

# Génération de requêtes pour les systèmes de Q/R avec un modèle d'apprentissage statistique

Nicolas Usunier<sup>1</sup>, Massih-Reza Amini<sup>1</sup>, Patrick Gallinari<sup>1</sup>, et  
Brigitte Grau<sup>2</sup>

<sup>1</sup> Laboratoire d'Informatique de Paris 6  
8 rue du capitaine Scott, 75015 Paris  
{usunier, amini, gallinari}@poleia.lip6.fr

<sup>2</sup> LIMSI-CNRS  
BP 133, 91403 Orsay cedex  
Brigitte.Grau@limsi.fr

**Résumé.** Nous présentons dans cet article un modèle d'apprentissage permettant de générer des requêtes booléennes pour les systèmes de question/réponse qui peut-être employé aussi bien sur des collections fermées que sur l'internet.

Nous proposons deux approches pour cette tâche de génération. Dans la première, le modèle est entraîné à faire de la classification de requêtes. La seconde approche est nouvelle et consiste à entraîner le modèle à comparer les requêtes entre elles afin de les ordonner. Dans ce cadre, le modèle est appris pour prendre en compte les dépendances entre les requêtes générées à partir d'une même question.

Nous évaluons les deux approches proposées sur les questions de l'évaluation TREC 11 et sur le corpus de documents *Aquaint*, utilisé lors de cette même évaluation.

## 1 Introduction

Les systèmes de question/réponse ont pour but de trouver la réponse exacte à une question formulée en langage naturel dans un grand corpus de documents. Actuellement, le cadre principal de la recherche sur ces systèmes est défini par les évaluations annuelles TREC organisées par NIST<sup>3</sup>.

Les systèmes développés dans ce cadre utilisent une étape de Recherche d'Information (RI), dont le but est de déterminer les passages susceptibles de contenir la réponse dans la collection de documents considérée. Seuls ces passages seront traités dans les étapes ultérieures pour rechercher une réponse précise à la question. Cette étape de pré-sélection est cruciale, il s'agit de trouver les passages pertinents susceptibles de contenir la réponse.

La sélection des passages commence par la génération d'une requête à partir

---

<sup>3</sup> <http://trec.nist.gov/>

de la question. Cette tâche de génération de requête est fortement dépendante du corpus de documents considérés. En effet, les systèmes considérant plusieurs sources de documents ([?], [?], [?], [?]) utilisent des heuristiques de génération de requête totalement différentes suivant qu'ils utilisent un corpus fermé du type *Aquaint* ou le web. En particulier, la sélection des mots de la requête et l'établissement de seuils sur le nombre de documents renvoyés par le moteur de recherche sont spécifiques d'une collection donnée.

Aujourd'hui, migrer un système de question/réponse vers de nouvelles bases de documents nécessite l'écriture de nouvelles heuristiques de génération de requêtes adaptées à ces bases. Pour augmenter la généralité des systèmes, il est important de disposer de méthodes de sélection aussi indépendantes que possible de la collection de documents utilisée.

Dans cet article nous présentons un modèle de génération de requêtes booléennes générique, utilisant l'apprentissage automatique pour s'adapter au corpus de documents considéré.

Le principe de la génération de requêtes utilisé dans notre approche est classique. À partir de la question utilisateur, on réalise une pré-sélection des mots informatifs en éliminant pronoms, déterminants, articles et auxiliaires. Si un verbe est présent, on ajoute les formes verbales irrégulières. Les requêtes candidates correspondent à tous les sous ensembles de mots que l'on peut construire à partir de ces mots sélectionnés. La sélection doit permettre de choisir la "meilleure" de ces requêtes. Le critère de choix sera pour nous la précision de la requête sur le corpus. L'apprentissage doit permettre cette sélection. Les systèmes proposés prendront comme entrée un codage de la requête sous une forme synthétique qui intègre des informations provenant de l'analyse de la question fournie par le système du LIMSI [?].

Nous étudierons deux approches distinctes pour effectuer l'apprentissage de notre modèle. La première est basée sur la classification de requêtes. Avec cette méthode, nous cherchons à trouver parmi un ensemble de requêtes potentielles générées pour une question, celle qui permettra de ramener les passages contenant la réponse à une question donnée, et qui sera donc considérée comme la *meilleure*. Nous présentons également une nouvelle approche dite de comparaison qui permet d'exploiter les liens existant entre des requêtes issues d'une même question, afin de sélectionner celle qui semble la plus intéressante.

Nous évaluons notre modèle de génération de requête et les deux approches d'apprentissage utilisées sur l'ensemble de questions de l'évaluation TREC 2002, en utilisant la collection de documents *Aquaint*<sup>4</sup> issue de cette évaluation.

L'article sera organisé de la façon suivante. La section ?? décrit les travaux ex-

---

<sup>4</sup> <http://www.ic-arda.org/InfoExploit/aquaint/>

istants dans le cadre des questions/réponses en terme de génération de requête, ainsi qu'en terme d'apprentissage de fonctions de sélection. La section ?? décrit les caractéristiques utilisées pour représenter les requêtes dans notre modèle. La section ?? présente les deux approches d'apprentissage utilisées. La section ??, expose les expériences réalisées ainsi que leurs résultats. La conclusion est donnée à la section ??.

