Une approche à base d'heuristiques pour la génération des requêtes spatio-temporelles à partir des questions en langage naturel

Ghada Landoulsi, Khaoula Mahmoudi, Sami Faiz

**Laboratoire de Télédétection et Systèmes d'Information à Références Spatiales,*
Ecole Nationale d'Ingénieurs de Tunis
BP 37 le Belvédère 1002 Tunis, TUNISIE

ghadalandoulsi@yahoo.fr, kamahmoudi@yahoo.fr, sami.faiz@insat.rnu

RESUME. De nos jours, les bases de données géographiques sont devenues largement omniprésentes et ont le potentiel d'assister les décideurs s'occupant des données spatiales, temporelles ou spatio-temporelles. Elles sont souvent amenées à être exploitées par des non-experts qui n'ayant pas une grande compétence et ne maîtrisant pas rigoureusement l'utilisation des systèmes d'information géographiques. Dans ce contexte, les systèmes de question réponse se sont développés pour faciliter l'accès au grand public en lui offrant la possibilité d'utiliser le langage naturel pour exprimer ses besoins informationnels. Dans ce présent article, nous proposons une approche basée sur deux phases: La première consiste à assurer le prétraitement de la question de l'utilisateur en langage naturel et la deuxième est dédiée à assurer la traduction de cette question en un format structuré à savoir une requête SQL spatio-temporelle.

ABSTRACT. Nowadays, geographic databases have become largely ubiquitous and have the potential to assist decision-makers dealing with spatial, temporal or spatio-temporal data. These latter, are often exploited by non-experts who do not have a great deal of expertise and do not master the use of geographical information systems rigorously. In this context, question answering systems are developed to facilitate access to the general public by offering them the possibility of using natural language to express their information needs. In this paper, we propose a two staged approach: The first stage is dedicated to ensure the preprocessing of the natural language user's question and the second is devoted to ensure the translation of this question into a structured format namely spatio-temporal SQL query.

Mots-clés : Langage naturel, bases de données géographiques, requête spatio-temporelle.

KEYWORDS: Natural language, geographic database, Spatio-temporal query.

2

1. Introduction

De nos jours, les Bases de données géographiques (BDG) représentent une source d'information très importante pour tous les types d'utilisateurs, experts ou non experts, stockant une partie importante des informations géographiques (Gesbert, 2004). Dans ce cadre, il est intéressant d'une part qu'un utilisateur soit capable d'accéder à une BDG et exploiter les informations stockées dedans. D'autre part, les Systèmes d'informations Géographiques (SIG) restent des outils de spécialistes qui nécessitent des compétences spécialisées, ce qui engendre un problème d'accès à l'information géographique par le grand public.

Notre objectif dans ce présent article consiste à faciliter l'accès à ces bases de données en utilisant le langage naturel comme moyen de communication. L'utilité d'une telle fonction appelée par la suite les systèmes de question réponse est évidente. Ces systèmes sont dédiés à faciliter la recherche et à réduire les obstacles entre un utilisateur non-expert et une machine. En revanche, cette dernière est incapable d'interpréter une question en langage naturel. Pour ce faire, nous proposons dans ce papier une nouvelle approche basée sur des techniques de Traitement Automatique du Langage Naturel (TALN). Ces dernières, permettent d'assurer l'interprétation et le prétraitement des questions des utilisateurs et les traduire en un format structuré capable d'extraire les informations souhaitées à partir d'une BDG.

Dans le reste de cet article, la section 2 est dédiée à une revue de la littérature. La section 3 détaille les deux phases de l'approche proposée. La section 4, expose les résultats de simulation de notre approche. Une évaluation est présentée dans la section 5 pour la validation de la contribution présentée. Finalement, une conclusion fournit un récapitulatif du travail réalisé et quelques perspectives qui découlent de cette étude.

2. Etat de l'art

Pour accéder à une base de données il est nécessaire d'utiliser un langage formel sophistiqué qui est le plus souvent le langage de requête structuré SQL. Dans ce contexte, un utilisateur qui ne connait pas la structure d'une base et n'avait aucune idée sur la syntaxe utilisée pour chercher l'information représente un des premiers problèmes intéressants que soulève l'interrogation des bases de données en langage naturel. En fait, pour chercher une information, les utilisateurs n'utilisent pas le même vocabulaire existant dans une base de données ce qui entraîne un problème d'ambiguïté.

Chaudhari (2013) propose d'utiliser un dictionnaire de synonymes afin d'améliorer le vocabulaire qu'il peut utiliser pour exprimer ses besoins en informations. À chaque fois qu'il choisit d'utiliser une nouvelle base de données il doit tout d'abord faire des entrées manuelles au niveau du dictionnaire. Dans le même contexte, plusieurs autres recherches ont donné de l'importance à l'aspect sémantique en plus de la gestion de la synonymie. Ils ont proposé de résoudre les

problèmes liés à la restriction de vocabulaires des bases de données afin d'assurer la correspondance soit entre la question de l'utilisateur et la réponse (Giordani et Moschitti, 2012) soit entre la question de l'utilisateur et la requête SQL (Giordani et Moschitti, 2009). Concernant la génération des requêtes SQL, plusieurs méthodes et techniques ont été proposées dans la littérature. Certains sont basées sur des grammaires probabilistes (Deshpande et Devale, 2012) ou des grammaires avec des règles prédéfinies (Alexander et al., 2013). D'autres ont utilisé un système d'apprentissage permettant d'exploiter la structure d'une base de données pour générer des requêtes SQL classifiées selon leurs degrés de plausibilité grâce à un SVM-ranker (Giordani et Moschitti, 2012). Ces méthodes ne fonctionnent que sur des bases de données particulières traitant différents types de données telles que des données médicales (Abacha, 2012), des données géographiques (Chen, 2014), etc. Plus récemment, une approche à base d'esquisse est proposée par Yaghmazadeh et al. (2017). Une esquisse est caractérisée par une grammaire plus simple que le SQL et elle est déterminée en utilisant un système à base de règles, puis raffinée et réparée de manière itérative en se basant sur un ensemble de règles et d'heuristiques. Pour la génération d'une esquisse de requête, Yaghmazadeh et al. (2017) utilise l'apprentissage automatique et Word2Vec (Mikolov et al., 2013). En fait, elle représente le squelette d'une requête SQL avec ses différentes clauses, mais elle ne contient aucune information sur le schéma de la base de données utilisée. En s'inspirant des différentes méthodes proposées dans les travaux présentés dans la littérature, et afin d'accomplir le processus de génération des requêtes spatio-temporelles, nous proposons dans ce présent travail une approche à base d'heuristiques.

