal d'un système de dialogue est d'obtenir le dialogue le plus interactif possible et de simuler une conversation avec un humain. Pour cela, il est important que le système comprenne et prenne en compte l'entièreté du dialogue pour répondre efficacement aux questions posées par l'utilisateur. Ce mémoire rend compte de l'étude de plusieurs pistes d'amélioration d'un système de dialogue orienté tâches sur le domaine de la cuisine. Les améliorations explorées se sont en particulier focalisées sur l'intégration d'un historique de conversation et sur l'enrichissement du contexte de dialogue à l'aide de plongements de mots.

KEYWORDS : machine learning, dialogue system, chatbot, word embeddings

ABSTRACT

The main challenge with a dialogue system is to obtain an interactive discussion similarly to one with a human. Thus, it is important for the system to understand and take into account the whole dialogue in order to give correct answers. This paper reports back on the study of several experiments to improve a task-oriented chatbot working on the cooking domain. The experiments are mainly the addition of a dialogue history integration and the use of words embeddings in order to enrich the dialog context.