## 2 État de l'art

### 2.1 Génération de requête dans les systèmes de question/réponse

Comme le soulignent [?], la difficulté de la tâche de question/réponse dans une collection de documents donnée dépend essentiellement de la relation qui existe entre la forme de la réponse et celle de la question correspondante. Ainsi, les méthodes de génération de requêtes sont fortement dépendantes du corpus de documents utilisé.

**Sur le Web** ([?], [?], [?], [?]) utilisent le moteur de recherche Google. La collection étant alors extrêmement riche, il s'agit de restreindre le nombre de documents renvoyés. Les méthodes de génération consistent en une suite d'heuristiques générant des requêtes de moins en moins contraintes. Moins la requête est contrainte plus elle renverra de documents. Pour ([?], [?]), les requêtes les plus contraintes consistent à des recherches Google, sous forme d'expression régulière, d'une reformulation sous forme affirmative de la question. Les requêtes les moins contraintes sont une conjonction des mots non vides de la question. Les heuristiques sont appliquées les unes après les autres, jusqu'à ce qu'il y ait une requête pour laquelle le moteur de recherche renvoie un nombre de documents supérieur à 0.

**Sur un corpus fermé** Il y a principalement deux types de requêtes utilisées dans ce cas, les requêtes booléennes ([?], [?], [?]), et vectorielles ([?], [?]). D'une façon générale, il y a deux types d'heuristiques utilisées pour générer les requêtes à partir de la question initiale. Elles concernent respectivement la sélection de mots-clés de la question et l'enrichissement de la requête.

Pour les requêtes booléennes, les deux types d'heuristiques sont appliqués à la suite, avec toujours des seuils sur le nombre de documents renvoyés par le moteur de recherche. C'est ce type de requête qui a montré les meilleurs résultats à l'évaluation TREC 2002 [?]. [?] et [?] décrivent des suites d'heuristiques pour sélectionner les mots-clés selon leur fonction dans la question et leur étiquette morpho-syntaxique. [?] décrivent des heuristiques pour sélectionner les reformulations en utilisant uniquement l'outil linguistique WordNet [?]. [?] étudient la combinaison de WordNet avec une méthode d'extension locale de requête utilisant la technique Local Context Analysis (LCA) [?] sur le Web pour trouver les reformulations.

[?] utilisent des requêtes vectorielles préalablement enrichie en effectuant un LCA sur un corpus encyclopédique. Ils sélectionnent alors les 100 meilleurs documents renvoyés par leur moteur de recherche. [?] génère des requêtes booléennes sur le web et vectorielles sur le corpus Aquaint. Ces requêtes vectorielles sont composées de tous les mots de la question, avec un poids particulier attribué au focus. Les 750 premiers passages de 20 lignes renvoyés par le moteur de recherche sont conservés pour les traitements ultérieurs. Le problème des requêtes vectorielles est qu'il n'existe pas de moteur de recherche puissant sur le Web permettant d'effectuer de telles requêtes, l'avantage est qu'elles permettent d'ordonner les documents.

Un troisième type d'interrogation de moteur de recherche est réalisé par [?] et [?]. Il consiste à prendre en compte dans l'interrogation le type attendu de la réponse. Cette approche a d'excellentes performances en terme de précision pour la tâche de question/réponse sur des corpus fermés, mais ne peut là encore être utilisée sur le Web pour lequel il n'existe pas de moteur disposant de cette fonctionnalité.

**L'apprentissage pour la génération de requête** À notre connaissance cette approche n'a été étudiée pour l'instant que par [?]. Ils proposent une méthode de génération de requêtes sur le Web anglophone prenant en compte le type de la question. Pour cela, ils génèrent une requête à partir les mots non-vides de la question à laquelle ils rajoutent une chaîne de caractères en fonction du type de la question (par exemple pour une question du type *définition* ils rajoutent à la requête des expressions comme "is a" ou "refers to"). Ils proposent alors une méthode d'apprentissage pour trouver ces expressions par type de questions. Par contre, dans leur modèle la sélection des mots de la question dans la requête n'est pas apprise.

Par rapport aux méthodes précédentes, l'originalité de notre système est de posséder une caractérisation vectorielle des requêtes, indépendante du corpus étudié. Les requêtes générées sont booléennes, afin de pouvoir être appliquées sur le Web comme sur des corpus plus restreints. À partir de cette caractérisation, une sélection de la meilleure requête, parmi celles qu'il est possible de générer à partir d'une question donnée, est effectuée grâce à un algorithme d'apprentissage.

## 2.2 Fonctions de sélection dans les systèmes de question/réponse

La sélection du meilleur élément ou des  $K$  meilleurs éléments d'un ensemble d'éléments connus a priori est un problème que l'on retrouve à différentes étapes de traitement dans les systèmes de question réponse. Suivant les étapes, les éléments considérés seront des passages de texte, des phrases ou des réponses possibles.

**Dans les systèmes issus de la communauté du traitement automatique des langues** Ces fonctions sont généralement des sommes pondérées d'un petit nombre de caractéristiques dont les poids sont attribués à la main. [?] utilise une fonction de sélection des passages en sortie du moteur de recherche à partir d'une fonction heuristique calculée grâce à un système de reconnaissance de termes et de variantes de termes nommé FastR [?]. Une fois les passages sélectionnés, ([?], [?], [?]) utilisent une fonction de scoring permettant de sélectionner les meilleures phrases, de ces phrases sont extraites des réponses candidates, une autre fonction est alors utilisée pour déterminer la meilleure réponse parmi les candidates.

