Une approche combinée pour l'enrichissement d'ontologie à partir de textes et de données du LOD

Céline Alec*, Chantal Reynaud-Delaître*, Brigitte Safar*

*LRI, Univ. Paris-Sud, CNRS, Université Paris-Saclay, Orsay, F-91405
prenom.nom@lri.fr

Résumé. Cet article porte sur l'étiquetage automatique de documents décrivant des produits, avec des concepts très spécifiques traduisant des besoins précis d'utilisateurs. La particularité du contexte est qu'il se confronte à une triple difficulté : 1) les concepts utilisés pour l'étiquetage n'ont pas de réalisations terminologiques directes dans les documents, 2) leurs définitions formelles ne sont pas connues au départ, 3) toutes les informations nécessaires ne sont pas forcément présentes dans les documents mêmes. Pour résoudre ce problème, nous proposons un processus d'annotation en deux étapes, guidé par une ontologie. La première consiste à peupler l'ontologie avec les données extraites des documents, complétées par d'autres issues de ressources externes. La deuxième est une étape de raisonnement sur les données extraites qui recouvre soit une phase d'apprentissage de définitions de concepts, soit une phase d'application des définitions apprises. L'approche SAUPODOC est ainsi une approche originale d'enrichissement d'ontologie qui exploite les fondements du Web sémantique, en combinant les apports du LOD et d'outils d'analyse de texte, d'apprentissage automatique et de raisonnement. L'évaluation, sur deux domaines d'application, donne des résultats de qualité et démontre l'intérêt de l'approche.

1 Introduction

Ce travail se situe dans le cadre d'un partenariat entre le LRI et la startup Wepingo [1], qui développe des applications en ligne proposant des produits à des internautes. Pour faciliter la conception de systèmes flexibles, adaptables à différentes catégories de produits et à différents points de vue sur ces produits, notre objectif est de concevoir une approche permettant d'étiqueter automatiquement des documents décrivant des produits, avec des concepts très spécifiques traduisant des besoins précis des utilisateurs. La particularité de notre approche est qu'elle se confronte à une triple difficulté : 1) les concepts utilisés pour l'étiquetage n'ont pas de réalisations terminologiques directes dans les documents, 2) les définitions formelles de ces concepts ne sont pas connues au départ même si le concepteur du système sait à partir de quelles informations elles pourraient être construites, 3) toutes les informations nécessaires ne sont pas forcément présentes dans les documents mêmes.

1. http://www.wepingo.com/fr-fr/

Par exemple dans le domaine du Tourisme, l'expression "Destination où l'on peut pratiquer des sports nautiques (WaterSport) durant l'hiver (Winter)" correspond à une demande fréquente des utilisateurs. Ce concept, dénoté DWW dans la suite de l'article, est très spécifique et n'apparaît jamais tel quel dans les documents décrivant des lieux touristiques. En revanche, le concepteur du système sait que le concept fait référence à un lieu suffisamment chaud en hiver et à des activités nautiques. Il sait aussi qu'il peut trouver dans les documents, des termes faisant référence aux activités nautiques mais qu'il n'y trouvera pas les valeurs des températures en hiver, et qu'il lui faudra les rechercher dans une ressource externe. Une fois les informations nécessaires acquises, il pourra alors étiqueter une description comme étant ou non une instance du concept DWW. L'automatisation du processus demande d'expliciter formellement la définition dans un langage adapté, ce qui peut être délicat à réaliser (qu'est-ce qu'un lieu suffisamment chaud ?). En revanche, elle peut être apprise par un outil automatique si le concepteur du système peut répartir un certain nombre de descriptions en exemples positifs ou négatifs du concept. Une fois la définition apprise, de nouvelles descriptions touristiques vérifiant cette définition peuvent être automatiquement annotées comme des instances du concept et utilisées dans l'application, pour répondre à des requêtes d'un utilisateur.

Le fait de connaître explicitement la définition donne de la flexibilité au système. Si aucune destination ne correspond exactement à l'ensemble des contraintes émises par un utilisateur particulier, relâcher certaines conditions de la définition permet au système de proposer des réponses approchées : un utilisateur recherchant une DWW pourra éventuellement se satisfaire d'une destination où la température est légèrement plus basse.

La contribution de cet article est donc une approche originale d'enrichissement d'ontologie qui combine différents outils d'analyse de texte et du LOD (Linked Open Data), d'apprentissage automatique et de raisonnement, dans un contexte triplement contraint. Le reste de l'article est organisé comme suit. La section 2 présente l'état de l'art. La section 3 décrit l'approche, ses entrées et les différentes tâches invoquées. La section 4 présente les expérimentations réalisées. Nous concluons en section 5 sur les futures directions de travail.

2 Travaux proches

L'enrichissement d'ontologie est un vaste champ de recherche dans lequel nous distinguerons trois catégories de travaux, portant principalement sur l'extraction de connaissances sémantiques à partir de textes plus ou moins structurés.

