Université Mohammed V
Faculté des Sciences
Rabat

Master Ingénierie de Données et Développement Logiciel

-Temps Aménagé-

Module: NLP

NLP Project

Question answering[4]

Réalisé par :

YASSINE LBASRI
ZIZAN Bouchra

Année universitaire 2020-2021

Chatbot

Définition

Un chatbot est un logiciel conversationnel qui communique avec les utilisateurs dans des langues naturelles écrites ou parlées. L'agent fournit une interface conversationnelle qui permet aux utilisateurs de saisir une entrée dans différents formats, tels que du texte, de la voix, des documents Web, etc.

Les types de chatbot

- **Les chatbots basés sur des règles**

Les chatbots basés sur des règles sont également appelés bots à arbre de décision. Comme son nom l'indique, ils utilisent une série de règles définies. Ces règles sont à la base des types de problèmes que le chatbot connaît et pour lesquels il peut apporter des solutions.

- **les chatbots IA**

En comparaison, les chatbots IA qui utilisent l'apprentissage automatique comprennent le contexte et l'intention d'une question avant de formuler une réponse.

Ces chatbots génèrent leurs propres réponses à des questions plus complexes en utilisant des réponses en langage naturel. Plus vous utilisez et entrainez ces bots, plus ils apprennent et mieux ils fonctionnent avec l'utilisateur.

- **Choisir entre les bots basés sur des règles et les bots IA**

Jusqu'à il y a dix ans, la seule option que les gens avaient pour contacter une entreprise était d'appeler ou d'envoyer un e-mail à leur équipe de service client. Désormais, les entreprises proposent une équipe de chat pour offrir un meilleur service client 24 heures sur 24. Selon une étude commandée par Facebook par Nielsen, 56% des personnes préféreraient envoyer un message plutôt que d'appeler le service client, et c'est là que les bots entrent en jeu.

Les bots révolutionnent la façon dont les entreprises interagissent avec leurs clients. Il y a dix ans, les bots étaient considérés comme une mode technologique passagère. Cependant, ce débat est désormais clos, car de grandes entreprises comme Amazon, Microsoft, Facebook et d'autres ont commencé à déployer des robots dans presque tous les domaines de leur activité. Le nouveau débat qui se prépare dans la communauté des bots concerne le choix entre les bots basés sur des règles ou les bots IA. Lequel choisir ? Quel est le meilleur? Telles sont les questions que se posent les chefs d'entreprise qui ont l'intention d'utiliser des bots dans leurs organisations. De nombreux facteurs contribuent à l'efficacité des robots pour différentes applications, et la compréhension de ces facteurs peut aider les entreprises à prendre des décisions éclairées quant au choix entre les robots basés sur des règles et les robots IA.

La création et le déploiement de bots font désormais partie des listes de tâches de la plupart des entreprises, s'ils ne sont pas déjà déployés. Néanmoins, la plupart ne savent pas s'ils doivent utiliser des robots basés sur des règles ou des robots IA. Évaluons les avantages et les inconvénients de chacun.

Avantages des bots basés sur des règles

Les robots basés sur des règles peuvent répondre à des questions en fonction d'un ensemble prédéfini de règles qui y sont intégrées. L'ensemble de règles peut varier considérablement en complexité. Construire de tels bots basés sur des règles est beaucoup plus simple que de construire des bots IA. Les bots basés sur des règles sont généralement plus rapides à entraîner. Les bots sont construits sur une base conditionnelle si/alors, ce qui les rend plus simples à former. Les robots basés sur des règles peuvent prendre des mesures en fonction du résultat des instructions conditionnelles. La formation facile des robots basés sur des règles réduit simultanément les coûts de mise en œuvre. Les robots basés sur des règles sont hautement responsables et sécurisés. Ces robots ne peuvent pas apprendre par eux-mêmes et fourniront les réponses que les entreprises souhaitent qu'ils fournissent. Étant donné que les

robots basés sur des règles ne peuvent pas apprendre par eux-mêmes, cela garantit qu'ils fourniront un service client cohérent. Les robots basés sur des règles peuvent transmettre la conversation de manière professionnelle à un agent humain si le client demande quelque chose qui est absent de la base de données. La pratique consistant à transférer la conversation à l'agent humain garantit qu'aucune information inutile n'est transmise au client.