**Les systèmes issus de la communauté d'apprentissage** Les problèmes de sélection dans ces systèmes sont généralement ramenés à des problèmes de classification bi-classe "pertinent/non-pertinent". On associe la classe "pertinent" aux éléments qui doivent être sélectionnés, et la classe "non-pertinent" aux autres éléments.

Ainsi, [?] extraient au sein des passages des réponses potentielles puis utilise la classification pour sélectionner les couples (passages, réponse possibles), un score étant associé à chaque couple. Une même réponse pourra ainsi avoir plusieurs scores selon le passage. [?], [?] entraînent des classifieurs pour la tâche de sélection de réponse parmi celles qui sont extraites, leur permettant d'attribuer un score aux réponses trouvées par le système. Pour une question, une seule réponse sera choisie qui est celle de plus fort score. [?] proposent une alternative à la classification bi-classes pour la sélection de réponse. Dans la classification bi-classes, le score des réponses potentielles à une question est calculé indépendamment pour les différentes réponses. Leur méthode permet de prendre en compte l'ensemble des réponses possibles pour calculer ce score. Pour cela, étant données  $C$  réponses possibles  $A_1, \dots, A_C$  à une question, ils utilisent un vecteur représentant l'ensemble de ces  $C$  réponses qui sera traité par un classifieur à  $C$  classes entraîné à produire la classe  $c \in \{1, \dots, C\}$  si et seulement si la bonne réponse est la réponse  $A_c$ . Cette nouvelle méthode a été utilisée lors de l'évaluation TREC 2003 par [?] pour la sélection de réponses fournies par divers extracteurs, et apporte une amélioration générale du système de l'ordre de 9% par rapport à une approche de classification bi-classe.

La différence majeure entre notre approche de *comparaison* et celle *re-ranking* de [?] est la possibilité de comparer les requêtes deux à deux (cf section ??). Cela permet: **(a)** de considérer des informations plus riches liées à la pertinence relative des requêtes prises deux à deux, **(b)** de pouvoir considérer des ensembles d'éléments à ordonner qui soient de taille variable et non plus de taille fixée comme dans [?].

Cette dernière considération est importante car les questions possèdent un nombre de mots variables. Le nombre de requêtes qu'il est possible de générer à partir d'une question est donc lui aussi variable. Enfin, l'approche par com-

paraison paraît intuitivement plus intéressante pour ordonner des éléments que l'approche classification.

### 3 Représentation des requêtes

Dans notre approche, une fois les différentes requêtes potentielles générées, elles doivent être ordonnées afin de choisir la meilleure au sens d'un certain critère (cf section ??). Pour attribuer un score aux requêtes, nous allons utiliser une représentation synthétique de celles-ci. Cette représentation prend en compte les mots constituant la requête, la fonction de ces mots dans la question déterminée par son analyse, et une information de généralité de la requête.

Le modèle développé permet d'apprendre des requêtes booléennes. Le choix de ce type de requête est dû tout d'abord au fait que les moteurs de recherche les plus efficaces, comme `Google`, `Yahoo!` ou `AltaVista` utilisent des requêtes booléennes. De plus, pour la tâche de question/réponse ces requêtes donnent de meilleurs résultats que les requêtes vectorielles sur des corpus restreints comme la collection `Aquaint`.

Pour effectuer la sélection, nous adoptons une représentation des requêtes sous forme d'un vecteur à 12 caractéristiques. Une grande partie de ces caractéristiques utilise des informations syntaxiques et morpho-syntaxiques issues de l'analyseur de questions utilisé en 2002 lors de l'évaluation TREC dans le système Q/R réalisé au LIMSI [?].

#### 3.1 Informations fournies par l'analyseur de questions

Parmi les informations fournies par l'analyseur de questions de LIMSI, nous nous concentrons sur :

- Les étiquettes morpho-syntaxiques des mots de la question,
- Les lemmes des mots de la question,
- Le verbe principal de la question,
- Le focus de la question, défini comme étant le groupe nominal représentant l'objet exact de la question,
- Le type générique de la question, qui, lorsqu'il est présent, permet de déterminer une catégorie sémantique de la réponse à la question.

Il est à noter que dans certains cas, le focus et le type générique d'une question sont confondus.

À titre d'exemple, nous montrons ci-dessous le focus, le type générique et le verbe principal des questions TREC 1394 et 1396.

Question 1394: In what country did the game of croquet originate ?  
 Verbe principal: originate

Focus: game of croquet  
 Type générique: country

Question 1396: What is the name of the volcano that destroyed the ancient city of Pompeii ?  
 Verbe principal: destroy  
 Focus: volcano  
 Type générique: volcano

### 3.2 Caractéristiques de requêtes

À partir des informations données par l'analyseur de question, nous représentons une requête  $R$  composée d'un sous-ensemble de mots d'une question  $Q$  selon 12 caractéristiques. Ces caractéristiques peuvent être regroupées suivant trois catégories :