3. Approche proposée

Dans ce papier, nous proposons une approche à base de deux phases : La première consiste à assurer le prétraitement de la question de l'utilisateur en langage naturel et la deuxième est dédiées à assurer la traduction de cette question en un format structuré à savoir une requête SQL spatio-temporelle.

3.1. Prétraitement des questions des utilisateurs

Cette étape est caractérisée principalement par les trois étapes suivantes : étiquetage morphosyntaxique, génération de l'arbre de dépendance et la reconnaissance des entités et des relations géographiques.

3.1.1 Étiquetage morphosyntaxique

L'étiquetage morphosyntaxique consiste à attribuer la catégorie grammaticale (notée Cat_g) qui correspond à chaque mot de la question de l'utilisateur (noté QU) (Ferraro et al., 2013). La figure 1(1) présente le processus d'étiquetage morphosyntaxique de la question suivante : *« Determine the country where the wildfire disaster reaches the highest number of victims in 2014 ».* Une variété d'étiquettes morphosyntaxiques peut être utilisée pour désigner la catégorie

grammaticale des mots telles que : les adjectifs comparatifs (JJR), les noms propres singuliers (NNP), les adverbes comparatifs (RBR), etc.

3.1.2 Génération de l'arbre de dépendance

La génération de l'arbre de dépendance est dédiée à extraire les différentes relations lexicales qui existent entre chaque paire de mots en utilisant l'analyseur syntaxique de Marneffe et al, (2006). Ce dernier permet en fait de spécifier le lien syntaxique entre deux mots W_i et W_j. Dans une question (QU) il est possible d'avoir des mots composés dont la relation de dépendance (notée DRel) est « compound ». Dans ce cas, nous proposons un ensemble d'heuristiques permettant de combiner les noms composés et de les traiter en tant qu'un seul mot (figure 1(2)):

- Heuristique 1 : $\forall W_i, W_{i+1} \in QU\ (Cat_g(W_i) = WRB) \wedge (Cat_g(W_{i+1}) = JJ \vee RB) \wedge (DRel(W_{i+1}, W_i) = advmod) \Rightarrow Cat_g(W_i_W_{i+1}) = RB$

Les mots composés tels que (how many, how longer, how far, etc.) doivent être considérés comme un seul mot. Dans ce cas, cette heuristique peut être interprétée comme suit : quelque soit un mot i (W_i) et un mot i+1 (W_{i+1}) appartenant à une question d'utilisateur (QU), Si W_i, un adverbe en WH (WRB) est suivi par un adjectif (JJ) ou par un adverbe (RB) (W_{i+1}) et que leur relation de dépendance notée DRel(W_{i+1}, W_i) représente un modificateur adverbial (advmod) alors on considère W_i et W_{i+1} en tant qu'un seul adverbe (RB).

- Heuristique 2 : $\forall W_i, W_{i+1} \in QU\ (Cat_g(W_i) = NN \vee NNS) \wedge (Cat_g(W_{i+1}) = NN \vee NNS) \wedge (DRel(W_{i+1}, W_i) = compound) \Rightarrow Cat_g(W_i_W_{i+1}) = NN \vee NNS$

Heuristique 2 indique que si un nom (W_i) singulier (NN) ou pluriel (NNS) est suivi par un autre nom W_{i+1} et que leur relation de dépendance DRel(W_{i+1}, W_j) est compound, alors nous considérons W_i et W_{i+1} comme un seul mot dont la catégorie grammaticale est un nom (NN ou NNS).

- Heuristique 3 : $\forall W_i, W_{i+1}, W_{i+2} \in QU\ (Cat_g(W_i) = NN \vee NNS) \wedge (Cat_g(W_{i+1}) = IN) \wedge (Cat_g(W_{i+2}) = NN \vee NNS) \wedge (DRel(W_{i+2}, W_i) = nmod) \Rightarrow Cat_g(W_i_W_{i+1}_W_{i+2}) = NN \vee NNS$

La troisième heuristique suppose que Si W_i et W_{i+2} sont deux noms (NN ou NNS) reliés par une préposition ou une conjonction de subordination (IN) et que leur relation de dépendance DRel(W_{i+2}, W_i) est un modificateur nominal (nmod), alors W_i et W_{i+1} et W_{i+2} peuvent être considérés comme un seul mot.

- Heuristique 4 : $\forall W_i, W_{i+1} \in QU\ (Cat_g(W_i) = JJ) \wedge (Cat_g(W_{i+1}) = NN \vee NNS) \wedge (DRel(W_{i+1}, W_i) = adjmod) \Rightarrow Cat_g(W_i_W_{i+1}) = NN \vee NNS$

Si Wi est un adjectif (JJ) suivi par un nom W_{i+1} et que leur relation de dépendance DRel(W_{i+1}, W_i) est un modificateur adjectival (adjmod) tel que (last year, previous day, etc) alors W_i et W_{i+1} peuvent être considérés comme un seul mot.

- Heuristique 5 : $\forall W_i, W_{i+1}, W_{i+2} \in QU\ (Cat_g(W_i) = JJ) \wedge (Cat_g(W_{i+1}) = CD) \wedge (Cat_g(W_{i+2}) = NN \vee NNS) \Rightarrow Cat_g(W_i_W_{i+1}_W_{i+2}) = NN \vee NNS$

Dans ce cas, W_i et W_{i+2} peuvent être reliés par un nombre cardinal (CD) W_{i+1} tel que (last three years) dans ce cas, W_i, W_{i+1} et W_{i+2} peuvent être traités comme un seul mot.

