Interprétation linguistique de requêtes pour un moteur de questions réponses grand public

Application industrielle

Michel Plu* — **Johannes Heinecke***

Orange Labs
2 avenue Pierre Marzin
22307 Lannion cedex
{michel.plu,johannes.heinecke}@orange-ftgroup.com

RÉSUMÉ. *Cet article décrit l'utilisation d'une plateforme de traitement automatique des langues naturelles pour le développement d'une fonction de réponses à des questions dans un moteur de recherche. Cette plateforme est utilisée pour faire une interprétation linguistique des requêtes. L'intérêt de cette approche est triple. Premièrement elle permet d'identifier uniquement les requêtes qui correspondent à des questions factuelles pour lesquelles le moteur a une réponse précise. Deuxièmement, elle reconnait ces questions quelque soit leur forme linguistique y compris avec des erreurs. Troisièmement, elle désambiguïse certains mots de la requête. Une évaluation a pu montré que l'interprétation linguistique de la requête a permis la production d'une réponse à environ quatre fois plus de requêtes avec un taux de précision supérieur à 98% par rapport à une approche plus classique d'indexation de requêtes.*

ABSTRACT. *In this article we describe the use of Natural Language Processing platform to interpred user queries to be fed into a question-answering system. The advantage of this system is threefold: first, we are able to identify only those requests which correspond to factual questions for which the engine has a precise answer; second, error correction is semantically controlled, in order to avoid bad or missing answers due to typos and third the language processing is used to disambiguate ambiguous word forms. With this interpretation we are able to identify and respond to four times as many queries as without interpretation with an accuracy rate $> 98\%$.*

MOTS-CLÉS : *moteur de recherche, moteur de questions réponses, traitement automatique des langues naturelles, web sémantique*

KEYWORDS: *Search engine, question answering, computational linguistics, semantic web*

1. Introduction

Une nouvelle tendance des moteurs de recherche sur le web est d'enrichir leur liste réponses en répondant directement aux questions posées dans les requêtes des utilisateurs. Par exemple à une requête comme « population de la région bretagne », un tel moteur affiche en première réponse « 3 120 288 habitants » (cf. figure 1) alors qu'un moteur de recherche plus classique ne fournirait qu'une liste de documents.

Figure 1. *Response à une question*

L'utilité d'une telle fonction appelée par la suite moteur de questions-réponses est évidente. En trouvant directement la réponse à sa question, l'utilisateur gagne du temps et est encouragé à poser d'autres questions. En effet, sans cette fonction l'utilisateur est censé parcourir les documents ou leurs extraits proposés dans la liste réponses pour éventuellement trouver la réponse attendue. Ce parcours de documents est le plus souvent fastidieux et inconfortable pour l'utilisateur qui doit chercher lui même la réponse. Cette pénibilité est encore plus grande lorsque le terminal est un téléphone mobile.

Le développement d'un moteur de questions réponses comporte de nombreuses difficultés. Lorsque celui-ci utilise une base de données pour répondre aux questions, il faut tout d'abord transformer une requête constituée de mots clés ou parfois d'une phrase en langage naturel en une requête formelle compréhensible par le système de gestion de base de données utilisé. Ensuite, pour répondre le plus précisément possible à un maximum de réponses, il faut d'une part disposer d'un large ensemble de connaissances et d'autre part être capable de reconnaître le plus possible de formes de requêtes correspondant aux réponses que l'on peut produire.

Cet article présente le développement d'un tel moteur de questions réponses pour un moteur de recherche web grand public accessible depuis trois portails internet français[1]. La section suivante commence par décrire l'architecture du moteur. Ensuite, la section 3 explique comment un outil de traitement automatique de langues naturelles permet d'identifier et d'analyser précisément les requêtes correspondant aux réponses que l'on sait produire tout en supportant de multiples formes de requêtes possibles pour une même question. Après une évaluation en section 4, la section 5 conclut cet article et liste quelques perspectives d'améliorations dans le cadre d'une vision plus générale d'une interface utilisateur sur le web sémantique.