1. Caractéristiques liées à la fonction des mots de la requête dans la question. Les trois premières caractéristiques valent 0 si le type générique et le focus sont confondus et sont égales dans le cas contraire au :
  - nombre de mots de  $R$  qui appartiennent au focus de  $Q$  si focus et type générique ne sont pas confondus,
  - nombre de mots de  $R$  qui appartiennent au type générique de  $Q$  si focus et type générique ne sont pas confondus,
  - nombre de mots de  $R$  qui appartiennent au focus de  $Q$  si focus et type générique sont confondus,
  - caractéristique binaire valant 1 si  $R$  contient le verbe principal de  $Q$  et 0 sinon.
2. Caractéristiques liées aux étiquettes morpho-syntaxiques des mots de  $R$  qui ne sont pas le verbe principal de  $Q$  et qui ne sont pas non plus présents ni dans le focus ni dans le type générique de  $Q$  :
  - nombre de noms communs,
  - nombre de cardinaux,
  - nombre de verbes,
  - nombre d'adjectifs,
  - nombre d'adverbes,
  - nombre de noms propres qui sont immédiatement suivis dans  $Q$  par un autre nom propre. Cette caractéristique reconnaît les prénoms en différenciant, dans une suite de noms propres du type *Prénom Nom* le rôle du *prénom* et du *nom*.
  - nombre de noms propres qui ne sont pas immédiatement suivis par un autre nom propre. Cette caractéristique permet de reconnaître les noms de famille dans une suite *Prénom Nom*.
3. Caractéristique, liée au nombre de passages renvoyés par  $R$  après interrogation du moteur de recherche:

- le logarithme du nombre de passages renvoyés plus une constante  $K$ . Cette valeur "nombre de passages" caractérise la généralité de la requête dans le corpus traité. La constante  $K$  a une valeur qui est de l'ordre de grandeur du nombre de passages que l'on veut renvoyer. Pour nos expériences, nous l'avons fixé à 100.

## 4 Apprentissage de requêtes: classification et comparaison

La sélection de requête nécessite de définir un critère pour juger de la pertinence des différentes requêtes candidates et sélectionner la meilleure. Pour cela, nous définissons  $pres(R, Q)$  la précision d'une requête booléenne  $R$  générée à partir d'une question  $Q$  comme suit :

$$pres(R, Q) = \frac{\# \text{ de documents renvoyés par le MR avec } R \text{ contenant la réponse à } Q}{\# \text{ de documents renvoyés par le MR après interrogation par } R}$$

Parmi l'ensemble  $\{R_1, \dots, R_P\}$  de requêtes générées pour une question  $Q$  donnée, nous définissons **la meilleure requête**  $R_{max}$  **pour**  $Q$  comme étant celle qui a la précision maximale.

Cette définition de la meilleure requête nous permet de comparer les requêtes entre elles. En effet, étant donné deux requêtes  $R$  et  $R'$ , générées à partir des mots d'une question  $Q$ , nous pouvons donner les deux définitions suivantes:

- Nous dirons que  $R$  et  $R'$  sont **comparables** si et seulement si  $R$  et  $R'$  sont générées à partir des mots d'une même question  $Q$  et que  $pres(R, Q) \neq pres(R', Q)$ ,
- De plus, pour deux requête comparables  $R$  et  $R'$ , nous dirons que  $R$  **est meilleure que**  $R'$  et nous noterons  $R \succ R'$  si et seulement si  $pres(R, Q) > pres(R', Q)$ .

Nous allons par la suite présenter deux approches d'apprentissage utilisant la caractérisation de requêtes présentée à la section ???. La première approche est définie dans le cadre classique de la *classification* de requêtes, elle associe un score unique à chaque requête par une approche classification, et la meilleure est sélectionnée. La seconde approche utilise la *comparaison* de requêtes.

### 4.1 Approche par classification

L'approche de classification que nous adoptons est définie dans le cadre des problèmes de classification bi-classes. Elle suit l'approche classique de sélection d'éléments par un classifieur utilisée dans le cadre Q/R pour des passages ou des réponses ainsi qu'il a été décrit en section ???. L'originalité du travail tient à l'utilisation de l'approche sur des requêtes, ce qui n'a jamais été testé à notre connaissance. Étant donné une question  $Q$  et une requête  $R$  composée d'un sous-ensemble des mots de la question, nous allons associer à  $R$  une des deux classes suivantes:

- $c(R) = 1$  (classe de  $R$  est 1) si et seulement si  $R$  est la meilleure requête pour la question  $Q$ .
- $c(R) = 0$  (classe de  $R$  est 0) si  $R$  n'est pas la meilleure requête pour la question  $Q$ .

N'importe quel classifieur peut être utilisé pour cela à condition qu'il produise un estimateur des probabilités a posteriori. Le choix du classifieur lui même n'est pas crucial pour étudier le potentiel de l'approche. Nous utilisons dans nos tests un modèle de discrimination logistique [?]. Ce modèle fait l'hypothèse suivante : Pour toute requête  $R$ , issue de la question  $Q$  et représentée par le vecteur  $\mathbf{R} = (R_1, \dots, R_p)$ , la probabilité  $P(c = 1|\mathbf{R})$  que  $\mathbf{R}$  soit de classe 1 a une forme logistique :

$$P(c = 1|\mathbf{R}) = \frac{1}{1 + e^{-\sum_{i=1}^p \lambda_i R_i}} \quad (1)$$

où les paramètres  $\lambda = (\lambda_1, \dots, \lambda_p)$  ( $p$  est égal à 12 dans notre cas) seront estimés par apprentissage sur une base de cas étiquetée. La probabilité  $P(c = 0|\mathbf{R})$  que  $R$  soit de classe 0 peut se déduire de l'expression précédente par

$$P(c = 0|\mathbf{R}) = 1 - P(c = 1|\mathbf{R}) = \frac{e^{-\sum_{i=1}^p \lambda_i R_i}}{1 + e^{-\sum_{i=1}^p \lambda_i R_i}} \quad (2)$$