3.1.3 Reconnaissance des entités et des relations géographiques

En général, une question spatio-temporelle implique une recherche dans les dimensions spatiales et temporelles. Ce type de question est dédié à décrire des événements en identifiant leur localisation, leur date, leur durée, leur relation temporelle et leur relation spatio-temporelle. Dans la littérature, quatre types de requêtes spatio-temporelles sont mentionnés (Yuan et McIntosh, 2002) : Requête spatio-temporelle simple, requête spatio-temporelle d'intervalle, requête spatio-temporelle comportementale et requête spatio-temporelle relationnelle. Dans ce présent article, nous proposons de traiter les requêtes spatio-temporelles simples et les requêtes spatio-temporelles d'intervalles.

Le type de requêtes spatio-temporelles simples est destiné à chercher des informations sur la position d'un objet à un moment donné, le moment où un objet peut exister à une localisation particulière ou l'objet pouvant exister à un endroit et à un moment donné. En ce qui concerne le type de requêtes spatio-temporelles d'intervalles, il est dédié à chercher des informations sur les changements pouvant survenir dans une région au cours d'une période donnée. Dans ce cadre, les utilisateurs peuvent utiliser des expressions temporelles ou spatiales ambigües et imprécises pour décrire leurs besoins en informations. Pour identifier ces ambiguïtés, nous avons utilisé un outil permettant de reconnaitre les entités nommées (Manning et al., 2014). Dans le contexte des catastrophes naturelles, toutes les entités géographiques sont étiquetées avec la classe LOCATION, telle que les forêts, les terres, les pays, etc. Les types de catastrophes sont identifiés comme étant DISASTERS. Concernant les expressions temporelles, nous avons adopté un étiqueteur temporel basé sur des règles (Chang et Manning, 2012) supportant cinq types d'expressions temporelles : DATE, TIME, DURATION, SET et INTERVAL.

Dans les études antérieures, seules les relations spatiales (Worboys, 1992) ou temporelles (Allen, 1983) sont prises en compte. Les relations spatiales valides pour une période de temps particulière sont classées comme des relations spatio-temporelles (Raza, 2008). Ces dernières sont identifiées à l'aide d'un ensemble d'expressions régulières (Chang et Manning, 2014) capables de détecter chaque type de relation. De plus, si un utilisateur introduit une question de type WH, dans ce cas, des mots tels que (Where, when, how longer, etc) doivent être identifiés. Ces mots révèlent indirectement ce que l'utilisateur cherche. Par exemple, le mot « Where » peut être étiqueté avec la classe LOCATION et « How_longer » avec la classe DURATION. La figure 1(3) illustre un exemple de l'étape de reconnaissance des classes des entités et des relations géographiques notées Geo_{ERC}.

6

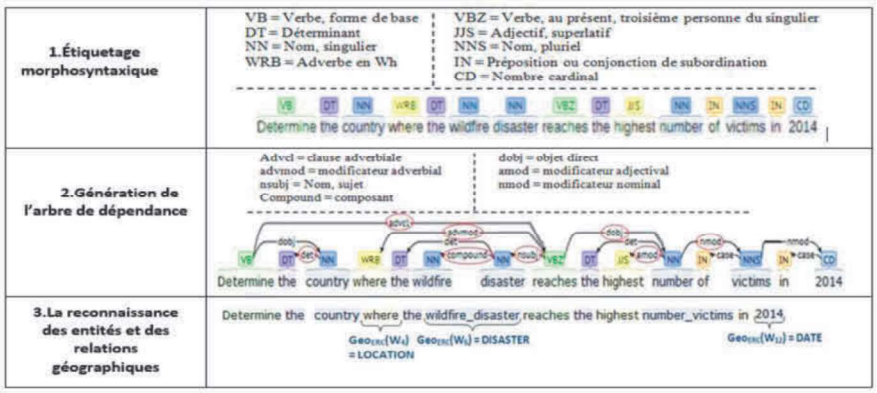

Figure 1. Les différentes étapes de la phase de prétraitement

3.2. Génération des requêtes spatio-temporelles

Pour parvenir à une traduction efficace des questions en requêtes spatio-temporelles, nous proposons d'utiliser une formulation qui conserve la syntaxe standard des requêtes SQL classiques, avec l'intégration de nouvelles fonctions et agrégats liés aux aspects spatiaux, temporels et spatio-temporels. Cette formulation présente la syntaxe suivante : *SELECT nom(s)_attributs(s) FROM nom(s)_table(s) WHERE condition(s) GROUP BY nom(s)_attribut(s) HAVING condition(s).*

3.2.1. Construction de l'instruction SELECT

Le processus de construction de l'instruction SELECT est réalisé en s'appuyant sur un ensemble d'heuristiques (Algorithme 1). Ces dernières sont basées sur les relations et les entités spatiales, temporelles et spatio-temporelles reconnues, la liste des étiquettes morphosyntaxiques et la liste des relations de dépendance.

Algorithme 1. Construction de l'instruction SELECT

---

1. **Entrée** $QU = \{w_1, w_2, w_3 \ldots, w_n\}$, WHC // WHC = Classe du mot de question en WH, $QU = \{w_1, w_2, w_3 \ldots, w_n\}$: QU est constituée d'un ensemble de mots
2. **Sortie** $Sst_{att}L$ // $Sst_{att}L$ = liste des attributs de l'instruction Select
3. **Début**
4. $Sst_{att}L \leftarrow \varnothing$
5. **Pour tout** $W_{i \in \{1, 2, 3, n\}}$ **Faire**
   //Heuristique 1
6. **Si** ($Cat_g(W_i) \in$ {NN, NNS}) et ($DRel(W_j, W_i) \in$ {root,nsubj, det, dobj} **Alors** \\ NN = nom singulier, NNS = nom pluriel, root = racine, nsubj = nom sujet, det = déterminant, dobj = objet direct
7. Ajouter W_i à $Sst_{att}L$
8. **Fin Si**
   //Heuristique 2

9. **Si** ($Cat_g(W_i) \in$ {NN, NNS}) et ($Cat_g(W_j) \in$ {NN, NNS}) et ($DRel(W_j, W_i) \in$ {conj:and})) **Alors** // conj:and = Conjonction
10. Ajouter W_i et W_j à $Sst_{att}L$
11. **Fin Si**

//Heuristique 3

12. **Si** ($Cat_g(W_i) \in$ {NN, NNS}) et ($Cat_g(W_j) \in$ {NNP}) et ($DRel(W_j, W_i) \in$ {nmod, compound}) **Alors** \\ nmod = modificateur nominal, compound = composé
13. Ajouter W_i à $Sst_{att}L$
14. **Fin Si**

