université
PARIS-SACLAY

POLYTECH
PARIS-SUD

Rapport TAL

Léo GALMANT
Sacha LENARTOWICZ
Maxime SAMSON

6 mai 2017

Introduction

Pour notre projet en Traitement Automatique des Langues (TAL), nous avions le choix entre trois thèmes, et nous avons choisi de créer un chatbot. Nous devions tout d'abord choisir quel type de chatbot implémenter en choisissant un domaine de compétence.

Parmi les objectifs demandés, notre programme doit être capable de converser au mieux avec un humain, de façon naturelle. De plus, les réponses formulées doivent être grammaticalement correctes. Nous avions également la tâche de permettre à notre chatbot de prendre en compte les données entrées par l'utilisateur pour formuler ses réponses.

Nous avions le choix de la méthode sur laquelle construire notre chatbot (heuristiques, motifs, mots-clefs, grammaires, réponses génériques...). Nous devions également inclure des énoncés montrant que le programme a compris et prend en compte ce que dit l'utilisateur.

1 Nos choix

Dans le contexte actuel des élections présidentielles, nous avons décidé que le centre d'intérêt de notre programme serait la politique. Nous vous laisserons le soin de découvrir si "Shabo" (nom que nous avons donné au programme) est plutôt pro-capitaliste, ou au contraire s'il possède un esprit plus communiste.

Nous avons également décidé de baser le fonctionnement sur une recherche de mots-clefs. Nous avons trouvé ceci très adapté à notre objectif car le vocabulaire que Shabo est capable de reconnaître est un vocabulaire précis, et donc facilement détectable.

Enfin, nous avons choisi de travailler en anglais pour deux raisons. La première, l'anglais est une langue plus facile à étudier que le français, bien que nous maitrisons mieux le français, notre langue natale. La deuxième raison est qu'il existe plus d'outils à notre disposition pour l'anglais.

2 Les outils utilisés

Dans l'objectif d'implémenter le plus efficacement notre programme, nous devions utiliser le langage Python, pour lequel les cours de TAL que nous avons reçus ont été d'une grande aide. En plus d'être un langage particulièrement utile pour le traitement des chaînes de caractères, il est possible, avec Python, d'inclure et d'utiliser des bibliothèques que nous allons vous présenter.

2.1 PyEnchant

PyEnchant est une librairie Python, permettant d'effectuer des vérifications de langage, basée sur la librairie Enchant. Cet outil nous a permis de vérifier une première chose (après la tokenisation de la phrase, dont nous reparlerons dans la suite). On a pu exclure les phrases composées de mots n'appartenant pas au dictionnaire anglais britannique et anglais américain. Cette vérification est faite par la fonction *is_english_sentence()*.

Pour utiliser cette bibliothèque, vous devez l'installer à l'aide de la commande suivante sous Linux : *sudo apt-get install python3-enchant*. Nous avons remarqué que cette vérification ne prenait que peu de temps (quasi-immédiate), contrairement à nos craintes, et nous avons donc choisi de la garder afin d'exclure les phrases incorrectes pour faciliter l'exploitation des données entrées.

2.2 Wordnet

Wordnet peut être décrit comme étant une base de données lexicale portant sur la langue anglaise. On retrouve cet outil dans la librairie Nltk. Pour installer Wordnet, lancez un script Python contenant *nltk.download("wordnet", "chemin_du_programme")*. Ensuite, lancez le script Python suivant : *nltk.data.path.append('chemin_du_programme')*.

Wordnet nous permet de trouver un ensemble de synonymes pour un mot donné. Nous avons remarqué qu'une seule recherche prenait environ trois secondes d'exécution. Nous avons donc tout d'abord prévu d'utiliser ce module uniquement pour permettre d'enrichir nos propres dictionnaires de mots-clefs.

3 Logique de fonctionnement

3.1 Le pré-traitement

Dans un premier temps, il a été nécessaire de pouvoir identifier les mots entrés par l'utilisateur. Pour cela, nous avons repris la "tokenisation" expérimentée lors des TP précédents, nous permettant ainsi de repérer les éléments de ponctuation par la suite. Les majuscules des mots sont également remplacées par des minuscules.