Une approche basée sur des règles permet également une implémentation plus rapide des bots. Contrairement aux robots IA, les robots basés sur des règles n'ont pas besoin d'attendre des années pour collecter des données pouvant être analysées par des algorithmes afin de comprendre les problèmes des clients et de fournir des solutions. Les robots basés sur des règles peuvent être facilement mis en œuvre en y intégrant des scénarios connus et leurs sorties. Ces robots peuvent ensuite être intégrés à plus de données en fonction de nouveaux modèles de conversation issus de nouvelles interactions avec les clients. Bien que les robots basés sur des règles présentent de nombreux avantages, leurs limites ne peuvent être ignorées.

Inconvénients des bots basés sur des règles

Le problème avec les robots basés sur des règles prédéfinies est qu'ils doivent être intégrés à des règles pour effectuer toutes les tâches petites à complexes. Si quelque chose qui sort de la base de données leur parvient, les robots basés sur des règles transmettent la conversation aux humains. Cela signifie que les robots basés sur des règles ne peuvent pas fonctionner de manière autonome ; ils ont besoin d'une intervention humaine à un moment donné.

Une autre limitation des robots basés sur des règles est celle de la communication personnalisée. Les chatbots peuvent servir différentes personnes parlant différentes langues. De plus, non seulement les langues, mais le mode de communication varient également d'une personne à l'autre. Par exemple, pour réserver un vol pour Paris, une

personne peut dire : « Je veux réserver un vol pour Paris » et une autre peut dire « J'ai besoin d'un billet pour Paris ». Les deux déclarations signifient la même chose, mais si le bot basé sur des règles est incapable de comprendre cela, il transmettra la conversation à un humain, ce qui peut frustrer le client.

Les robots basés sur des règles peuvent être intégrés avec des informations provenant de modèles de conversation au fil du temps. Néanmoins, il devient difficile pour les développeurs d'intégrer tous les scénarios possibles dans des bots basés sur des règles. Bien que les robots basés sur des règles puissent être rapidement mis en œuvre, ils sont difficiles à maintenir après un certain temps.

Avantages des robots IA

Les robots d'IA sont des robots d'auto-apprentissage qui sont programmés avec le traitement du langage naturel (NLP) et l'apprentissage automatique. Il faut beaucoup de temps pour former et construire un bot AI au départ. Cependant, les robots IA peuvent économiser beaucoup de temps et d'argent à long terme. L'utilisation de robots IA fonctionne bien avec les entreprises qui disposent de beaucoup de données, car elles peuvent auto-apprendre à partir des données. La capacité d'auto-apprentissage des robots IA permet d'économiser de l'argent, car contrairement aux robots basés sur des règles, ils n'ont pas besoin d'être mis à jour après un certain intervalle de temps. Les robots IA peuvent être programmés pour comprendre différentes langues et peuvent relever les défis de communication personnalisés auxquels sont confrontés les robots basés sur des règles.

Grâce au deep learning, les robots IA peuvent apprendre à lire les émotions d'un client. Ces robots peuvent interagir avec les clients en fonction de leur humeur. Par exemple, une startup basée en Chine, Emotibot, aide à développer des chatbots capables de détecter l'humeur actuelle du client et de réagir en conséquence. Grâce à un apprentissage constant, les robots d'IA peuvent aider à fournir un service client personnalisé pour améliorer l'engagement

client. Étant donné que les robots IA peuvent gérer les requêtes des clients de bout en bout sans qu'aucune interaction humaine ne soit requise, ils peuvent être déployés pour un service client 24 heures sur 24.