//Heuristique 4

15. **Si** ($Cat_g(W_i) \in$ {NN, NNS}) et ($Cat_g(W_j) \in$ {NN, NNS}) et ($DRel(W_j, W_i) \in$ {nmod}) **Alors**
16. Ajouter W_i et W_j à $Sst_{att}L$
17. **Fin Si**

//Heuristique 5

18. **Si** ($Cat_g(W_i) \in$ {WRB, WP}) **Alors** //WRB = adverbe WH, WP = pronom WH
19. Ajouter la classe de mot WH (WHC) du mot W_i à $Sst_{att}L$
20. **Fin Si**
21. **Fin pour**
22. **Fin**

3.2.2. Construction de la clause WHERE

Conjointement avec l'instruction SELECT il est indispensable d'utiliser la clause WHERE (notée Wc) pour filtrer les données. Pour ce faire, nous proposons un ensemble d'heuristiques qui sont déterminées en se référant aux classes d'entités et de relations géographiques (notées Geo_{ERC}) et aux catégories grammaticales (Cat_g).

Algorithme 2. Construction de la clause WHERE

1. **Entrée** $QU = \{w_1, w_2, w_3 \ldots, w_n\}$, Geo_{ERC} // Geo_{ERC} : la classe des entités et des relations géographiques
2. **Sortie** $Wc_{conditions}$ // liste des conditions de la clause WHERE
3. **Début**
4. $Wc_{conditions} \leftarrow \varnothing$
5. **Pour tout** $W_{i \in \{1, 2, 3, n\}}$ **Faire**

//Heuristique 1

6. **Si** ($Cat_g(W_i) \in$ {NNP, NNPS, CD}) **Alors** // NNP: nom propre singulier, NNPS: nom propre pluriel,CD: nombre cardinal
7. $Wc_{conditions} \leftarrow (Geo_{ERC}(W_i) =$ 'W_i')

//Heuristique 2

8. **Si** ($Cat_g(W_i) \in$ {NN, NNS}) suivi par ($Cat_g(W_{i+1}) \in$ {VBZ}) suivi par ($Cat_g(W_{i+2}) \in$ {NNP, NNPS}) **Alors** //NN : nom singulier, NNS : nom pluriel, VBZ : Verbe, au présent, troisième personne du singulier
9. $Wc_{conditions} \leftarrow (W_i =$ 'W_{i+2}')

//Heuristique 3

10. **Si** ($Cat_g(W_i) \in$ {NN, NNS}) et ($Geo_{ERC}(W_i) =$ valeur) **Alors**
11. $Wc_{conditions} \leftarrow (Geo_{ERC}(W_i) =$ 'W_i')

//Heuristique 4

8

12. **Si** ($Cat_g(W_i) \in$ {NN, NNS, NNP, NNPS}) et ($Cat_g(W_{i+1}) \in$ {JJR}) **Alors** //JJR: adjectif comparatif
13. *Calculer la similarité entre les mots : (W_{i+1}), Inferior et Superior*
14. **Si** Similarité (W_{i+1},"Inferior") $\geq$ seuil **Alors**
15. $Wc_{conditions} \leftarrow W_i < 'W_{i+3}'$
16. **Sinon Si** Similarité (W_{i+1}, "Superior") $\geq$ seuil **Alors**
17. $Wc_{conditions} \leftarrow W_i > 'W_{i+3}'$
18. **Fin Si**
19. **Fin Pour**
20. **Fin**

Au niveau de cet algorithme, les trois premières heuristiques consistent à vérifier si un mot i correspond à une catégorie grammaticale bien déterminée, alors la clause WHERE représente une égalité soit entre la classe d'une entité (mot) i et le mot i ($Geo_{ERC}(W_i)$ = 'W_i'), soit entre deux mots (W_i = 'W_{i+2}'). En ce qui concerne l'heuristique 4, nous supposons que le mot W_i correspond à une des catégories grammaticales suivantes {NN, NNS, NNP, NNPS} et qu'il est suivi par un adjectif comparatif (W_{i+1}). Dans ce cas, pour déterminer la condition relative à la clause WHERE, nous proposons de calculer la similarité sémantique entre l'adjectif (W_{i+1}) et les mots *Inferior* et *Superior*. Si la similarité entre W_{i+1} et le mot « *inferior* » est supérieur à un seuil donné alors le nom (NN ou NNS) ou le nom propre (NNP ou NNPS) W_i est inférieur à la valeur W_{i+3} puisque W_{i+2} doit être obligatoirement une préposition qui relie l'adjectif W_{i+1} et la valeur W_{i+3} ($Wc_{conditions} \leftarrow W_i < 'W_{i+3}'$). Sinon si la similarité entre W_{i+1} et le mot « *superior* » est supérieur à un seuil donné alors W_i est supérieur à la valeur W_{i+3} dans ce cas ($Wc_{conditions} \leftarrow W_i > 'W_{i+3}'$).

3.2.3. Construction de la clause HAVING

Différents agrégats peuvent être appliqués pour utiliser les opérateurs d'agrégation spatiaux (SA_{op}) : area, perimeter, distance, etc. Opérateurs d'agrégation temporels (TA_{op}) d'Allen (Allen, 1983) : duration, intersect, equal, etc. Enfin des opérateurs d'agrégation spatio-temporels (STA_{op}) : moving-distance, etc. Le processus de construction de la clause HAVING (Hc) repose principalement sur deux heuristiques. Ces heuristiques sont dédiées à l'identification des attributs d'agrégation encapsulés dans des adjectifs comparatifs ou superlatifs. La première heuristique dans l'algorithme 3 se concentre sur l'idée que si W_i est un nom ou un nom propre suivi d'un adjectif comparatif (noté JJR), les attributs d'agrégation (SA_{att}, TA_{att}, STA_{att}) doivent être identifiés. Dans ce cas, nous proposons d'utiliser la similarité sémantique de Wu-Palmer (Wu et Palmer, 1994) pour calculer la similarité entre W_i et les opérateurs d'agrégation.