La première catégorie concerne les travaux sur les ontologies expressives et la génération de définitions de concepts. Certaines approches travaillent sur des textes décrivant des concepts. LExO (Völker et al., 2007), par exemple, applique des règles de transformation syntaxiques sur des définitions en langue naturelle pour générer des axiomes en Logique de Description (LD). Ma et Distel (2013b) ont eux une approche basée sur l'extraction de relations et s'appuient sur des contraintes formelles pour assurer la qualité des définitions apprises (Ma et Distel, 2013a). Ces approches ne sont pas appliquables sur les documents que nous traitons car ceux-ci ne contiennent que des descriptions d'instances. D'autres, comme (Chitsaz, 2013) et (Lehmann et Hitzler, 2010), ne disposent comme nous que de descriptions d'instances. Ils s'appuient sur la programmation logique inductive pour trouver de nouvelles descriptions de concepts à partir des assertions d'une ontologie. (Lehmann et Hitzler, 2010) s'applique à des ontologies expressives en LD, (Chitsaz, 2013) à des ontologies légères et les deux nécessitent

un grand nombre d'assertions relatives aux instances. En comparaison, nos entrées sont des textes incomplets et très peu structurés dont il faut extraire les assertions.

La seconde catégorie de travaux concerne la génération d'ontologies légères à l'expressivité limitée, souvent réduites à des taxonomies. Ils étudient comment extraire différents éléments ontologiques à partir de ressources textuelles (Cimiano, 2006). Pour l'extraction de concept, l'étape primordiale est l'extraction de la terminologie pertinente du domaine (Cimiano et al., 2006) en utilisant différentes mesures de pondération des termes. Les techniques de classification sont ensuite appliquées pour détecter les synonymes et une classe ontologique peut être dérivée pour chaque groupe de termes similaires. D'autres s'intéressent à l'apprentissage de hiérarchies de concepts. Ils utilisent principalement des techniques de classification hiérarchiques non supervisées afin d'apprendre en même temps les concepts et leurs relations de subsomption (Cimiano, 2006). Enfin, lorsque l'ontologie n'a pas à être intégralement construite et qu'une hiérarchie de concepts existe déjà et doit être étendue avec de nouveaux concepts, les méthodes supervisées deviennent possibles. Des classifieurs doivent être entrainés pour chaque concept de l'ontologie existante qui ne doit pas être très grande. Des mesures de similarité appropriées sont ensuite utilisées pour comparer un nouveau concept avec ceux déjà existants (Cimiano et Völker, 2005). Tous ces travaux visent à reconnaître les termes désignant des concepts (ou des instances) dans les textes, puis à les extraire. Cependant, parfois, les textes n'évoquent que les propriétés des instances sans nommer le concept sous-jacent, comme dans notre travail. D'autres approches, comme celles présentées ci-dessous, sont alors nécessaires.

La troisième catégorie inclut les travaux qui utilisent le raisonnement pour remplacer partiellement les techniques traditionnelles d'extraction. Dans le système BOEMIE (Petasis et al., 2013), les concepts sont divisés en concepts primitifs et composites, ces derniers étant définis à partir des premiers. Les concepts primitifs sont peuplés avec des outils d'extraction classiques. Les instances des concepts composites ne sont pas explicitement présentes dans les textes mais leurs propriétés le sont. Les concepts composites sont donc peuplés par raisonnement sur les propriétés extraites et sur les instances de concepts primitifs. Yelagina et Panteleyev (2014) extraient des faits de textes grâce à des outils de traitement automatique de la langue et à une ontologie. À partir de ces faits et de connaissances et règles d'inférences introduites au préalable, ils peuvent dériver de nouveaux faits, non mentionnés dans le texte. Notre travail est proche des deux derniers cités mais se différencie par le fait que nous ne disposons pas des définitions des concepts à peupler.

Cet état de l'art montre qu'aucune des approches prises isolément n'est une solution à notre problème. Le paragraphe suivant montre comment l'approche SAUPODOC en combine certaines pour s'adapter à notre contexte.

3 L'approche SAUPODOC

Notre approche, nommée SAUPODOC (Semantic Annotation Using Population of Ontology and Definition Of Classes) est automatique, générique et travaille sur trois entrées : (1) une liste de termes nommés *Classes Cibles* ou CC, qui traduisent des besoins des utilisateurs et qu'il faut définir, (2) un corpus regroupant des instances de ces classes sous la forme de descriptions textuelles, annotées par le concepteur du système comme des instances positives ou négatives de chaque CC et (3) une ontologie relative au domaine considéré, supposée définie par ailleurs, et que le concepteur aura éventuellement enrichie des propriétés qu'il aura identifiées comme devant intervenir dans les définitions à apprendre.

Les paragraphes suivants présentent ces entrées, l'enchaînement des tâches mises en œuvre dans l'approche et les différents composants qui les réalisent.