Les paramètres  $\lambda$  sont appris en minimisant le critère Erreur Quadratique Moyenne (EQM) sur une base d'apprentissage (BA):

$$EQM = \sum_{R \in BA} \{c(R) - P(c = c(R)|R)\}^2 \quad (3)$$

où  $c(R)$  désigne la classe de la requête. Une fois que les paramètres du générateur automatique sont *appris* sur une base d'apprentissage, la sortie du système est utilisée pour affecter un score aux requêtes pour une question quelconque  $Q$  donnée.

Ainsi la meilleure requête  $R_Q^{max}(classif)$  pour un ensemble  $R$  des requêtes qu'il est possible de générer à partir d'une question  $Q$  avec le classifieur est celle qui maximise la probabilité a posteriori d'être de classe 1, estimée par la sortie du modèle :

$$R_Q^{max}(classif) = \underset{R \in R}{argmax} P(c = 1|R) \quad (4)$$

## 4.2 Approche par comparaison

Nous proposons une nouvelle approche pour l'apprentissage de la meilleure requête qui consiste à apprendre une fonction de comparaison.

À la différence du cadre habituel de la classification bi-classe, cette approche consiste à déterminer de façon automatique une fonction binaire *cmp*, prenant en entrée deux requêtes **comparables**  $R$  et  $R'$ , et telle que:

$$cmp(R, R') = \begin{cases} 1 & \text{si } R \succ R' \\ 0 & \text{sinon} \end{cases}$$

**Description du modèle** Afin de déterminer cette fonction, nous représentons un couple de requêtes comparables  $(R, R')$  par la différence de leurs vecteurs représentatifs.

Si  $R$  et  $R'$  sont respectivement représentées par  $\mathbf{R} = (R_1, \dots, R_p)$  et  $\mathbf{R}' = (R'_1, \dots, R'_p)$ , alors le couple  $(R, R')$  est représenté par :

$$\mathbf{R} - \mathbf{R}' = (R_1 - R'_1, \dots, R_p - R'_p) \quad (5)$$

À partir de cette nouvelle représentation, nous pouvons ramener le problème de l'apprentissage de la fonction de comparaison à un problème de classification bi-classes dans l'espace  $\mathbf{CR}$  formé par les couples de requêtes comparables de notre base d'apprentissage  $\mathbf{BA}$ ,  $(R, R')$  sera de classe 1 si  $R \succ R'$  et de classe 0 sinon.

La classification sera réalisée en utilisant le modèle logistique décrit précédemment mais sur la représentation (??). En notant par  $A = (\alpha_1, \dots, \alpha_p)$  les paramètres de ce nouveau modèle, les probabilités a posteriori seront :

$$P(c_{cmp} = 1|(R, R')) = \frac{1}{1 + e^{-\sum_{i=1}^p \alpha_i (R_i - R'_i)}} \quad (6)$$

et  $P(c_{cmp} = 0|(R, R')) = \frac{e^{-\sum_{i=1}^p \alpha_i (R_i - R'_i)}}{1 + e^{-\sum_{i=1}^p \alpha_i (R_i - R'_i)}}$ . Nous utilisons aussi le même critère EQM pour l'apprentissage sur la base d'exemples  $\mathbf{CR}$  constituée par les couples de requêtes comparables issus de la base  $\mathbf{BA}$  :

$$EQM = \sum_{(R, R') \in CR} [c_{cmp}(R, R') - P(c_{cmp} = c_{cmp}(R, R')|(R, R'))]^2$$

**Sélection de requête** Une fois le modèle appris, la comparaison de requêtes est faite simplement par une fonction lineaire des caractéristiques des requêtes. En effet, si l'on note :

$$f_{cmp}(R) = \sum_{i=1}^p \alpha_i R_i$$

on a immédiatement :

$$f_{cmp}(R) > f_{cmp}(R') \Leftrightarrow P(c_{cmp} = 1|(R, R')) > 0,5 \quad (7)$$

Pour la sélection de la meilleure requête à partir de toutes celles  $R$  qu'il est possible de générer pour une question  $Q$  donnée, il suffit de sélectionner la requête  $R_Q^{max}(comp)$  qui maximise la fonction  $f_{cmp}$  :

$$R_Q^{max}(comp) = \underset{R \in \mathbf{R}}{\operatorname{argmax}} f_{cmp}(R)$$

## 5 Expériences

Nous avons évalué notre modèle d'apprentissage avec les questions de l'évaluation TREC 2002, sur le corpus Aquaint, utilisé lors de cette même évaluation.

Nous avons constitué notre base d'apprentissage sur les 200 premières questions. Les 300 dernières questions ont servi de base de test.

Le moteur de recherche utilisé est le moteur Managing Gigabyte (MG) [?], utilisé dans l'évaluation TREC 2002 par ([?], [?]). Il permet une interrogation booléenne. La collection a été découpée par paragraphes, afin de trouver des unités textuelles plus petites que des documents. Le découpage en paragraphes utilise les balises fournies dans la collection. De tels découpages de la collection Aquaint ont été largement utilisés lors de l'évaluation TREC 2002 ([?], [?], [?]).