Algorithme 3. Construction de la clause HAVING (Heuristique 1)

1. **Entrée** $QU = \{w_1, w_2, w_3 \ldots, w_n\}$, SA_{op}, TA_{op}, STA_{op} // SA_{op} : Opérateur d'agrégation Spatiale, TA_{op} : Opérateur d'agrégation Temporel, STA_{op} : Opérateur d'agrégation Spatio-Temporel

2. **Sortie** $Hc_{conditions}$ \\ $Hc_{conditions}$:la liste des conditions de Hc
3. **Début**
4. **Pour tout** $W_{i \in \{1, 2, 3, n\}}$ **Faire**
5. **Si** ($Cat_g(W_i) \in$ {NN, NNS, NNP, NNPS}) suivie de ($Cat_g(W_j) \in$ {JJR}) **Alors**
6. **Pour tout** (SA_{op}) **Faire**
   7. **Si** la similarité (W_i, ($SA_{op(k)}$)) ≥ seuil **Alors**
      8. $SA_{att} = SA_{op(k)}(W_i)$ //SA_{att} : attribut d'agrégation spatial
   9. **Si** la similarité (W_j, "Inferior") ≥ seuil **Alors**
      10. $Hc_{conditions} \leftarrow SA_{op(k)}(W_i) < W_{j+2}$
   11. **Sinon Si** la similarité (W_j, "Superior") ≥ seuil **Alors**
      12. $Hc_{conditions} \leftarrow SA_{op(k)}(W_i) > W_{j+2}$
   13. **Fin Si**
14. **Fin Si**
15. **Fin Pour**
16. **Pour tout** (TA_{op}) **Faire**
   17. **Si** la similarité (W_i, ($TA_{op(l)}$)) ≥ seuil **Alors**
      18. $TA_{att} = TA_{op(l)}(W_i)$ //TA_{att}: attribut d'agrégation temporel
   19. **Si** la similarité (W_j, "Inferior") ≥ seuil **Alors**
      20. $Hc_{conditions} \leftarrow TA_{op(l)}(W_i) < W_{j+2}$
   21. **Sinon Si** la similarité (W_j, "Superior") ≥ seuil **Alors**
      22. $Hc_{conditions} \leftarrow TA_{op(l)}(W_i) > W_{i+2}$
   23. **Fin Si**
   24. **Fin Si**
25. **Fin Pour**
26. **Pour tout** (STA_{op}) **Faire**
   27. **Si** la similarité (W_i, ($STA_{op(m)}$)) ≥ seuil **Alors**
      28. $STA_{att} = STA_{op(m)}(W_i)$ //STA_{att}: attribute d'agrégation spatio-temporal
   29. **Si** la similarité (W_j, "Inferior") ≥ seuil **Alors**
      30. $Hc_{conditions} \leftarrow STA_{op(m)}(W_i) < W_{j+2}$
   31. **Sinon Si** la similarité (W_j, "Superior") ≥ seuil **Alors**
      32. $Hc_{conditions} \leftarrow STA_{op(m)}(W_i) > W_{i+2}$
   33. **Fin Si**
   34. **Fin Si**
35. **Fin Pour**
36. **Fin Si**
37. **Fin Pour**
38. **Fin**

La seconde heuristique (Algorithme 4) indique que si W_i est un adjectif superlatif (noté JJS) suivi d'un nom (NN/NNS), il est nécessaire d'utiliser les fonctions d'agrégation MAX() et MIN() dans une sous-requête.

Algorithme 4. Construction de la clause HAVING (Heuristique 2)

---

1. **Entrée** QU = $\{w_1, w_2, w_3 \ldots, w_n\}$, SA_{op}, TA_{op}, STA_{op}, $Wc_{conditions}$, Fc \\ Fc: la clause FROM
2. **Sortie** $Hc_{conditions}$
3. **Début**
4. **Pour tout** $W_{i \in \{1, 2, 3, n\}}$ **Faire**
5. **Si** ($Cat_g(W_i) \in$ {JJS}) suivie de ($Cat_g(W_j) \in$ {NN, NNS}) **Alors**
   6. **Si** la similarité (W_i, "Inferior") ≥ seuil **Alors**

10

7. $Hc_{conditions} \leftarrow W_j$ = Select MIN(W_j From Fc Where $Wc_{conditions}$)
8. **Sinon Si** la similarité (W_i, "Superior") ≥ seuil **Alors**
9. $Hc_{conditions} \leftarrow W_j$ = Select MAX(W_j From Fc Where $Wc_{conditions}$)
10. **Sinon**
11. **Pour tout** (SA_{op}) **Faire**
12. **Si** la similarité (W_j, ($SA_{op(k)}$)) ≥ seuil **Alors**
13. $SA_{att} = SA_{op(k)}(W_j)$
14. **Si** la similarité (W_i, "Inferior") ≥ seuil **Alors**
15. $Hc_{conditions} \leftarrow SA_{att}$ = Select MIN(SA_{att} From Fc Where $Wc_{conditions}$)
16. **Sinon si** la similarité (W_i, "Superior") ≥ seuil **Alors**
17. $Hc_{conditions} \leftarrow SA_{att}$ = Select MAX(SA_{att} From Fc Where $Wc_{conditions}$)
18. **Fin si**
19. **Fin si**
20. **Fin pour**
21. **Pour tout** (TA_{op}) **Faire**
22. **Si** la similarité (W_j, ($TA_{op(l)}$)) ≥ seuil **Alors**
23. $TA_{att} = TA_{op(l)}(W_j)$
24. **Si** la similarité (W_i, "Inferior") ≥ seuil **Alors**
25. $Hc_{conditions} \leftarrow TA_{att}$ = Select MIN(TA_{att} From Fc Where $Wc_{conditions}$)
26. **Sinon Si** la similarité (W_i, "Superior") ≥ seuil **Alors**
27. $Hc_{conditions} \leftarrow TA_{att}$ =Select MAX(TA_{att} From Fc Where $Wc_{conditions}$)
28. **Fin Si**
29. **Fin Si**
30. **Fin Pour**
31. **Pour tout** (STA_{op}) **Faire**
32. **Si** la similarité (W_j, ($STA_{op(m)}$)) ≥ seuil **Alors**
33. $STA_{att} = STA_{op(m)}(W_j)$
34. **Si** la similarité (W_i, "Inferior") ≥ seuil **Alors**
35. $Hc_{conditions} \leftarrow STA_{att}$=Select MIN($STA_{att}$ From Fc Where $Wc_{conditions}$)
36. **Sinon Si** la similarité (W_i, "Superior") ≥ seuil **Alors**
37. $Hc_{conditions} \leftarrow STA_{att}$=Select MAX($STA_{att}$ From Fc Where $Wc_{conditions}$)
38. **Fin Si**
39. **Fin Si**
40. **Fin Pour**
41. **Fin Si**
42. **Fin Pour**
43. **Fin**

3.2.4. Construction de la clause GROUP BY

En fait, pour déterminer la clause GROUP BY (notée GByc), les attributs doivent appartenir à la liste des attributs de l'instruction SELECT et différents des attributs d'agrégation (SA_{att}, TA_{att}, STA_{att}) (Algorithme 5).