3.1 Les entrées de l'approche

- **L'ontologie :** elle définit le domaine considéré. Elle est un guide pour l'analyse des documents, la recherche des informations complémentaires et le raisonnement sur les annotations obtenues. Cela implique qu'elle contient tous les éléments définissant les entités du domaine d'application. Elle est spécifique au domaine mais pas à l'approche et sa constitution n'est pas le sujet de l'article. Plus formellement, l'ontologie $\mathcal{O}$ est une ontologie OWL définie comme un tuple ($\mathcal{C}$, $\mathcal{P}$, $\mathcal{I}$, $\mathcal{A}$) où $\mathcal{C}$ est l'ensemble des classes, $\mathcal{P}$ l'ensemble des propriétés (datatype, object et annotation) caractérisant les classes, $\mathcal{I}$ un ensemble d'instances et d'assertions de propriété, et $\mathcal{A}$ un ensemble d'axiomes.

$\mathcal{C}$ regroupe ***la classe principale*** qui correspond au type général d'entités considérées et ***les classes descriptives*** qui sont toutes les autres classes utilisées pour définir la classe principale.

$\mathcal{P}$ est l'ensemble des propriétés caractérisant les classes. $\mathcal{A}$ contient les contraintes sur les propriétés : subsomption, équivalence, type, domaine/co-domaine, caractéristiques (fonctionnelle, transitive, etc), disjonction. $\mathcal{I}$ ne contient au départ que des instances des classes descriptives. Par exemple, _rainForest est une instance de Forest (descendant de Environment) et dense forest est un de ses labels.

La figure 1 présente un extrait de l'ontologie sur les destinations touristiques où la classe principale est Destination. Les classes descriptives Activity, Environment, FamilyType et Season sont respectivement des racines de hiérarchie. Ainsi, Environment représente l'environnement naturel (Aquatic, Desert, etc) ou ses qualités (Beauty, View). Certaines des propriétés inter-classes ont des sous-propriétés non présentées ici. Les principaux attributs apparaissent sous les classes.

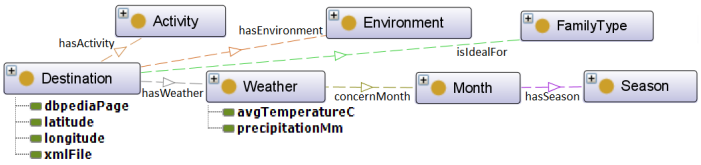

FIG. 1 – *Structure de l'ontologie Destination*

- **Les Classes Cibles (CC) :** ce sont de simples noms de concept comme "Destination_Sports_Nautiques_Hiver" (DWW), listés par le concepteur du système et qui seront introduits comme des spécialisations de la classe principale, sans définition. L'ensemble des CC ne forme pas une partition, un produit pouvant être une instance de plusieurs CC.

- **Un corpus annoté :** ce sont des documents XML très peu structurés décrivant chacun une instance particulière de la classe principale. La structuration fait apparaître d'une part le nom de l'instance et d'autre part, la description textuelle qui contient des labels d'instances des classes descriptives de l'ontologie mais pas de réalisations lexicales des classes cibles. Chaque texte est annoté manuellement par le concepteur du système, comme décrivant une instance positive ou négative de chaque CC. Notons que dans notre contexte, les documents sont soit extraits de catalogues publicitaires (et ils vantent les vertus des entités décrites), soit de très

courtes descriptions. Dans tous les cas, peu d'expressions négatives sont présentes.

3.2 Description fonctionnelle

L'approche SAUPODOC est basée sur quatre tâches guidées par l'ontologie. Les deux premières consistent à peupler l'ontologie (étape 1) en extrayant les données associées aux entités. Les deux suivantes sont des tâches de raisonnement effectuées alternativement sur cette ontologie peuplée (étape 2) : soit la découverte des définitions formelles des classes cibles (étape 2a) soit le peuplement de ces classes cibles ou annotation des documents (étape 2b). La figure 2 décrit cet enchaînement de tâches où l'ontologie initiale O est progressivement peuplée. Premièrement, en utilisant les descriptions textuelles des entités du corpus et des ressources externes, les instances de la *classe principale* représentant les entités du corpus et les données associées à ces instances sont introduites (O^{+}). Puis les *classes cibles* CC sont insérées comme des spécialisations de la *classe principale* dans l'ontologie (O^{++}) et les *définitions cibles* sont apprises en s'appuyant sur les exemples annotés manuellement. Finalement, les définitions sont appliquées ainsi les CC sont peuplées (O^{+++}) et le corpus annoté avec les CC.