Inconvénients des robots IA

L'IA peut rendre les chatbots intelligents, mais elle ne peut pas leur faire comprendre le contexte des interactions humaines. Par exemple, les humains peuvent changer leur mode de communication en fonction de qui ils communiquent. S'ils communiquent avec de jeunes enfants, ils utilisent des mots plus simples et des phrases plus courtes. De plus, lorsque les employés humains communiquent avec les clients, ils utilisent un ton plus formel. Étant donné que les robots ne peuvent pas comprendre le contexte humain, ils communiquent avec tout le monde de la même manière, quel que soit l'âge ou le sexe. La capacité d'auto-apprentissage des robots IA peut sembler utile aux entreprises, mais elle peut parfois causer des problèmes. Les robots IA ne possèdent pas une qualité de prise de décision précise et peuvent donc apprendre quelque chose qu'ils ne sont pas censés apprendre.

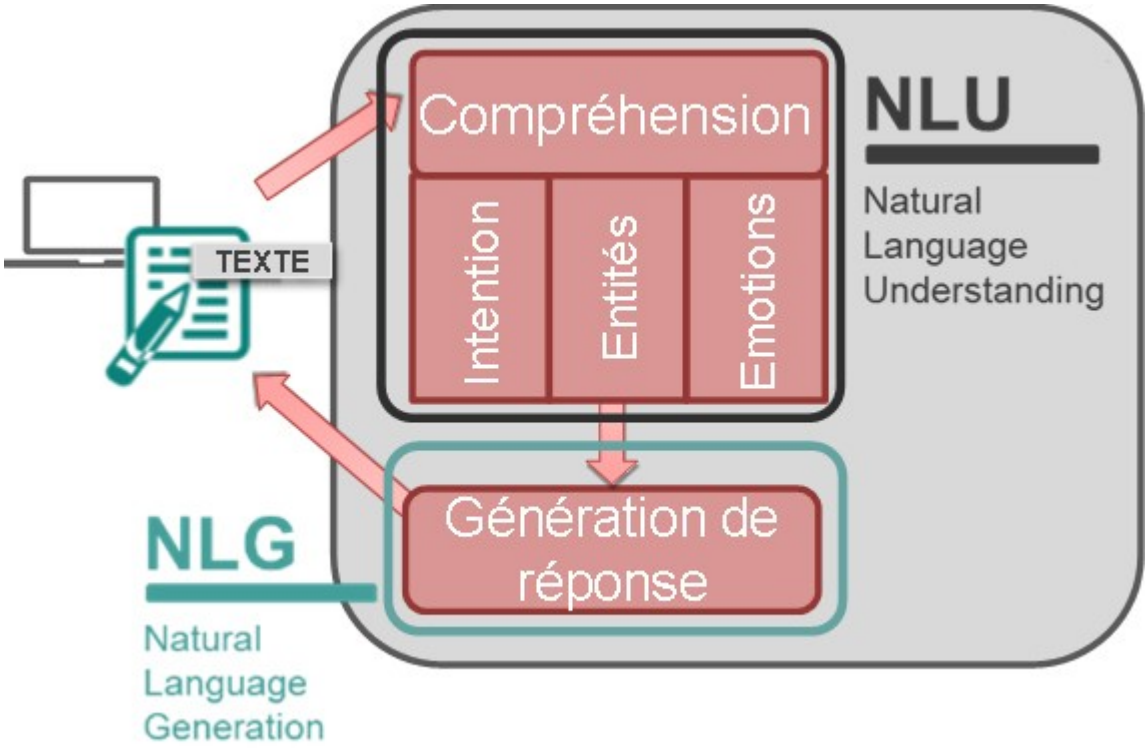
Ces avantages et inconvénients peuvent aider les entreprises à décider d'utiliser des robots basés sur des règles ou des robots IA, mais seulement dans une certaine mesure. Il existe de nombreux autres facteurs que les entreprises doivent prendre en compte avant de mettre en œuvre des chatbots dans leur entreprise. Les robots basés sur des règles et les robots IA ont tous deux leurs propres avantages et inconvénients, et les deux peuvent être utiles à leur manière. Un service client amélioré est roi lorsqu'il s'agit de la croissance d'une entreprise. Comprendre comment différents bots amélioreront leur service client les aide finalement à choisir le bot le mieux adapté à leur entreprise.