1. *http://www.orange.fr*, *http//lemoteur.fr* et *http//voila.fr*

2. Architecture du moteur questions-réponses

L'architecture globale du moteur de question réponse qui est interrogé est présentée dans la figure 2. L'utilisateur saisit d'abord sa requête textuelle via un navigateur web. Ce texte est ensuite analysé par un interpréteur de requêtes. Cette interprétation est produite au format XML et est transformée en requête SQL avec une feuille de style écrite en XSLT. La requête SQL est alors soumise à un système de gestion de base de données. Les réponses sont ensuite transformées pour être insérées dans le document HTML correspondant à la page de résultat du moteur de recherche.

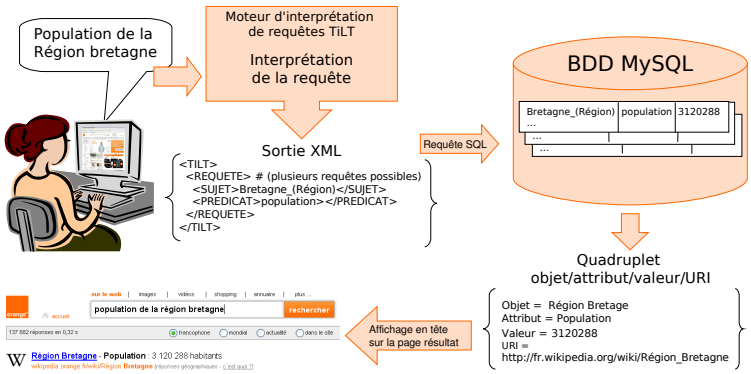

Figure 2. *Architecture du moteur questions-réponsesn*

La base de données interrogée contient une extraction des valeurs de propriétés définies dans les « infobox » des pages de la version française de l'encyclopédie en ligne Wikipedia[2]. Aujourd'hui, seules les pages correspondant à des lieux et à des personnes ont été utilisées. La base contient ainsi plus de 85 000 lieux avec plus de 163 propriétés et 75 000 personnes avec 17 propriétés.

3. Analyse linguistique de la requête

Comme nous l'avons vu précédemment, pour être capable de répondre à une question, le moteur doit pouvoir transformer la requête textuelle de l'utilisateur dans un langage d'interrogation de base de données (SQL). Pour cela, il faut tout d'abord pouvoir identifier quelles sont les mots de la requête qui correspondent à des valeurs possibles de certain champs de la base de données relationnelle interrogée. Pour reconnaître ces mots, il faut être capable de gérer leurs formes multiples ; par exemple au singulier ou pluriel et au masculin ou au féminin pour les adjectifs. Ainsi dans les requêtes « devise française » et « président français », on doit reconnaître le même sujet « France ».

2. *http://fr.wikipedia.org*

De nombreuses formes linguistiques peuvent désigner une même requête. Par exemple, les requêtes « population de la France », « population française », « nombre d'habitants en France » « combien d'habitants y a-t-il en France ? » ou « combien sommes nous en France ? » donnent la même réponse. Afin de simplifier les traitements d'analyses, toutes ces formes sont traitées de la même manière en tant que « sens » sans devoir toutes les distinguer. Cela serait trop complexe et ingérable pour faire simplement des évolutions.

Tout ceci se complique encore lorsque l'on accepte des erreurs de frappe dans la requête et que l'on tente de les corriger. En remplaçant juste un caractère par un autre, il est si facile de transformer complètement le sens d'une requête que le risque de répondre à côté est grand. Il faut alors vérifier des contraintes sémantiques entre les corrections possibles et les mots reconnus dans la requête. Par exemple si la requête « site de cabrel » ne nécessite pas de correction, la requête « maire de cabrel » mérite d'être corrigé en « maire de caurel », Caurel étant le nom d'une ville dans les Côtes d'Armor et d'une ville dans la Marne.

Gérer la sémantique de la requête permet aussi de désambiguïser le sens des mots et ainsi de répondre plus précisément. Par exemple le mot « Bretagne » peut désigner évidemment la région mais aussi deux villes dont l'une est dans l'Indre et l'autre dans le territoire de Belfort. Mais si la requête « population bretagne » est ambigüe, la requête « population de la ville bretagne » exclut la région et la requête « population de la ville bretagne dans l'indre » ne l'est plus du tout. Le module d'interprétation de la requête doit donc tenir compte du contexte des mots et sélectionner les entités désignées par un mot (ici le mot « bretagne ») qui s'accordent sémantiquement avec les autres. Dans le domaine de la géographie cette ambigüité est très fréquente. Sur un log de 10 millions de requêtes réelles nous avons identifié que parmi les requêtes relatives à des lieux et correspondant à des questions auxquelles le moteur de question réponses sait répondre, environ 25% des mots avaient plusieurs significations.