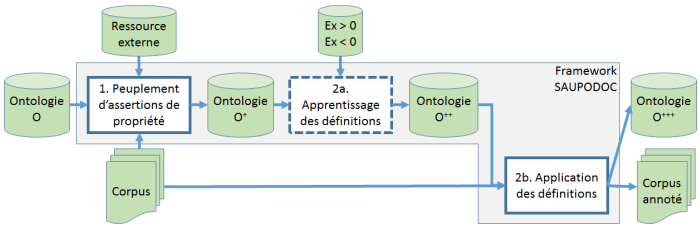

FIG. 2 – *L'enchaînement des tâches de* SAUPODOC

3.3 Les tâches de SAUPODOC

L'approche implique des tâches variées qui exploitent les données à différents niveaux d'abstraction (classes et instances) et qui doivent coopérer pour atteindre l'objectif final. Le rôle de pivot de l'ontologie est donc central dans l'approche. Une tâche préliminaire crée, pour chaque document, une instance de la classe principale représentant l'entité décrite. Ainsi, à partir du document décrivant la République dominicaine, l'instance <Dominican_Republic rdf :type Destination> est créée. Pour chaque entité, les deux tâches de l'étape 1 permettent de peupler l'ontologie avec les informations qui seront utilisées par les deux tâches de raisonnement de l'étape 2. Dans ce qui suit, nous présentons comment l'ontologie guide chacune de ces tâches.

3.3.1 Acquisition de données à partir des documents

Dans la première étape, des données sont extraites des descriptions textuelles des entités par une tâche d'annotation de documents dirigée par l'ontologie du domaine. Nous avons choisi d'utiliser le logiciel d'annotation GATE (Cunningham et al., 2011; Bontcheva et al., 2004), car il réalise différentes tâches d'analyse de texte en permettant de choisir l'ontologie qui le guidera alors que d'autres outils comme Open Calais (`http://www.opencalais.com/`) ne le permettent pas. Par exemple dans la figure 3, qui présente un court extrait du document

décrivant la destination République dominicaine, les termes scuba divers et diving sont associés par GATE à l'individu _diving de l'ontologie, instance d'un concept spécialisant WaterSport.

Les assertions de propriété relatives à chaque entité instance de la classe principale sont ensuite recherchées parmi les annotations obtenues. Pour ce faire, le langage JAPE (Java Annotation Patterns Engine) utilisable avec GATE, permet entre autres, de bâtir des règles transformant les annotations en assertions de propriété dans l'ontologie. Nous avons construit un patron générique, utilisable pour toute ontologie, automatiquement transformable en autant de règles JAPE que de propriétés de l'ontologie devant être peuplées. C'est au concepteur du système de préciser dans l'ontologie, quelles propriétés doivent être peuplées par les règles JAPE. Par exemple, hasActivity doit être considérée et pas hasWeather. Pour chaque propriété, le processus est guidé par les contraintes sur les co-domaines exprimées dans l'ontologie. Par exemple, la contrainte <Destination, hasActivity, Activity> impose que la valeur de co-domaine prise par la propriété hasActivity appartienne à l'extension de la classe Activity. À partir de cette contrainte, si la description d'une entité e contient une annotation correspondant à une instance a d'Activity, alors l'assertion <e, hasActivity, a> est construite. Dans l'exemple figure 3 ci-dessus, comme _diving est une instance d'un concept spécialisant WaterSport (lui-même sous-classe d'Activity), l'assertion <Dominican_Republic, hasActivity, _diving> est ajoutée.

especially loved by scuba divers. Over 20 exiting diving sites and 3 old shipwrecks are waiting to be discovered.

FIG. 3 – *Extrait du document sur la République dominicaine annoté par GATE*

Dans notre contexte où les documents vantent les atouts des entités décrites et donc ne contiennent pas (ou peu) d'expressions négatives, ce simple processus de peuplement apparaît suffisant.

3.3.2 Complétion des données par des ressources externes

Les descriptions textuelles sont souvent courtes et ne contiennent pas toutes les informations nécessaires. Par exemple, définir une DWW demande de connaître les températures et les précipitations par saison ou par mois pour chaque destination. Ces données n'apparaissant pas dans les descriptions, la collecte doit être enrichie en exploitant les ressources disponibles sur le Web. Ici encore, cette tâche est guidée par l'ontologie. Elle implique de trouver une ressource RDF relative aux entités du corpus et d'identifier dans cette ressource quelles données correspondent à celles requises par l'ontologie.

Nous avons choisi de travailler avec DBpedia et nous utilisons DBpedia Spotlight (Mendes et al., 2011), un outil permettant d'annoter automatiquement dans un texte les références à des entités de DBpedia. Appliqué sur le nom d'instance de chaque document du corpus, il donne un accès direct à la page DBpedia représentant l'entité du document.

Les vocabulaires de l'ontologie et de la ressource pouvant différer, des mécanismes (succinctement détaillés ici faute de place) doivent être conçus pour établir des mises en correspondance entre les éléments requis et ceux de la ressource et pour effectuer les requêtes utiles. Des mécanismes complexes ont été mis en place pour traiter les cas où par exemple une propriété source (de l'ontologie) est traduite par plusieurs propriétés cibles équivalentes avec des syntaxes différentes. Ainsi, la propriété precipitation_in_January est représentée par 6 propriétés différentes dans DBpedia (janPrecipitationMm, janRainMm, janPrecipitationInch, janRainInch, janPrecipitationIn et janRainIn). D'autres mécanismes non cités ici, permettent d'établir des correspondances avec des propriétés cibles calculées ou obtenues suite à un processus d'agrégation.