Les concepts du NLP

Le NLP (Natural Language Processing), ou « Traitement du Langage Naturel ») est une branche de l'informatique axée sur le développement de systèmes permettant aux ordinateurs de communiquer avec des personnes en utilisant le langage de tous les jours.

Typiquement, ces systèmes peuvent répondre à des questions, extraire des informations, analyser des sentiments, traduire, analyser des contextes ou encore résumer des textes.

Le système de conversation intelligent est le socle sur lequel est construit tout Chatbot. Il permet de comprendre les demandes des utilisateurs et de leur répondre de manière pertinente. Ce type de système est souvent construit au-dessus d'un algorithme de compréhension et de catégorisation. Focalisons-nous maintenant sur les différents éléments de traitement du langage : le NLG et le NLU.



Le NLG (Natural Language Generation) permet l'écriture automatique de texte de qualité similaire à celui écrit par un humain à partir de données structurées. Nous n'entrerons pas plus dans les détails ici, car cette brique ne fait pas partie de la solution Rasa.

Le NLU (Natural Language Understanding) est la sous-entité du NLP axée sur la compréhension de phrases parlées ou écrites. Le système effectue un « mapping » de langage naturel en une représentation intention/entités, de la même façon qu'une classification. C'est une façon d'extraire de l'information d'un texte écrit.

Les intentions sont des classes d'énoncés qui correspondent souvent au verbe utilisé par l'utilisateur pour écrire l'action qu'il souhaite faire. Par exemple :

- « Je veux réserver un hôtel sur New York pour le prochain Noël » ==> l'intention ici est « réservation d'une chambre d'hôtel »
- « Peux-tu me divertir en me racontant une blague ? » ==> l'intention ici est « divertir »
- « Où en est ma demande de remboursement des lunettes de mon fils ? » ==> l'intention ici est « suivi de dossier »
- Les entités sont des sous-unités de la phrase écrite contenant des informations clés utilisables dans les applications, elles correspondent à tout ce qui accompagne les intentions. Dans les exemples précédents, les entités sont respectivement :
- « New York", « prochain Noël »
- « blague »
- « remboursement », « lunettes », « fils »

Les frameworks chatbot :

Les frameworks Chatbot sont des Frameworks logiciels qui fournissent un ensemble prédéfini de fonctions qui résument les complexités de la construction d'un chatbot, comme le moteur NLP. Raj décrit les cadres de chatbot suivants [3] :
QnA Maker : un cadre basé sur le cloud fourni par Microsoft qui permet le développement d'un chatbot Q&A simple basé sur des FAQ, des URL et des documents structurés.
Dialogflow : un cadre basé sur le cloud populaire fourni par Google qui est simple à utiliser et permet l'intégration avec de nombreuses plates-formes.

IBM Watson : est un système basé sur l'IA qui traite le langage naturel pour faciliter le développement d'applications d'intelligence artificielle et de chatbots.
Rasa NLU & Core : un framework open-source fourni pour l'environnement de développement Python. C'est une boîte à outils puissante avec une courbe d'apprentissage abrupte.
Wit.ai - un cadre basé sur le cloud fourni par Facebook qui est similaire à Dialogflow mais pas aussi riche en fonctionnalités. Il fonctionne mieux lorsqu'il est intégré à Facebook Messenger.

Luis.ai : un cadre basé sur le cloud fourni par Microsoft qui a des fonctionnalités similaires à Dialogflow et Wit.ai.
Botkit.ai : similaire à Rasa en ce qu'il s'agit essentiellement d'une bibliothèque de programmation utilisant Javascript, mais offre une interface graphique.