Comme expliqué dans la partie 2, nous avons vérifié que chaque token correspondait soit à un élément de ponctuation, soit à un mot appartenant au dictionnaire anglais britannique ou au dictionnaire américain. Pour les phrases incorrectes, Shabo indiquera à l'utilisateur qu'il a bien commis une erreur (réponse différente du cas où Shabo ne comprend pas la phrase).

3.2 Les affirmations

Nous traitons différemment les deux types de requêtes données par l'utilisateur : les questions et les affirmations.

Dans le cas d'une affirmation, nous utilisons un premier dictionnaire de mots-clefs associés à la gauche, et un second correspondant au vocabulaire de droite. Nous avons aussi défini, à l'aide de Wordnet, un dictionnaire de mots positifs associant à chaque mot un niveau indiquant la "force" du mot. Par exemple, "good" est de niveau 1, "great" de niveau 2 et "wonderful" de niveau 3. Nous avons fait de même pour composer un dictionnaire de mots négatifs.

Nous avons aussi cherché à reconnaître les adverbes, pouvant influencer sur la "force" de l'affirmation. Les adverbes de base sont stockés dans la liste *MULTIPLICATORS*, mais nous utilisons une liste agrémentée de synonymes toujours grâce à Wordnet, appelée *MULTIPLICATORS_FINAL*.

Nous nous servons de ces différents éléments pour définir si, dans sa globalité, une phrase affirmative va contrarier ou au contraire flatter Shabo. En fonction des différents niveaux enregistrés, Shabo va définir son humeur (*humor* dans le code) et va associer un degré de réponses parmi sept niveaux (trois négatifs, un neutre et trois positifs). L'ensemble du traitement s'effectue dans la fonction *politicalArguments()*.

Vous trouverez ici la liste des mots considérés de gauche par Shabo :

— left
— communism
— leftist
— ussr
— gulag
— gulags
— healthcare
— cuba
— socialism
— communists
— comrade
— socialists
— equality
— anarchism
— anarchists
— collectivism
— collectivists
— marx
— staline
— lenine
— stalin
— lenin
— kropotkine
— proudhon
— mao
— castro
— che

Liste des mots de droite :

— capitalism
— bourgeois
— bourgeoisie
— shareholder
— shareholders
— property
— boss
— CEO
— company
— companies
— usa
— america
— united states

Liste des mots positifs :

— good
— like
— acceptable
— valuable
— positive
— satisfactory
— fine
— great
— marvelous
— admirable
— amazing
— excellent
— exceptional
— wonderful

Liste des mots négatifs :

— bad
— wrong
— indecent
— odious
— despicable
— immoral
— hateful
— repugnant
— evil

Parmi les affirmations, Shabo a également un comportement par défaut aux phrases de salutation. De même, pour quitter le programme, vous pouvez, en plus du Ctrl+C sous Linux, dire "bye" à Shabo, et le programme se fermera.

3.3 Les questions

Pour la gestion des différentes questions, nous réutilisons les dictionnaires liés aux vocabulaires de droite et de gauche.

Shabo est capable de reconnaître si la question posée est une question personnelle. Le programme va alors répondre, suivant les réponses prédéfinies associées au mot-clef reconnu. Vous pouvez demander à Shabo son nom, son âge, s'il est humain et s'il va bien.

Pour chacun des mots de la question, Shabo vérifie s'il est dans le dictionnaire de gauche ou de droite. Dans le cas où le mot est dans le dictionnaire de réponses, il affiche la réponse associée, contenant une définition du mot et l'avis que porte le chatbot sur celui-ci. Sinon, si le mot appartient au vocabulaire de gauche ou de droite, on incrémente un score, indiquant si la phrase est plus composée de mots de gauche ou de droite. Si aucun mot n'appartient au dictionnaire de réponse, alors une réponse par défaut, basée sur le score de la question, est

envoyée à l'utilisateur. Cette partie est traitée dans la fonction *politicalQuestion()*.

Voici une liste de mots pour lesquels Shabo est capable de répondre convenablement :

- left
- communism
- leftist
- ussr
- gulag
- gulags
- healthcare
- cuba
- socialism
- communists
- socialists
- equality
- anarchism
- anarchists
- collectivism
- collectivists
- marx
- lenin
- kropotkine
- che
- capitalism
- bourgeois
- bourgeoisie
- shareholder
- shareholders
- property
- CEO
- company
- companies
- usa

4 Améliorations envisagées

Comme dans de nombreux domaines, le projet consistant en la création d'un chatbot n'est jamais terminé, et ceci est vrai même lorsque l'on parle d'équipes importantes. C'est donc particulièrement le cas pour notre groupe composé de seulement trois étudiants.