3.1. *Développement de l'interpréteur de requêtes avec la plate-forme Tilt*

Nous avons développé cet interpréteur de requête en utilisant la plate-forme pour le traitement automatique des langues Tilt développée dans notre laboratoire à Orange Labs. Tilt à une architecture modulaire qui permet d'activer uniquement les traitement nécessaires et utilise des données linguistiques telles que des lexiques et des grammaires (Heinecke *et al.,* 2008). Les modules sont les suivants :

– Un système de correction qui permet de corriger des mots d'une requête absents du lexique en fonction de son contexte dans la phrase ainsi qu'en fonction d'un vocabulaire cible spécifié pour l'application.

– Un analyseur grammatical permet d'étiqueter des parties du texte analysé. Les grammaires supportées s'inscrivent dans le cadre des grammaires hors contexte (Chomsky, 1956) Pour notre application, les règles définies correspondent plutôt à une grammaire de contraintes sémantiques qu'à une grammaire syntaxique plus communément utilisée. Pour cela, les règles ne portent pas sur les catégories syntaxiques

normales du français, mais plutôt sur des fonctions applicatives que nous nommons par la suite catégories lexicales.

Nous avons défini dans notre lexique quatre catégories lexicales applicatives :

– « SUJET » associée dans notre cas aux formes désignant un lieu ou une personne

– « PRED » associée auxformes désignant des propriétés connues d'un sujet

– « CREUX » associée à des mots dits creux autorisés dans une requête

– « PLEIN » associée à des mots interdits dans une requête

Ce lexique applicatif est étendu par un lexique des formes fléchies du français. Ces formes sont catégorisées automatiquement comme mots pleins ou mots creux en fonction de traits syntaxiques définis dans ce lexique. Cette extension du lexique applicatif permet de reconnaître les mots corrects de la langue française et d'éviter leur correction vers un mot de notre lexique applicatif. La propriété et le sujet exprimé dans la requête traitée sont identifiés à partir des règles de grammaires exploités par l'analyseur de Tilt. Par exemple les règles :

```
CREUX PRED ⇒ PRED          | PRED SUJET ⇒ OK
PRED CREUX ⇒ PRED          | OK CREUX ⇒ OK
CREUX CREUX ⇒ CREUX        |
```

signifient qu'une requête peut commencer par plusieurs mots creux et un prédicat suivis éventuellement de plusieurs mots creux. Ce sujet peut être suivi par plusieurs mots creux.

La présence d'un mot avec la catégorie mot plein n'est traitée par aucune règle et ne permet donc pas de produire un état OK correspondant à la reconnaissance d'une requête acceptable pour l'interpréteur. Afin d'aller plus loin dans la désambiguïsation sémantique des sujets évoqués dans la requête, la catégorie sujet du lexique peut être spécialisée par des sous-catégories sémantiques. Par exemple une sous catégorie de sujet est nommée *Place*, elle-même spécialisée par *Department*, *Town*, *Region*. Ceci permet alors d'écrire des règles spécifiques pour des sujets de certains types. Par exemple on peut alors écrire des règles suivantes :

```
PRED DECL-TOWN ⇒ PRED-TOWN     | PRED-TOWN TOWN ⇒ OK
PRED-TOWN CREUX ⇒ PRED-TOWN    |
```

Les catégories lexicales de la forme PRED-*NNN*, sont associées dans le lexique à des mots qui précisent la catégorie du sujet. Ainsi alors que le mot « Bretagne » corresponde à un nom de région ou de des villes une dans l'Indre et une autre dans le Territoire de Belfort, la requête « population de la ville de bretagne » pourra sélectionner pour sujet, un sujet ayant pour forme lexical le mot « bretagne » mais uniquement avec une catégorie *Town*, et non la région de même nom, car le mot « ville » est associé à une catégorie lexicale DECL-TOWN.