	Qna Maker	DialogFlow	Rasa	Wit.ui	IBM Watson	Botkit.ai
Compagnie	**Microsoft**	**Google**	**RASA**	**Facebook**	**IBM**	**Botkit**
Payé / gratuit	**Gratuit**	**Gratuit**	**Gratuit**	**Gratuit**	**Gratuit** (jusqu'à 10K requêtes/mois)	**Gratuit**
Facilité d'utilisation	**Oui**	**Oui**	**Non**	**Oui**	**Oui**	**Non**
Intégration prête à l'emploi	**Oui**	**Oui**	**Non**	**Oui**	**Oui**	**Oui**
Open source	**Non**	**Non**	**Oui**	**Non**	**Non**	**Non**
Popularité	**Faible**	**Élevé**	**Élevé**	**Élevé**	**Faible**	**Faible**
Basé sur le Web	**Oui**	**Oui**	**Non**	**Oui**	**Oui**	**Non**
Langage framework	**C#**	**JavaScript**	**Python**	**Node.js**	**Python**	**JavaScript**
Support NLP	**_**	**Oui**	**Oui**	**_**	**Oui**	**Non**
Type de moteur	**_**	**NLP**	**NLU**	**NLU**	**NLP**	**_**

Intégration	**Azure**	**Facebook Messenger, Skype, Google Assistant, and Amazon Alexa**	**Facebook Messenger, Skype, Google Assistant, and Amazon Alexa**	**Facebook**	**Voice agent, Slack, Facebook Messenger, Wordpress, Custom APIs, ets.**	**Facebook, Microsoft, IBM Watson, Slack, Telegram, etc.**
Base sur le cloud	**Oui**	**Oui**	**Non**	**Oui**	**Oui**	**Non**
Hébergement	**Azure**	**Cloud GOOGLE**	**_**	**Inconnu**	**Inconnu**	**_**

Implémentation :

NLTK[1]

Le principal problème avec les données textuelles est qu'elles sont toutes au format texte (strings). Cependant, les algorithmes d'apprentissage automatique ont besoin d'une sorte de vecteur de caractéristiques numériques pour effectuer la tâche. Ainsi, avant de commencer un projet de PNL, nous devons le pré-traiter pour le rendre idéal pour le travail. Le pré-traitement d'un texte de base comprend :

- La conversion de l'ensemble du texte **en majuscules ou en minuscules**, afin que l'algorithme ne traite pas les mêmes mots dans différents cas comme des mots différents
- **Tokenization :** La tokenisation est juste le terme utilisé pour décrire le processus de conversion des chaînes de texte normales en une liste de tokens, c'est-à-dire des mots que nous voulons vraiment. Le tokenizer de

phrases peut être utilisé pour trouver la liste des phrases et le tokenizer de mots peut être utilisé pour trouver la liste des mots dans les chaînes.

- **Removing Noise :** Supprimer le bruit, c'est-à-dire tout ce qui ne se trouve pas dans un numéro ou une lettre standard.
- Removing Stop words : Suppression des mots d'arrêt. Parfois, certains mots très communs qui semblent peu utiles pour aider à sélectionner les documents correspondant à un besoin de l'utilisateur sont entièrement exclus du vocabulaire. Ces mots sont appelés "mots d'arrêt
- **Stemming:** Le "stem" est le processus qui consiste à réduire les mots infléchis (ou parfois dérivés) à leur tige, leur base ou leur racine - généralement une forme de mot écrit. Exemple : si nous devions trouver la racine des mots suivants : “Stems”, “Stemming”, “Stemmed”, “and Stemtization”, le résultat serait un seul mot "stem".
- **Lemmatization:** La lemmatisation est une légère variante du mot "stemming ". La différence majeure entre les deux est que le stemming peut souvent créer des mots inexistants, alors que les lemmes sont de purs mots. Ainsi, votre racine, c'est-à-dire le mot avec lequel vous vous retrouvez, n'est pas quelque chose que vous pouvez simplement chercher dans un dictionnaire, mais vous pouvez chercher un lemme. Des exemples de lemmatisation sont que "run" est une forme de base pour des mots comme "running" ou "ran" ou que les mots "better" et "good" sont dans le même lemme, donc ils sont considérés comme identiques.

CAMeL tools

CAMeL Tools is a suite of Arabic natural language processing tools developed by the CAMeL Lab at New York University Abu Dhabi.[2]

Installation

You will need Python 3.6 and above (64-bit) as well as the Rust compiler installed.