### 5.1 Construction des bases d'apprentissage

Nous avons créé deux bases d'apprentissage **BA** et **CR** respectivement pour le classifieur et pour le comparateur.

La première base a été réalisée en considérant toutes les requêtes que l'on peut constituer à partir des mots d'une question. Un prétraitement a été effectué dans les questions, pour ne retenir que les noms, verbes, adjectifs, adverbes, cardinaux et noms propres. Nous avons ensuite enlevé les auxiliaires (*be*, *have*, *do* lorsqu'ils n'étaient pas le verbe principal de la question).

Pour chaque requête que l'on peut constituer à partir des mots restant des questions, nous avons calculé les caractéristiques décrites dans la section ???. Pour l'estimation de la précision des requêtes nous avons estimé qu'un paragraphe contenait la réponse à une question donnée s'il vérifiait les deux conditions suivantes:

1. Le paragraphe contient la chaîne de caractères contenant la réponse. Cette vérification est faite en utilisant les expressions régulières décrivant les réponses fournies par le NIST,
2. Le paragraphe est extrait d'un document évalué par le NIST comme contenant la réponse.

Nous avons ensuite supprimé de la base d'apprentissage les requêtes associées aux questions n'ayant pas de réponse dans la collection. De plus, toutes les requêtes ne renvoyant aucun document ont été supprimées, de même que les requêtes ayant une précision inférieure à 1%. Cette valeur n'est pas dépendante du corpus, elle correspond à une précision admissible minimale pour le système.

Comme les classes sont fortement déséquilibrées (environ 30 requêtes de classe 0 pour 1 requête de classe 1), nous avons rééchantillonné la distribution des exemples d'apprentissage pour avoir une répartition équilibrée.

La seconde base d'apprentissage, utilisée pour apprendre le comparateur, à été créée à partir de la précédente. Pour une question donnée, nous considérons les couples de requêtes ayant une précision différente. Pour limiter l'importance d'une question dans la base d'apprentissage, nous limitons pour cette question à 30 le nombre de requêtes à prendre qui ont une précision non nulle, et à 200 le nombre de requêtes à prendre qui ont une précision nulle. La sélection de ces requêtes est aléatoire. Nous mettons alors dans la base d'apprentissage tous les couples  $(R, R')$  comparables ainsi trouvés. Pour équilibrer le nombre d'exemples de chaque classe, pour un couple  $(R, R')$  présent dans la base, nous rajoutons le couple  $(R', R)$ , qui sera de l'autre classe.

## 5.2 Expériences réalisées

Nous avons testé notre modèle avec les deux approches d'apprentissage présentées en section ???. Pour comparaison nous avons utilisé deux autres méthodes de générations de requêtes sans apprentissage; **(a)** un système `Vectoriel` qui est proche de celui utilisé par le LIMSI [?] et **(b)** un système nommé par la suite `All words`, qui génère une requête booléenne.

Le système `All Words` génère une requête booléenne qui est la conjonction des mots de ce sous-ensemble pré-sélectionné. Le système `Vectoriel` génère une requête vectorielle composée de tous les mots de ce sous-ensemble en allouant un poids double aux mots du focus. Les systèmes `Classifieur` et `Comparateur` génèrent une requête booléenne selon les deux approches décrites respectivement dans les sections ??? et ???.

Les systèmes d'apprentissage et `All words` utilisent le même ensemble de mots pré-sélectionnés. L'apprentissage sélectionne une requête correspondant à un sous ensemble de ces mots, on peut ainsi mesurer simplement son apport.

## 5.3 Mesures d'évaluations

Nous évaluons les différentes méthodes de génération de requêtes selon trois critères:

- Le nombre de passages renvoyés moyen, qui permet de déterminer si l'ordre de grandeur du nombre de passages renvoyés est acceptable,
- Le rappel moyen, définit comme suit :

$$\frac{\sum_Q \# \text{ de passages renvoyés par le MR avec } R \text{ contenant la réponse à } Q}{\sum_Q \# \text{ de passages contenant la réponse à } Q \text{ dans la collection}}$$

- Le pourcentage de questions pour lesquelles le système trouve au moins une bonne réponse. Cette valeur donne le nombre maximum de questions auxquelles un système utilisant la méthode de génération testée pourra répondre.

#### 5.4 Exemple de requêtes générées

Nous donnons à titre d'exemple, les requêtes générées par les différentes méthodes pour la Question 1402 de TREC 2002 pour laquelle il y a 9 passages contenant la réponse dans la collection. Nous donnons dans la section suivante une évaluation globale sur les 300 dernières questions de TREC 2002 qui constitue l'ensemble de notre base test.

Question 1402: What year did Wilt Chamberlain score 100 points ?

Vectoriel: year Wilt Chamberlain score 100 points Chamberlain Wilt  
 All Words: ((((((year) (chamberlain)) (wilt)) (100)) (score)) (point))  
 Classifieur: (((year) (wilt)) (score)) (point))  
 Compareteur: ((wilt) (100))

Le tableau ??, récapitule le nombre de passages total renvoyés ainsi que le nombre de passages contenant la réponse par le moteur MG [?] avec les requêtes générées par les différents systèmes.