D'autres mécanismes dits de contraintes d'accord sont disponibles dans Tilt pour affiner cette désambiguïsation sémantique. Là aussi, ces mécanismes généralement utilisés pour tester des contraintes d'accord syntaxique ont été utilisés pour faire des accords plutôt sémantiques. Par exemple une contrainte d'accord d'égalité de trait sur

un trait nommé « department » et associé dans le lexique applicatif à des sujets de la catégorie *Town* et *Department* permet de désambiguïser la requête « population de la ville de bretagne dans l'indre ». L'accord sur l'égalité de trait « department » permet de sélectionner le sujet correspondant à la ville de l'Indre et pas celle dans le Territoire de Belfort.

3.2. *Génération des données linguistiques*

Afin de faciliter la définition et l'évolution de l'interpréteur de requêtes en fonction des données disponibles dans la base de données à interroger, nous utilisons une base de connaissances pour générer automatiquement différentes données linguistiques. Cette base de connaissances est construite à partir de données issues du projet DBpedia[3] (Plu *et al.,* 2011). Elle permet de générer automatiquement le lexique reliant les différentes formes textuelles à des identifiants de sujets et de prédicats et permet d'écrire automatiquement les règles de grammaire spécifiques à chaque catégorie sémantique en fonction d'une hiérarchie des classes associées aux sujets gérés par l'interpréteur linguistiques.

4. Evaluation

Afin d'évaluer l'apport de l'interpréteur linguistique nous avons comparé deux versions du moteur de question réponses : l'une avec l'interpréteur linguistique et une autre version recherchant dans un index textuel la requête. L'index de cette base de données contient des formes exactes de requêtes composées en concaténant chaque nom de prédicat avec chaque nom de sujet et chaque nom de sujet avec chaque nom de prédicat. A chacune de ces requêtes correspond la réponse à la question. Cette comparaison a été faite pour plus de 10 millions de requêtes (sans suppression des doublons) réellement soumises au moteur de recherche web généraliste d'un des principaux portails internet français[4]. Compte tenu du nombre important de requêtes nous nous sommes restreint aux réponses relatives à des lieux, domaine le plus ambigüe. Nous avons aussi seulement évalué les propositions de réponses faites par les deux versions du moteur. Nous ne pouvons donc pas évaluer le rappel de manière absolue de manière relative en comparant les deux versions du moteur. Nous avons ainsi évalué que l'interprétation linguistique de la requête a permis la production d'une réponse à environ quatre fois plus de requêtes avec un taux de précision supérieur à 98%.

L'interpréteur linguistique a une meilleure précision car il est capable de gérer l'ambiguïté de mots dans la requête. Il ne produit donc pas une requête avec l'ensemble des interprétations possibles d'un même nom de sujet lorsque la requête permet de le désambiguïser. Nos tests ont aussi montré que la correction de requête permet d'augmenter significativement de 9% le taux des requêtes correctement identifiées

3. *http://www.dbpedia.org*
4. *http://www.orange.fr*

par rapport à une recherche exacte dans l'index de la base de données. De plus l'usage de la plateforme Tilt a permis de satisfaire les exigences en temps de traitements de requêtes. Ainsi toutes les requêtes soumises au moteur de recherche web de plus de deux mots sont systématiquement soumises a notre moteur de question réponses qui peut traiter plus de 300 requêtes à la seconde sur un seul serveur[5].

Après une étude bibliographique de l'état de l'art, il s'avère que notre système est globalement assez conforme aux nombreux systèmes décrits dans la littérature académique. En effet l'application du traitement automatique des langues naturelles à l'interrogation de base de données a déjà une longue histoire (Androutsopoulos *et al.,* 1995). Néanmoins nous n'avons pas trouvé de description de système déployé pour un grand nombre d'utilisateurs et annonçant des performances en temps de requêtes similaires au notre. (Popescu *et al.,* 2003), annonce que le temps de traitement d'une requête est d'environ 6 secondes. Le système décrit dans (Cimiano *et al.,* 2007) ne résout pas l'ambiguïté de la requête avant d'interroger leur base de connaissance. La résolution est censée se faire par celle-ci. Nous n'avons pas choisi cette approche en désambiguïsant la requête lors de son analyse afin de favoriser les performances.

Nous n'avons pas trouvé non plus de systèmes explicitement tolérants aux erreurs dans les requêtes alors que nous avons clairement identifié son importance. Or la correction d'erreur est souvent sources d'imprécision dans les résultats fournis et pose donc de nombreux problème à résoudre. Les deux systèmes cités précédemment excluent toutes requêtes comprenant des mots hors du vocabulaire de la base de données (ou base de connaissances pour le deuxième). Or nous avons vu dans nos exemples qu'il y a de nombreuses formes linguistiques possibles pour désigner aussi bien les propriétés que les sujets et d'autre part que certains mots sont porteurs de sens pour désambiguïser la requête et ne doivent pas être ignorés comme de simples mots souvent appelés creux (*stopwords*).