4.1 Les améliorations sur les affirmations

Concernant cette première partie, nous envisageons dans un premier temps de modifier notre façon de répondre. Pour le moment, Shabo va considérer une sorte de moyenne sur la totalité de la phrase, et répondre en conséquence. Nous prévoyons de pouvoir diviser la phrase en plusieurs parties, et ainsi de pouvoir répondre pour chacun des points mentionnés par l'utilisateur. De plus, nous espérons pouvoir permettre à notre chatbot de réagir sur les mots-clefs eux-mêmes, et pas seulement sur leur orientation droite/gauche.

Nous avons commencé à définir un vocabulaire suffisamment large pour pouvoir tester Shabo. Cependant, nous avons conscience que ce vocabulaire ne représente qu'une partie des affirmations politiques dont nous avons potentiellement envie de parler avec Shabo. Un enrichissement du vocabulaire s'avère donc nécessaire.

4.2 Les améliorations sur les questions

Afin de ne pas seulement donner une définition des différents mots-clefs reconnaissables, nous imaginons pouvoir différencier les différents types de questions, entre les "yes or no questions" et les questions ouvertes. On pourrait également adapter, pour les questions ouvertes, le type de réponse en fonction du mot interrogatif en début de phrase.

Comme pour les affirmations, nous pouvons améliorer Shabo en continuant notre travail sur l'enrichissement du vocabulaire, tout en restant dans le contexte politique. Il en va de même pour les questions personnelles.

4.3 Autres améliorations

Actuellement, les noms propres ne sont pas reconnus comme tels. Or, cela peut être utile, car l'utilisateur peut avoir envie de demander ce que Shabo pense de tel ou tel acteur politique. Il faudra pour cela ne pas supprimer la majuscule lors de la tokenisation, ou alors ajouter un marqueur indiquant que le mot est un nom propre.

Pour rendre le chatbot plus humain, nous aurions également pu déterminer, pour la même phrase entrée, plusieurs réponses équivalentes. Il aurait ensuite été aisé de sélectionner une réponse, de façon aléatoire et en éliminant les réponses déjà renvoyées.

Nous imaginons également ne plus renvoyer des réponses toutes faites, mais de permettre à Shabo, dans certains cas, de pouvoir créer ses réponses respectant une structure donnée.

Enfin, nous n'enregistrons pas d'information portant sur les données entrées par l'utilisateur. On envisage de retenir des informations personnelles, comme le nom, le sexe de l'utilisateur, mais aussi son avis sur certaines questions, pouvant modifier le ton utilisé par Shabo pour répondre (plus ou moins agressif par exemple).

5 Répartition des tâches

Pour ce projet, nous avons commencé par travailler tous les trois ensemble, afin de choisir les lignes directrices sur lesquelles nous nous sommes orienté par la suite. Ainsi, nous avons pris, par exemple, la décision de vérifier si une phrase est correcte, de commencer à vérifier si la phrase de l'utilisateur est une affirmation ou une question, et nous avons pris le choix de nous orienter sur une recherche des mots-clefs.

Pour la deuxième moitié du projet, nous avons réparti les tâches comme suivant : Léo GALMANT s'est occupé de la gestion des affirmations, Maxime SAMSON des différents types de questions, et Sacha LENARTOWICZ de la rédaction de ce rapport, tout en continuant de suggérer certains choix pour améliorer Shabo. La rédaction du rapport, imposée en LaTeX, fut l'occasion de découvrir ce langage que nous ne connaissions que de nom.

Conclusion

Le projet que nous avons choisi nous a permis de mettre en application les différents TP réalisés en cours de TAL. Cela nous a aussi permis de travailler en Python, et de nous rendre compte des avantages que fournissait ce langage pour travailler sur le langage.

Bien que notre chatbot soit, de loin, moins perfectionné que ceux qui commencent à envahir notre quotidien, nous avons constaté les difficultés que consistent sa mise en place. Nous avons aussi pu étudier les différentes techniques utilisables pour l'implémentation d'un tel outil.