Nous avons aussi pris en compte l'incomplétude de DBpedia et donc l'absence de valeur pour certaines propriétés. En effet, dans notre contexte d'apprentissage de définitions à partir d'exemples, les données extraites pour ces exemples doivent être les plus complètes possibles, car la qualité des définitions apprises en dépend. Nous avons donc élaboré un mécanisme permettant, quand une propriété est manquante sur une page, d'explorer les pages proches dans le graphe afin d'obtenir une valeur qui puisse servir d'approximation acceptable. La figure 4 montre ainsi deux exemples (Alaska et Cephalonia) où la valeur de la propriété relative aux précipitations de janvier n'est pas sur la page de la destination elle-même mais où une valeur approximative peut être trouvée sur une page proche (celle de la capitale).

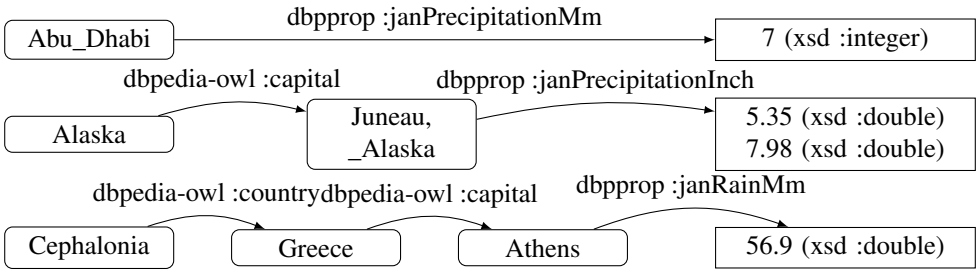

FIG. 4 – *Chemins d'exploration dans DBpedia*

Ce mécanisme est basé sur la composition de propriétés et permet au concepteur d'établir des chemins d'exploration alternatifs et ordonnés du graphe DBpedia, jusqu'à atteindre des pages contenant l'information requise. Des requêtes SPARQL de type Construct basées sur ces spécifications permettent de recueillir les données et de les insérer dans l'ontologie sous forme d'assertions de propriété.

3.3.3 Apprentissage des définitions cibles

La première phase de l'étape 2 est une étape de raisonnement et n'est exécutée qu'une seule fois. Elle a pour but de découvrir les *définitions cibles* des CC en s'appuyant sur les documents du corpus annotés comme des exemples positifs ou négatifs des différentes CC et les données collectées dans l'étape 1.

La plupart des outils d'apprentissage ne prennent pas en compte dans leurs représentations des exemples, les relations explicitées dans une ontologie (subsomption, propriétés établies entre les classes). Nous avons donc choisi d'utiliser DL-Learner (Lehmann, 2009) (version 1.0), car c'est le seul logiciel en libre accès utilisant une ontologie en entrée pour apprendre des définitions de classes exprimées en Logique de Description. Il nous permet d'obtenir les définitions explicites de chacune des CC ce qui est un atout important dans les applications concrètes.

Les définitions construites par DL-Learner sont des conjonctions ou des disjonctions d'éléments. Un élément peut être une classe (Destination) ou une expression concernant une propriété (hasActivity some Nightlife), un attribut à valeur numérique (avgTemperatureC some double[>= 23.0]), ou une contrainte de cardinalité (hasCulture min 3 Culture). Les co-domaines peuvent aussi être des conjonctions ou des disjonctions d'éléments.

La définition de la classe DWW peut alors être apprise comme suit :

```
(Destination and (hasActivity some Watersport)
        and (hasWeather min 2 ((concernMonth some (hasSeason some MidWinter))
                        and (avgTemperatureC some double[>= 23.0])
                        and (precipitationMm some double[<= 70.0])))).
```

Pour paramètrer DL-Learner, nous avons utilisé l'algorithme CELOE (Lehmann et al., 2011) annoncé comme le meilleur pour l'apprentissage de classes, et le raisonneur par défaut qui utilise l'hypothèse du monde clos (CWA). En revanche nous avons désactivé les constructeurs de négation (NOT) et de restriction universelle (ONLY) car les définitions apprises devaient être introduites dans l'ontologie OWL et le raisonnement dans celle-ci se fait avec l'hypothèse du monde ouvert (OWA). De plus, pour pouvoir apprendre et exploiter des contraintes de cardinalité minimum comme (hasActivity min 3 Activity) les instances ont été automatiquement explicitées comme disjointes (Unique Name Assumption) pour ne pas être supposées reliables par un owl :sameAs. En revanche, comme le raisonneur ne peut pas travailler en monde ouvert avec des définitions contenant des restrictions de cardinalités maximum, celles-ci ont été ignorées, i.e., nous avons automatiquement fait retenir la meilleure définition ne contenant pas ce type de restriction. En plus de la configuration de base, correspondant aux paramètres décrits ci-dessus, nous avons défini une configuration dite complexe non détaillée ici, qui permet d'apprendre d'éventuelles longues définitions comme celle attendue pour DWW.