**Table 1.** Nombre de passages renvoyés avec les requêtes générées par les différents systèmes pour la Question 1402 de TREC 2002

Générateur	# de passages renvoyés	# de passages contenant la réponse
All words	0	0
Vectoriel	300	5
Classifieur	10	0
Compareteur	81	5

La requête du système All Words ne renvoie aucun document. Ceci montre la nécessité de sélectionner parmi ce premier ensemble de mots, un sous ensemble qui est susceptible de renvoyer des passages contenant la réponse. Sur cet exemple nous remarquons que la sélection de mots avec le Compareteur est plus pertinente qu'avec le Classifieur. En effet, on voit dans les différentes requêtes données ci dessus que le compareteur sélectionne le terme 100 qui est pertinent pour cette question et qui n'est pas choisi par le classifieur.

Par ailleurs, la requête générée par le Classifieur semble être trop contrainte. Ainsi, le Compareteur propose une meilleure contrainte pour la question. Contrairement aux requêtes booléennes qui ne sont pas ordonnées, le système Vectoriel génère une requête qui permettra d'obtenir une liste ordonnée de passages. Le seuil de sélection sur le nombre de passages renvoyés est important dans ce cas. Dans le tableau nous avons fixé ce seuil à 300, bien que dans ce cas, avec un seuil de 750 le nombre de passages contenant la réponse aurait été identique.

## 5.5 Résultats obtenus

Le tableau ??, décrit les résultats obtenus avec les différents générateurs sur la base test.

**Table 2.** Comparaisons entre différents systèmes de génération de requêtes sur la base test composée des 300 dernières questions de TREC 2002.

Générateur	# de passages renvoyés	% de questions ayant une réponse	Rappel
All words	83	28%	24,8%
Vectorel	240	<b>50%</b>	34,8%
Classifieur	205	36%	28,9%
Compareur	242	45%	<b>42,1%</b>

Nous pouvons remarquer que, par rapport au système All words, l'utilisation de l'apprentissage permet de faire une sélection pertinente des mots clefs améliorant à la fois le rappel (+4,1% avec le Classifieur et +17,3% avec le Compareur) ainsi que le nombre de questions pour lesquelles le moteur de recherche renvoie au moins un passage contenant la réponse (+8% avec le Classifieur et +17% avec le Compareur).

Le Compareur obtient toutefois de meilleurs résultats en généralisation que le Classifieur. Une des raisons à cela est que les bases d'apprentissage sur lesquelles sont entraînées ces modèles ne contiennent pas la même information. Le Compareur est entraîné sur l'ensemble des couples de requêtes comparables alors que le Classifieur est entraîné à répondre "1" sur une unique meilleure requête.

Enfin, nous avons fixé à 240 le nombre de passages renvoyés par le moteur de recherche pour le système Vectorel. Ce nombre correspond au nombre moyen de passages renvoyés avec le Compareur. Le système Vectorel trouve au moins une réponse pour la moitié des questions de la base test (150). Ce qui représente 15 questions de plus que le Compareur, ceci peut s'expliquer par le fait que le système Vectorel prend en compte tous les mots pré-sélectionnés de la question de façon non binaire. Par contre le Compareur obtient un rappel de 7,3% supérieur à celui du système Vectorel. Ceci peut s'expliquer par le fait que l'ordonnement à base d'un critère statistique (cosine dans le cas du moteur MG) n'est pas adapté pour la tâche de Q/R. Si l'on veut comparer les deux, sur les expériences réalisées le système vectoriel renvoie les passages contenant les réponses pour un plus grand nombre de questions. Les réponses seront cependant plus faciles à extraire par la suite à partir des passages provenant du système compareur car elles seront présentes dans un plus grand nombre de passages et probablement sous des formes différentes. La précision moyenne est meilleure pour le compareur. Globalement les expériences montrent qu'il est possible d'apprendre à générer des requêtes de façon automatique à partir de

la question. Le système proposé améliore les performances du système vectoriel de base en terme de rappel et précision, la voie semble donc prometteuse. Il est clair cependant que les performances sont encore nettement inférieures à celle des meilleurs systèmes TREC. Ceux ci necessitent de nombreux réglages manuels et utilisent d'autres ressources en particulier pour effectuer des extensions. Nous nous sommes concentrés dans ce premier travail sur la sélection en tout automatique et sans utiliser de ressource externe, dans le but d'évaluer la viabilité de l'approche.

## 6 Conclusion

Nous avons proposé dans cet article un modèle d'apprentissage statistique pour la génération de requêtes dans les systèmes de question/réponse. Ce modèle est générique dans le sens où il permet de générer des requêtes booléennes pour différents types de corpus. Pour une question donnée, le modèle proposé sélectionne des mots clefs pertinents parmi un ensemble pré-sélectionné de mots. Ce modèle donne aussi de meilleurs résultats en terme de rappel qu'une méthode de génération de requêtes vectorielles sans enrichissement.

Nous avons proposé deux approches pour entraîner notre modèle basée l'une sur la classification de requêtes et l'autre sur la comparaison. L'approche de comparaison présentée dans cet article a obtenu de meilleur résultats que la classification. Cela permet de confirmer que dans une tâche de sélection du meilleur élément dans un ensemble déterminé, la considération des dépendances entre ces éléments au moment de l'apprentissage améliore les résultats en généralisation.