Algorithme 5. Construction de la clause GROUP BY

---

1. **Entrée** $Sst_{att}L$ // La liste des attributs de l'instruction SELECT
2. **Sortie** $GByc_{att}L$ // liste des attributs de la clause GROUP BY
3. $GByc_{att}L \leftarrow \varnothing$
4. **Début**

5. **Pour tout** $W_{i \in \{1, 2, 3, n\}}$ **Faire**
   // W_i OU $Geo_{ERC}(W_i) \in Sst_{att}L$: le mot i (W_i) OU la classe du mot i ($Geo_{ERC}(W_i)$) appartient à la liste des attributs de l'instruction SELECT
   // W_i OU $Geo_{ERC}(W_i) \notin \{SA_{att}, TA_{att}, STA_{att}\}$: le mot i (W_i) OU la classe du mot i ($Geo_{ERC}(W_i)$) est différent des attributs d'agrégation (SA_{att}, TA_{att}, STA_{att})
6. **Si** (W_i OU $Geo_{ERC}(W_i) \in Sst_{att}L$) et ($W_i$ OU $Geo_{ERC}(W_i) \notin \{SA_{att}, TA_{att}, STA_{att}\}$) **Alors**
7. Ajouter le mot (W_i OU $Geo_{ERC}(W_i)$) à $GByc_{att}L$
8. **Fin Si**
9. **Fin Pour**
10. **Fin**

3.2.4. Génération de la requête finale

Les différentes clauses sont déterminées sans faire référence à une ou à plusieurs tables de base de données. Les tables sont obligatoires et font partie intégrante de la requête. L'objectif principal de cette étape est de construire la clause FROM (algorithme 6) et de réaliser la correspondance entre les attributs des clauses générées et les attributs d'une base de données géographique donnée. Pour ce faire, nous proposons d'utiliser une mesure de similarité sémantique pour assurer cette correspondance entre les attributs. Dans ce cas, nous adoptons la mesure de similarité de Wu et Palmer (1994) permettant de calculer la similarité sémantique et la relation sémantique entre les mots en considérant leur longueur de chemin dans la taxonomie WordNet (Miller, 1995). Cette mesure est basée sur Least Common Subsumer (LCS) qui représente l'ancêtre commun le plus spécifique de deux attributs :

$$Sim_{wp}(Att_1, Att_2) = \frac{2depth(LCS(Att_1, Att_2))}{depth(Att_1) + depth(Att_2)} \quad (1)$$

depth(LCS(Att_1; Att_2)) représente le nombre de relations IS-A de l'attribut le plus courant au racine. depth (Att_1) et depth (Att_2) représentent le nombre de relations IS-A à partir de chaque attribut Att_1 et Att_2 jusqu'au racine de la taxonomie (Guessoum et al., 2016).

Algorithme 6. Construction de la clause FROM

1. **Entrée** $Sst_{att}L$, $Wc_{att}L$, $Hc_{att}L$, BDG \\ $Hc_{att}L$: la liste des attributs de la clause HAVING, BDG : base de données géographique
2. **Sortie** $Fc_{att}L$ // liste des attributs de la clause FROM
3. $Fc_{att}L \leftarrow \varnothing$
4. **Début**
5. **Pour tout** Table ∈ BDG **Faire**
6. Calculer la similarité sémantique entre les attributs de la table i et les attributs de l'instruction SELECT et des clauses HAVING et WHERE ($Sst_{att}L$, $Wc_{att}L$, $Hc_{att}L$).
7. **Si** la similarité ≥ seuil **Alors**
8. Ajouter la $Table_i$ à $Fc_{att}L$
9. Remplacer les attributs des différentes clauses ($Sst_{att}L$, $Wc_{att}L$, $Hc_{att}L$) par les attributs de la $Table_i$

12

10. **Fin Si**
11. **Fin Pour**
12. **Fin**

Si plusieurs tables sont utilisées, la clause FROM (notée Fc) pourrait être une opération de jointure qui regroupe des tables appariées. En ce qui concerne la construction de GByc, nous proposons de remplacer le $GByc_{att}$ par ceux modifiés dans le Sst. Dans certains cas, les tables de base de données peuvent contenir des noms de colonnes ambigües. Il est donc difficile de trouver une similarité sémantique ou même une relation sémantique entre les attributs de requête générée et les attributs des tables de base de données. Afin de résoudre ce problème, nous proposons d'assurer la correspondance en mesurant la relation sémantique entre les attributs de (Sst, Wc, GByc, Hc) et les valeurs que peut prendre chaque attribut des tables de base de données. L'algorithme 7 présente les détails de ce processus.