Comparativement aux moteurs de recherche du web les plus couramment utilisés, notre moteur de question réponses est capable de répondre à beaucoup plus de questions dans les domaines ciblés grâce à l'interprétation linguistique des requêtes qui supporte de multiples formulations possibles d'une même question y compris celles comprenant certaines erreurs.

5. Conclusions et perspectives

Nous avons présenté dans cet article comment utiliser un interpréteur de requêtes en langage naturel pour augmenter le rappel et la précision d'un moteur de questions réponses. L'interpréteur de requêtes permet de gérer les multiples formes possibles d'une requête en langue naturelle, y compris les erreurs de frappe, et la transformer en une requête formelle pouvant être envoyée à une base de données. Nous avons aussi expliqué comment la prise en compte de sous catégories sémantiques et les contraintes d'accord sémantique permettent aussi de désambiguïser certaines requêtes. Une parti-

5. PC sous Linux, 2,8GHz, 12GO mémoire vive

cularité notoire de ces travaux est leur mise en opération dans le moteur de recherche de trois portails internet français cité ci-dessus.

Néanmoins, nous considérons cette réalisation comme un embryon de notre vision d'un futur moteur de recherche pour le web sémantique aujourd'hui en plein développement (Shadbolt *et al.,* 2006). En effet, pour interroger ces milliards d'informations reliées entre elles, il est illusoire d'imaginer que les utilisateurs formulent leur recherche en SPARQL[6] ou puissent parcourir ces volumes d'informations avec des interfaces de navigation. Une interface en langue naturelle est une solution comme le montrent des tests utilisateurs (Kauffman *et al.,* 2007). Le défi reste alors la volumétrie et l'ambiguïté croissante de la langue lorsqu'on aborde de nombreux domaines. Pour aborder ce défi nous allons perfectionner l'automatisation de l'évolution d'un moteur de questions réponses en fonction de l'augmentation des connaissances collectées sur le web sémantique sous différents formats. Pour cela nous allons notamment étudier des mécanismes pour acquérir automatiquement les relations entre les mots et les syntagmes d'une langue avec les classes et propriétés des ontologies utilisées.

6. Remerciements

Nous nous remercions les collègues d'Orange Labs qui ont contribué aux travaux décrits dans cette article : Emilie Guimier De Neef, Christine Chardenon, Arnaud Debeurme et les équipes de développement du moteur de recherche *http://orange.fr*.

7. Bibliographie

Androutsopoulos I., Ritchie G. D., Thanisch P., « Natural Languages Interfaces to Databases - An introduction », *Natural Langugae Engineering*, vol. 1:1, p. 29-81, 1995.

Chomsky N., « Three models for the description of language », *IRE Transactions on Information Theory*, vol. 2, p. 113-124, 1956.

Cimiano P., Haase P., Heizmann J., Mantel M., Orakel: A portable natural language interface to knowledge bases, Technical report, AIFB, Universität Karlsruhe, 2007.

Heinecke J., Smits G., Chardenon C., Guimier De Neef E., Maillebuau E., Boualem M., « Tilt : plateforme pour le traitement automatique des langues naturelles », *TAL*, vol. 49:2, p. 17-41, 2008.

Kauffman E., Bernstein A., « How useful are natural language interfaces to the semantic web for casual users », *Lecture Notes in Computer Science*, vol. 4825, p. 281-294, 2007.

Plu M., Heinecke J., « Un moteur de questions réponses d'une base de connaissances », *11ème Conférence sur l'Extraction et la Gestion des Connaissances*, p. 593-598, 2011.

Popescu A.-M., Etzioni O., Kautz H., « Towards a theory of natural language interfaces to databases », *8th conference on Intelligent User Interfaces*, p. 149-157, 2003.

Shadbolt N., Hall W., Berners-Lee T., « The Semantic Web Revisited », *IEEE Intelligent Systems*, vol. 21:3, p. 96-101, 2006.

6. *http://www.w3.org/TR/rdf-sparql-query/*