L'apport de l'apprentissage bien qu'établi dans ce premier travail doit être amélioré. Nous pensons que les reformulations des mots des questions dans la génération de requêtes permettrait d'améliorer les résultats de nos modèles. Nous envisageons de nous concentrer sur l'apprentissage de requêtes en prenant en compte de telles reformulations.

## Remerciements

Ce travail a été sponsoré en parti par le programme IST de la communauté Européenne, sous le réseau d'excellence de PASCAL, IST-2002-506778. Cette publication reflète seulement les points de vus des auteurs.

## References

1. J. A. Anderson. Logistic discrimination. In *Handbook of Statistics*, P.R. Krishnaiah and L. Kanal (Eds.), 2:169–191, 1982.
2. E. Agichtein, S. Lawrence, L. Gravano. Learning engine specific query transformation for question/answering, Columbia University, NY, In Proceedings of the 10th WWW Conference (WWW10), p. 169–178, 2001.

3. G. Attardi, A. Cisternino, F. Formica, M. Simi, A. Tommasi, PiQASSo 2002, University of Pisa, TREC 2002
4. G. de Chalendar, T. Dalmas, F. Elkateb-Gara, O. Ferret, B. Grau, M. Hurault-Plantet, G. Illouz, L. Monceaux, I. Robba, A. Vilnat, LIMSI-CNRS, The Question Answering System QALC at LIMSI, Experiments in Using Web and WordNet, In Proceedings of the 11th Text REtrieval Conference (TREC-11), 2002.
5. E. Brill, S. Dumais and M. Banko., An Analysis of the AskMSR Question-Answering System, Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing
6. C. Carroll, J. Prager, C. Welty, K. Czuba, D. Ferrucci, IBM T.J. Watson Research Center, A Multi-Strategy and Multi-Source Approach to Question Answering, In Proceedings of the 11 th Text REtrieval Conference (TREC 11), 2002.
7. A. Echiabi & U. Hermjakob & E. Hovy & D. Marcu & E. Melz and D. Ravichandran, Multiple-Engine Question Answering in TextMap, In Proceedings of TREC 2003 conference, 2003.
8. C. Fellbaum. WordNet, an Electronic Lexical Database, MIT Press, Cambridge MA, 1998
9. S. Harabagiu, D. Moldovan, M. Pasca, R. Mihalcea, M. Surdeanu, R. Bunescu, R. Garju, V. Rus and P. Morarescu. The role of lexico-semantic feedback in open-domain textual question-answering. In Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (ACL-2001)
10. U. Hermjakob, A. Echiabi, D. Marcu, University of Southern California, Natural Language Based Reformulation Resource and Wide Exploitation for Question Answering, In Proceedings of the 11 th Text REtrieval Conference (TREC 11), 2002
11. C. Jacquemin. Syntagmatic and paradigmatic representations of term variation. 37th Annual Meeting of the Association for Computational Linguistics (ACL'99), p. 341–348, 1999
12. A. Ittycheriah, S. Roukos, IBM T.J. Watson Research Center, IBM's Statistical Question Answering System–TREC 11, In Proceedings of the 11 th Text REtrieval Conference (TREC-11).
13. B. Magnini, M. Negri, R. Prevete, H. Tanev, ITC-Irst , Mining Knowledge from Repeated Co-Occurrences: DIOGENE at TREC 2002, In Proceedings of the 11th Text REtrieval Conference (TREC-11), 2002.
14. D. Moldovan, S. Harabagiu, M. Pasca, R. Mihalcea, R. Girju, R. Goodrum, V. Rus. The Structure and Performance of an Open-Domain Question Answering System. In Proceedings of the Conference of the Association for Computational Linguistics (ACL-2000)
15. D. Moldovan, S. Harabagiu, R. Girju, P. Morarescu, F. Lacatusu, A. Novischi, A. Badulescu, O. Bolohan, Language Computer Corporation, LCC Tools for Question Answering, In Proceedings of the 11th Text REtrieval Conference (TREC-11), 2002.
16. H. T. Ng, J. Lai, P. Kwan, Y. Xia. Question answering using a large text database: a machine learning approach, In proceedings of EMNLP 01, 2001.
17. T. Nyberg & J. Mitamura & J. Carbonnell & K. Callan & K. Collins-Thompson & M. Czuba & L. Duggan & N. Hiyakumoto & Y. Hu & J. Huang & L.V. Ko & S. Lita & V. Murtagh & D. Pedro & Svoboda, Carnegia Mellon University, The JAVELIN Question-Answering System at TREC 2002, In Proceedings of the 11 th Text REtrieval Conference (TREC 11), 2002.

18. D. Ravichandranet, E. Hovy, F. J. Och. Statistical QA-classifier vs re-ranker, what's the difference?, In Proceedings of the ACL Workshop on Multilingual Summarization and Question Answering—Machine Learning and Beyond, Sapporo, Japan, 2003.
19. E.M. Voorhees, Overview of TREC 2002, National Institute of Standards and Technology
20. I. Witten & A. Moffat & T. Bell "Managing Gigabyte", Morgan Kaufmann, 1999.
21. J. Xu & W.B. Croft : Query Expansion Using Local and Global Document Analysis. in Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp.4-11. Zurich, Switzerland (1996)
22. H. Yang & T.-S. Chua, National University of Singapore, The Integration of Lexical Knowledge and External Resources for Question Answering, In Proceedings of the 11th Text REtrieval Conference (TREC-11), 2002.