Un dernier paramètre important de DL-Learner est le bruit, i.e., le pourcentage d'exemples positifs non couverts par une définition. Nous avons procédé par essais erreurs pour le fixer et avons établi une méthodologie à partir des expériences menées. Pour chaque CC, 10 configurations ont été testées : les configurations de base et complexe, chacune avec 5 valeurs différentes de bruit (5-15-25-35-45%). Pour chaque configuration, la solution la mieux classée par DL-Learner en termes d'exactitude et de taille a été retenue, puis pour chaque CC, la définition choisie est la meilleure des 10.

3.3.4 Étape de raisonnement

La seconde phase de l'étape 2 consiste à appliquer les définitions apprises pour peupler les CC dans l'ontologie. Elle est exécutée chaque fois que de nouvelles descriptions doivent être annotées. Nous avons choisi d'utiliser FaCT++ (Tsarkov et Horrocks, 2006), un raisonneur OWL-DL disponible et efficace même sur un grand nombre d'instances. FaCT++ s'appuie sur les définitions des CC pour identifier les entités des descriptions qui les vérifient. Pour chaque classe cible cc, si l'entité décrite dans un document d est reconnue comme une instance de cc, le document d est annoté par cc sinon il est annoté par $not\ cc$. En faisant cela, nous simulons un raisonnement en monde clos (CWA) alors que OWL raisonne en monde ouvert (OWA). Mais de fait, notre contexte particulier nous permet de simuler le CWA dans toutes les étapes. Ainsi, comme nous l'avons dit dans la section 3.3.3, les définitions sont construites dans l'hypothèse du monde clos, à partir des assertions de propriété extraites des exemples. Pour une entité, si une propriété ne peut pas être extraite, nous considérons qu'elle n'existe pas. Par exemple, si une description de lieu ne mentionne pas de plage, nous sommes sûrs qu'il n'y a pas de plages, car les documents sont supposés mentionner toutes les caractéristiques positives des lieux. De même, pour pallier l'incomplétude des données de DBpedia, nous avons défini un

modèle d'exploration qui permet de remplacer les valeurs de propriétés manquantes par des valeurs approchées et d'avoir toutes les données requises.

4 Evaluation expérimentale

Nous avons comparé l'approche SAUPODOC avec deux approches de classification l'une basée sur SVM et l'autre sur un arbre de décision. Les expérimentations ont été faites sur deux domaines d'application décrits ci-dessous.

4.1 Matériels

4.1.1 Le domaine des destinations touristiques

Le corpus des destinations touristiques est petit (80 documents), ce qui rend possible une vérification manuelle des assertions trouvées. Chaque document a été automatiquement extrait à partir du catalogue de Thomas Cook (`http://www.thomascook.com/`) et décrit une destination particulière (pays, région, île ou ville). Les documents sont promotionnels, i.e. ils mettent en avant les qualités des destinations et contiennent très peu d'expressions négatives.

L'ontologie du domaine comprend une classe principale, Destination, et 161 classes descriptives. Celles-ci sont utilisées pour caractériser la nature de l'environnement (46 classes), les activitées possibles (102 classes), le type de familles concernées, e.g. avec enfants, couples, etc (6 classes) et des classes relatives aux températures comme les saisons (7 classes). Les classes descriptives contiennent des instances et leurs formes terminologiques pour faciliter leur identification dans les textes. Par exemple, les termes archaeology, archaeological, acropolis, roman villa, excavation site, mosaic sont associés à l'instance archaeology. 39 CC ont été introduites.

4.1.2 Le domaine des films

Le corpus des films contient 10 000 documents, un nombre suffisamment grand pour vérifier l'applicabilité de l'approche avec beaucoup d'individus. Il a été construit automatiquement à partir de DBpedia. Chaque document correspond à la page d'un film, et contient l'URI de la page DBpedia (la page étant déjà connue, DBpedia Spotlight ne sera donc pas utilisé) ainsi qu'un résumé du film (avec très peu d'expressions négatives). Les 12 CC choisies correspondent aux catégories de DBpedia, données par la propriété dcterms :subject, ce qui permet d'obtenir automatiquement les exemples positifs de chaque CC.

4.2 Evaluation de l'approche SAUPODOC

4.2.1 Scénario expérimental

Les exemples positifs et négatifs de chaque CC doivent être donnés en entrées pour chaque approche testée. Ils sont donnés par le concepteur de l'application dans le cas des destinations et automatiquement générés pour les films : un film f est un exemple positif pour un CC correspond à la catégorie c s'il a la propriété <f dcterms :subject c>, et un exemple négatif sinon.