Algorithme 7. Correspondance entre les attributs de la requête (ST-SQL) et les valeurs que chaque attribut des tables de la BDG peut prendre

1. **Entrée** $Sst_{att}L$, $Fc_{att}L$, $Wc_{att}L$, $GByc_{att}L$, $Hc_{att}L$, V_1Att_i // V_1Att_i: La première valeur associée à chaque attribut de la $Table_i$.
2. **Sortie** ST-SQL \\ SQL spatio-temporelle
3. **Début**
4. **Pour tout** Table ∈ BDG **Faire**
5. Calculer la relation sémantique entre les attributs de ($Sst_{att}L$, $Wc_{att}L$, $GByc_{att}L$, $Hc_{att}L$) et V_1Att_i
6. **Si** relation sémantique ≥ seuil **Alors**
7. Remplacer les attributs de ($Sst_{att}L$, $Wc_{att}L$, $GByc_{att}L$, $Hc_{att}L$) avec les attributs de la $Table_i$
8. Formuler la requête ST-SQL finale utilisant ($Sst_{att}L$, $Fc_{att}L$, $Wc_{att}L$, $GByc_{att}L$, $Hc_{att}L$)
9. **Fin Si**
10. **Fin Pour**
11. **Fin**

4. Résultats de simulation

Notre outil permet aux utilisateurs des SIG d'accéder facilement aux données en utilisant le langage naturel comme un moyen de communication. Dans ce travail, nous nous sommes intéressés par des données recueillies à partir de la base de données NATural DISasters (NATDIS)[1]. Ce dernier comprend des informations sur les catastrophes naturelles dans le monde entier depuis janvier 2001. Le processus de prétraitement est déclenché une fois qu'un utilisateur introduit sa question. Ce processus consiste à exploiter les techniques de TALN. À cette fin, nous avons utilisé le Stanford CoreNLP (Manning et al., 2014) pour assurer l'étiquetage morphosyntaxique, extraire les relations de dépendance et reconnaître les entités et les relations géographiques. Plus précisément, pour assurer la reconnaissance des

1. http://www.catnat.net

entités et des relations géographiques, nous avons adopté le Framework TokensRegex (Chang et Manning, 2014) inclus dans Stanford CoreNLP (Manning et al., 2014). Ainsi, ce framework fournit des annotations basées sur des expressions régulières. TokensRegex a été utilisé pour développer SUTime (Chang et Manning, 2012). Ce dernier est en fait basé sur un ensemble de règles utilisées pour assurer la détection des expressions temporelles. Une fois la question est prétraitée, le système est capable de traduire cette question en une requête spatio-temporelle en se basant sur un ensemble d'heuristiques (figure 2).

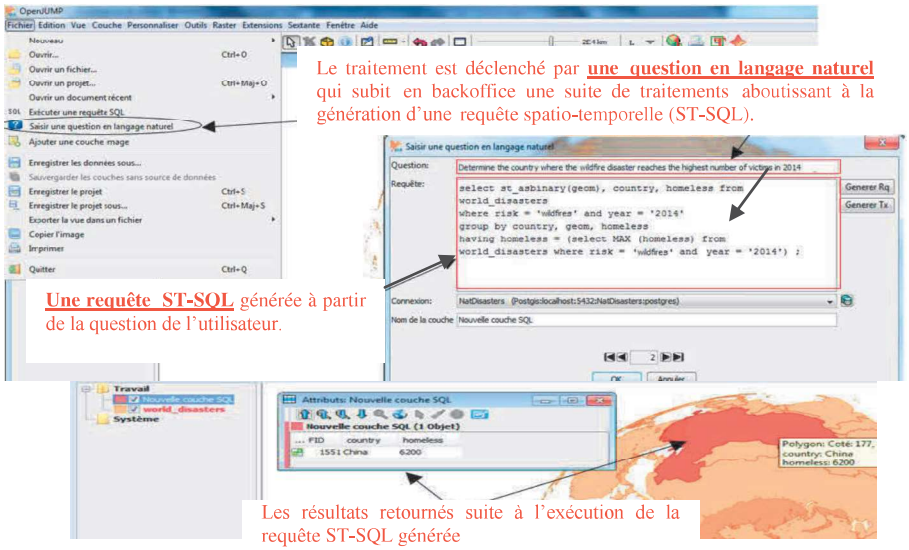

Figure 2. Génération d'une requête spatio-temporelle (ST-SQL)

5. Evaluation

Afin d'évaluer la performance de notre système à traduire une question en langage naturel sous une forme structurée à savoir une requête spatio-temprelle (ST-SQL) nous proposons d'utiliser les deux mesures bien connues : le Rappel et la Précision. En fait, nous proposons d'évaluer si les requêtes spatio-temporelles représentent bien la question. De ce fait, nous avons fourni un nombre de questions à des experts qui maitrisent bien le langage de requête SQL et plus précisément les requêtes géographiques pour qu'ils génèrent les requêtes adéquates. Dans ce cas, les résultats des requêtes générées par notre système bien évidement pour les mêmes questions seront comparés aux résultats retournés par les requêtes générées par les experts. En fait, quand un utilisateur introduit sa question, le système peut fournir différents types de résultats : soit des requêtes spatio-temporelles qui fournissent des réponses correctes en les comparant aux réponses des requêtes générées par les experts, soit des requêtes qui fournissent des réponses incorrectes à cause d'une mal traduction de la question en ST-SQL, soit, le système ne peut pas assurer la traduction de la question en requête spatiotemporelle. Notre système ne peut prendre

en considération que deux types de requêtes qui sont les requêtes spatio-temporelles simples et les requêtes spatio-temporelles d'intervalles. Les requêtes les plus sophistiquées ne sont pas encore prises en compte. Dans ce contexte, les deux mesures de rappel et de précision peuvent êtres définis comme suit : le rappel peut être défini comme étant le nombre de requêtes fournissant des réponses correctes, rapporté au nombre de requêtes spatio-temporelles générées. La précision représente le nombre de requêtes générées fournissant des réponses correctes, rapporté au nombre total de questions proposées.

$$\text{Rappel} = \frac{\text{Nombre de requêtes fournissant des réponses correctes}}{\text{Nombre de requêtes (ST-SQL) générées}} \quad (2)$$

$$\text{Précision} = \frac{\text{Nombre de requêtes fournissant des réponses correctes}}{\text{Nombre total de questions proposées}} \quad (3)$$

Dans le même contexte, nous avons adopté une mesure supplémentaire utilisée pour évaluer l'efficacité du système appelée Willingness (Minock, 2010). Cette mesure peut être définie comme étant le nombre de requêtes (ST-SQL) fournissant des réponses correctes et incorrectes, rapporté au nombre total de questions.

$$\text{Willingness} = \frac{\text{Nombre de requêtes fournissant des réponses correctes et incorrectes}}{\text{Nombre total de questions proposées}} \quad (4)$$

En se basant sur ces mesures, un ensemble de questions formulées en langage naturel (300 questions) ont été proposé. Les requêtes générées sont vérifiées manuellement, alors nous avons constaté différents types de réponses sont retournées lors de l'exécution de ces requêtes. Les résultats sont donnés dans le tableau suivant :

Tableau 1. Résultats expérimentaux

Le nombre total de questions	300
Le nombre de requêtes spatio-temporelles générées par notre système	269
Nombre de questions que le système est incapable de les traduire en requêtes spatio-temporelles	31
Nombre de requêtes fournissant des réponses correctes	232
Nombre de requêtes fournissant des réponses incorrectes	37
Précision	86.2%
Rappel	77.3%
Willingness	89%

D'après les taux de rappel (77.3%) et de précision (86.2%) obtenus, et en comparaison avec les requêtes spatio-temporelles générées par des experts nous pouvant constater que notre système a réussi à atteindre des résultats satisfaisants. Ainsi, le taux de Willingness obtenu est environ 89%, ce qui confirme la performance et l'efficacité de notre système.