Install using pip

```
pip install camel-tools

# or run the following if you already have camel_tools installed
pip install camel-tools --upgrade
```

Orthographic Normalization

It is common for Arabic speakers to use shortcuts when typing Arabic text.
For example, the different variants of the letter alef ('إ' ,'أ' ,'آ' ,'ا') may be typed as just 'ا'. Some of these substitutions can just be the result of typos. The presence of these variations can cause data sparsity and are usually normalized to a single form. Orthographic normalization is the process of converting letter variants or visually similar letters into a single form.

We provide a collection of orthographic normalization functions in CAMeL Tools. The example below demonstrates a few of them.

```
[ ] from camel_tools.utils.normalize import normalize_alef_maksura_ar
    from camel_tools.utils.normalize import normalize_alef_ar
    from camel_tools.utils.normalize import normalize_teh_marbuta_ar

    sentence = "هل ذهبت إلى المكتبة؟"
    print(sentence)

    # Normalize alef variants to 'ا'
    sent_norm = normalize_alef_ar(sentence)
    print(sent_norm)

    # Normalize alef maksura 'ى' to yeh 'ي'
    sent_norm = normalize_alef_maksura_ar(sent_norm)
    print(sent_norm)

    # Normalize teh marbuta 'ة' to heh 'ه'
    sent_norm = normalize_teh_marbuta_ar(sent_norm)
    print(sent_norm)
```

```
هل ذهبت إلى المكتبة؟
هل ذهبت الى المكتبة؟
هل ذهبت الي المكتبة؟
هل ذهبت الي المكتبه؟
```

Word Tokenization

Some CAMeL Tools components expect text to be pretokenized by whitespace and punctuation. This means that an input sentence should be an array of words and punctuation instead of a single string. This is to preserve the alignment of inputs and outputs.

Python does provide the `split()` method to tokenize words by whitespace but it doesn't seperate punctuation from words. For example:

```
[ ]  sentence = "هَلْ ذَهَبْتَ إِلَى المَكْتَبَةِ؟"
     print(sentence)

     sent_split = sentence.split()
     print(sent_split)
```

```
هَلْ ذَهَبْتَ إِلَى المَكْتَبَةِ؟
['هَلْ', 'ذَهَبْتَ', 'إِلَى', 'المَكْتَبَةِ؟']
```

Chatbot arabe : défis du traitement du langage naturel

Tout ce qui précède crée des défis pour le traitement du Arabic natural language processing (NLP). La première étape de tout algorithme de traitement du langage naturel consiste à donner un sens à la langue, c'est-à-dire à analyser les phrases en unités de sens discrètes. Cette tâche est officiellement appelée la tokenisation de la langue car chaque unité de sens discrète est appelée un token.

Plus la langue est systématique et ordonnée, plus il est facile de la symboliser.

Les mêmes défis qui rendent l'arabe difficile à apprendre pour les humains signifient que l'arabe est difficile à symboliser par rapport à la plupart des autres langues courantes.

Avant de pouvoir comprendre la signification des dernières avancées, nous devons d'abord comprendre comment un modèle de langage pour la PNL a été précédemment créé.

```
print("بوت: سميتي موروبوت , سولني فاي حاجة جات فبالك و غادي نحاول نجاوبك!")

while(flag==True):
    user_response = input()
    user_response=user_response.lower()
    if(user_response!='بسلامة'):
        if(user_response=='شكرا' or user_response=='شكرا جزيلا' ):
            flag=False
            print("العفو صديقي .. مرحبا في اي وقت")
        else:
            if(greeting(user_response)!=None):
                print("موروبوت :"+greeting(user_response))
            else:
                print("موروبوت: ",end="")
                print(response(user_response))
                sent_tokens.remove(user_response)
    else:
        flag=False
        print("بوت: بسلامة ! طهلا فراسك صديقي..")
```

Référence

https://www.nltk.org[1]

https://camel-tools.readthedocs.io/en/latest/index.html[2]

https://botpress.com/arabic-chatbot[3]

https://colab.research.google.com/github/FreeBirdsCrew/AI_ChatBot_Python/blob/master/Contextual%20Chatbot%20-%20NLP%20and%20Tensorflow.ipynb#scrollTo=gzkU4N0O_q9b[4]