SAUPODOC s'appuie sur une ontologie, ce que ne font pas les classifieurs. Nous utilisons donc la terminologie de l'ontologie comme dictionnaire du domaine. Ainsi, chaque document est modélisé par un vecteur (Vector Space Model), avec la représentation par sac de mots où

chaque élément du vecteur correspond à un mot du dictionnaire qui peut être un ou plusieurs mot-clés ou une phrase. Si, après une phase de lemmatisation, un document contient ce mot, la valeur de l'élément du vecteur correspondant est son TF-IDF, sinon la valeur est 0. Les représentations vectorielles obtenues ont été utilisées comme entrées des deux classifieurs. Ceux-ci ont été testés avec plusieurs paramètres et nous avons gardé les meilleurs résultats. Pour l'évaluation, nous avons utilisé les 2/3 des documents comme ensemble d'apprentissage et le 1/3 restant comme jeu de test. Cela signifie que l'apprentissage est fait sur 2/3 des données et que les résultats sont évalués sur les données restantes. Plusieurs métriques ont été calculées.

4.2.2 Résultats

Nous pouvons observer (cf. Tableau 1) que les trois approches donnent de bons résultats pour l'exactitude avec un léger avantage pour la nôtre. Cependant, l'exactitude n'est pas la mesure la plus appropriée dans notre problème car chaque CC a beaucoup d'exemples négatifs et peu de positifs. Or un classifieur qui ferait une prédiction systématiquement négative sur toutes les entrées aurait une exactitude élevée (91.76% en moyenne sur les CC des films). Les vrais positifs et les vrais négatifs n'ont pas la même importance dans notre problème. D'autres mesures comme la précision, le rappel et la F-mesure sont nécessaires pour évaluer la prédiction des positifs qui est centrale dans notre contexte. Le tableau 1 montre les résultats. Nous pouvons observer que notre approche est la meilleure dans tous les cas, pour ces deux domaines.

Exactitude $= \frac{VP+VN}{VP+FP+VN+FN}$ *F-mesure* $= \frac{2\times précision\times rappel}{précision+rappel}$

Précision $= \frac{VP}{VP+FP}$ *Rappel* $= \frac{VP}{VP+FN}$

Métrique (%)	Exactitude			F-mesure			Précision			Rappel		
Corpus	Nous	SVM	Arbre	Nous	SVM	Arbre	Nous	SVM	Arbre	Nous	SVM	Arbre
Destination	95.89	84.52	86.23	72.23	54.14	63.22	73.95	58.10	64.23	71.58	55.32	65.89
Film	95.46	94.41	94.32	75.65	61.74	61.40	76.27	69.90	67.72	77.76	57.59	58.99

TAB. 1 – *Résultats moyens pour les destinations (39 CC) et les films (12 CC)*

Nos résultats combinent les performances des différentes tâches effectuées par SAUPODOC. La tâche d'apprentissage de définitions permet une bonne classification, mais les tâches en amont de celle-ci ont aussi un impact sur les résultats dans le sens où elles affectent la qualité des données utilisées pour apprendre les définitions. Dans ce qui suit, nous analysons les deux tâches d'extraction sur le domaine des destinations. Une évaluation manuelle peut être faite puisque ce corpus contient peu de documents.

Nous commençons par analyser la phase d'extraction des propriétés à partir des textes. Nous avons dénombré 52 assertions de propriété fausses (faux positifs) sur 2 375 (2.19% de bruit). La précision atteint donc 97.81%. Le rappel est supposé être égal à 1. En effet, si une assertion de propriété n'est pas mentionnée dans le texte, alors cette propriété ne caractérise pas l'instance décrite puisque toutes les caractéristiques importantes sont supposées être mentionnées dans les descriptions. Ainsi, le nombre de faux négatifs (les assertions manquantes) devrait être extrêmement limité. Cela montre clairement que les résultats de la tâche d'extraction à partir de textes sont de bonne qualité. Les assertions pertinentes sont donc introduites dans l'ontologie avec un minimum de bruit.

Pour la tâche d'extraction à partir du LOD, les techniques proposées pour traiter les propriétés multiples ou multi-valuées ou manquantes ont prouvé leur utilité. Seules 29 des 80 destinations disposaient des données souhaitées sur les températures (tests effectués sur DBpedia 2014). La spécification de chemins d'exploration a permis d'obtenir des valeurs approchées : par exemple, les températures pour Boston ont été obtenues à partir de la page de Quincy,_Massachusetts.

Enfin, notons que les classifieurs connus ne fournissent pas de définitions explicites. Les classifieurs SVM créent un modèle qui n'est pas compréhensible. Les résultats des arbres de décision le sont un peu plus puisque les arbres peuvent être vus comme des ensembles de règles. Toutefois ces règles font référence aux valeurs de TF-IDF associées aux mots du dictionnaire ce qui les rend difficilement interprétables par un humain. Dans SAUPODOC, les définitions sont directement compréhensibles et peuvent être affinées si besoin.