5. Conclusion

Dans ce présent article, nous avons proposé une approche de génération de requêtes spatio-temporelles à partir des questions en langage naturel. D'abord une analyse profonde de ces questions est assurée afin de les traduire en un format structuré. Pour ce faire, nous avons eu recours à des techniques de TALN permettant le prétraitement de la question. Ensuite, nous avons proposé un ensemble d'heuristiques sur lesquelles est basé notre processus de génération des requêtes ST-SQL. Ainsi, une mesure de similarité est utilisée afin d'assurer la correspondance entre la question et la requête. Les expérimentations ont approuvé la validité de notre approche. Notamment, nous avons prouvé, que le fait d'exprimer des besoins d'information utilisant le langage naturel plutôt qu'un langage formel, pouvait évidemment fournir des résultats d'une qualité satisfaisante en assurant un confort supérieur pour le grand public. En guise de perspectives, notre approche peut être améliorée en ajoutant d'autres clauses permettant d'améliorer la recherche de l'information. Ainsi, nous envisageons d'étudier d'autres classes de questions spatio-temporelles, à savoir : les requêtes de comportement spatio-temporel et les requêtes de relation spatio-temporelles.

Références

Abacha A. B., (2012). *Recherche de réponses précises à des questions médicales : le système de questions-réponses MEANS.* Thèse en Informatique. Université Paris Sud - Paris XI.

Alexander R., Rukshan P., Mahesan S., (2013). Natural Language Web Interface for Database (NLWIDB). *In CoRR*

Allen J. F. (1983). Maintaining knowledge about temporal intervals. *Communications of the ACM*, p. 832–843.

Androutsopoulos I., Ritchie G., Thanisch P., (1995). Natural Language Interfaces to Databases - An Introduction. *In Journal of Natural Language Engineering*, vol 1, p 29–81.

Chang A. X., Manning C. D. (2012). Sutime: A library for recognizing and normalizing time expressions. *In: 8th International Conference on Language Resources and Evaluation*.

Chang, A. X., Manning C. D., (2014). *Tokensregex: Defining cascaded regular expressions over tokens,* Stanford University Technical Report.

Chaudhari P. P. (2013). Natural Language Statement to SQL Query Translator. *In International Journal of Computer Applications,* p. 18–22.

Chen W. (2014). Parameterized Spatial SQL Translation for Geographic Question Answering. *In Semantic Computing (ICSC), 2014 IEEE International*, p. 23–27

Deshpande A. K., Devale P. R., (2012). Natural Language Query Processing Using Probabilistic Context Free Grammar. *In International Journal of Advances in Engineering and Technology*, vol 3, p. 568–573.

Ferraro, J. P., Daumé H., DuVall S. L., Haug P. J., Harkema H., Chapman W., (2013). Improving performance of natural language processing part-of-speech tagging on clinical

narratives through domain adaptation. *Journal of the American Medical Information Association.*

Gesbert N. (2004). Formalisation des spécifications de bases de données géographiques pour une meilleure compréhension des données. *XXIIème congrès INFORSID*, p. 25–28.

Giordani A., Moschitti A., (2009). Semantic Mapping Between Natural Language Questions and SQL Queries via Syntactic Pairing. *In Natural Language Processing and Information Systems*, p. 207–221.

Giordani A., Moschitti A., (2012). Translating Questions to SQL Queries with Generative Parsers Discriminatively Reranked. *In COLING 2012: Posters*, p. 401–410.

Guessoum D., Miraoui M, Tadj C., (2016). A modification of wu and palmer semantic similarity measure. *In the Tenth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies.*

Manning, C. D., Surdeanu M, Bauer J., (2014). The stanford corenlp natural language processing toolkit. *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations.*

Marneffe M. C., MacCartney B., Manning C. D., (2006). Generating typed dependency parses from phrase structure parses. *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC).*

Mikolov T., Sutskever I., Chen K., Corrado G., Dean J. (2013). Distributed representations of words and phrases and their compositionality. *In Proceedings of the 26th International Conference on Neural Information Processing Systems, NIPS13*, p. 3111–3119, USA.

Miller G. A. (1995). Wordnet: A lexical database for english. *Communications of the ACM.*

Minock M. (2010). C-Phrase: A system for building robust natural language interfaces to databases. *Journal Data and Knowledge Engineering*, p. 290-302.

Popescu A. M., Etzioni O., Kauta H., (2003). Towards a Theory of Natural Language Interfaces to Databases. *In Proceedings of the 8th International Conference on Intelligent User Interfaces*, p. 149–157.

Rao G., Agarwal C., Chaudhry S., Kulkarni N., Patil D. S., (2010). Natural Language Query Processing using Semantic Grammar. *In International Journal on Computer Science and Engineering,* p. 219–223.

Raza A. (2008). *Deriving spatiotemporal relations from simple data structure.* The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences.

Worboys M. F. (1992). A generic model for planar geographical objects. *International Journal of Geographical Information Systems.*

Wu Z., Palmer M., (1994). Verb semantics and lexical selection. *In 32nd Annual Meeting of the Association for Computational Linguistics.*

Yaghmazadeh N., Wang Y., Dillig I., Dillig T. (2017). Type and content-driven synthesis of sql queries from natural language.

Yuan M., McIntosh J., (2002). A typology of spatiotemporal information queries. *The Springer International Series in Engineering and Computer Science,* Springer, Boston.