5 Conclusion et travaux futurs

Nous avons proposé une approche originale permettant d'étiqueter automatiquement des documents décrivant des entités avec des concepts spécifiques non mentionnés dans les documents. Cette approche combine des étapes de peuplement et d'enrichissement d'ontologie. Elle fait coopérer des tâches qui travaillent à différents niveaux d'abstraction (individus, concepts) et avec des hypothèses différentes (monde ouvert, clos). Elle met en œuvre des mécanismes innovants pour exploiter le LOD sans être pénalisée par l'incomplétude de celui-ci. Les résultats montrent la pertinence d'une telle approche combinée. Une perspective, en cours d'étude, est la génération automatique des requêtes SPARQL permettant le peuplement de l'ontologie avec les informations trouvées dans le LOD, en s'appuyant sur le modèle des spécifications des correspondances complexes entre l'ontologie source et les ressources externes cibles.

Remerciements

Nous remercions la société Wepingo qui a financé ce travail dans le cadre du projet PORASO.

Références

Bontcheva, K., V. Tablan, D. Maynard, et H. Cunningham (2004). Evolving GATE to Meet New Challenges in Language Engineering. *NLE 10*(3/4), 349–373.

Chitsaz, M. (2013). Enriching Ontologies through Data. In *Doctoral Consortium co-located with (ISWC 2013), Sydney, Australia,*, pp. 1–8.

Cimiano, P. (2006). *Ontology Learning and Population from Text : Algorithms, Evaluation and Applications*. Secaucus, NJ, USA : Springer-Verlag New York, Inc.

Cimiano, P. et J. Völker (2005). Text2onto : A framework for ontology learning and data-driven change discovery. In *Proceedings of the 10th International Conference on Natural Language Processing and Information Systems*, NLDB'05, Berlin, Heidelberg, pp. 227–238. Springer-Verlag.

Cimiano, P., J. Völker, et R. Studer (2006). Ontologies on Demand ? - A Description of the State-of-the-Art, Applications, Challenges and Trends for Ontology Learning from Text. *Information, Wissenschaft und Praxis 57*(6-7), 315–320.

Cunningham, H., D. Maynard, K. Bontcheva, V. Tablan, N. Aswani, I. Roberts, G. Gorrell, A. Funk, A. Roberts, D. Damljanovic, T. Heitz, M. A. Greenwood, H. Saggion, J. Petrak, Y. Li, et W. Peters (2011). *Text Processing with GATE*.

Lehmann, J. (2009). DL-Learner : Learning Concepts in Description Logics. *Journal of Machine Learning Research 10*, 2639–2642.

Lehmann, J., S. Auer, L. Bühmann, et S. Tramp (2011). Class Expression Learning for Ontology Engineering. *Journal of Web Semantics 9*, 71 – 81.

Lehmann, J. et P. Hitzler (2010). Concept Learning in Description Logics using Refinement Operators. *Machine Learning 78*(1-2), 203–250.

Ma, Y. et F. Distel (2013a). Concept adjustment for description logics. In *K-CAP '13*, New York, NY, USA, pp. 65–72. ACM.

Ma, Y. et F. Distel (2013b). Learning Formal Definitions for Snomed CT from Text. In *Proc. of Artificial Intelligence in Medicine*. Springer-Verlag.

Mendes, P. N., M. Jakob, A. García-Silva, et C. Bizer (2011). DBpedia Spotlight : Shedding Light on the Web of Documents. I-Semantics '11, NY, USA, pp. 1–8. ACM.

Petasis, G., R. Möller, et V. Karkaletsis (2013). Boemie : Reasoning-based information extraction. In *LPNMR 2013*, A Corunna, Spain, pp. 60–75.

Tsarkov, D. et I. Horrocks (2006). FaCT++ Description Logic Reasoner : System Description. IJCAR'06, Berlin, Heidelberg, pp. 292–297.

Völker, J., P. Hitzler, et P. Cimiano (2007). Acquisition of OWL DL Axioms from Lexical Resources. In *ESWC '07*, Berlin, Heidelberg, pp. 670–685. Springer-Verlag.

Yelagina, N. et M. Panteleyev (2014). Deriving of Thematic Facts from Unstructured Texts and Background Knowledge. In *KESW*, Volume 468, pp. 208–218.

Summary

This paper proposes an approach to automatically label documents describing products, with very specific concepts reflecting specific users' needs. The peculiarity of the approach is that it confronts a triple challenge: 1) the concepts used for labeling have no direct terminology in the documents, 2) their formal definitions are not initially known, 3) all the necessary information is not necessarily mentioned in the documents. To solve this problem, we propose an annotation process in two steps, guided by an ontology. The first step is to populate the ontology with information extracted from documents, completed by others from external resources. The second one is a reasoning step on the extracted data covering either a learning phase of concept definitions, or a phase of application of learned definitions. Thus, the SAUPODOC approach is a novel approach of ontology enrichment exploiting the foundations of the Semantic Web, by combining the contributions of the LOD and text analytics, machine learning and reasoning tools. The evaluation, on two domains of application, provides quality results and demonstrates the interest of the approach.