A Combined Approach for Ontology Enrichment from Textual and Open Data

Céline Alec, Chantal Reynaud-Delaître and Brigitte Safara

Abstract This paper proposes an approach for ontology enrichment for automatically labeling documents describing entities, with very specific concepts reflecting specific users' needs. The peculiarity of this approach is that it addresses 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) the information useful to label the documents is not necessarily mentioned in them. To solve those problems, we propose to use an existing ontology of the domain of concern and to enrich it with the definitions of the concepts used for labeling. To construct these definitions, we work on a set of manually labeled documents, used as examples. The ontology is populated with information extracted from these documents, and with information coming from external resources (Linked Open Data). The definitions that we want to get can then be learned based on this populated ontology and on the set of labeled documents. Learned definitions are then added to the ontology (ontology enrichment). Hence, whenever new documents of the same domain have to be labeled, the ontology can be populated in the same way and definitions apply, allowing the new documents to be labeled. This approach, named SAUPODOC, is a novel approach to ontology population and enrichment, exploiting the foundations of the Semantic Web by combining contributions of text analysis, linked open data extraction, machine learning and reasoning tools. An evaluation, on two application domains, provides quality results and demonstrates the interest of the approach.

C. Alec (✉) · C. Reynaud-Delaître · B. Safara
LRI, Univ. Paris-Sud, CNRS, Université Paris-Saclay, 91405 Orsay, France
e-mail: celine.alec@lri.fr

C. Reynaud-Delaître
e-mail: chantal.reynaud@lri.fr

B. Safara
e-mail: brigitte.safar@lri.fr

B. Pinaud et al. (eds.), *Advances in Knowledge Discovery and Management*,
Studies in Computational Intelligence 732, https://doi.org/10.1007/978-3-319-65406-5_1

1 Introduction

This work is the result of a collaboration between LRI (Laboratoire de Recherche en Informatique) and the Wepingo[1] company, a startup which develops on-line applications for proposing products to web users. Our goal is to design an approach for automatically labeling documents describing products (or more generally entities), with very specific concepts corresponding to specific user needs. This aims at facilitating the design of flexible systems that can be adapted to different product categories and different points of view on these products. The specificity of our approach is that it is faced with three problems: (1) the documents do not contain word for word the label of specific concepts; (2) each specific concept is expressed as a label, i.e., we do not have any formal definition, but the system designer knows what type of data must be found in order to label the documents with the specific concepts; (3) documents are incomplete, they may not contain all the information necessary for this labeling.

Our problem can be exemplified by the tourism domain. Let us suppose that we have a corpus of documents, each document describing a touristic destination. We would like to find out, for each destination (so, for each document), whether the destination is a *DWW*, i.e., "Destination where you can practice Watersports during Winter", or not. This concept is very specific and never appears as such in the documents describing the destinations. However, the system designer knows that the concept refers to a place that is warm enough in winter and with watersports. He/she also knows that one can find in the documents terms referring to watersports but he/she will not find the temperature ranges in winter, and he/she will have to look for them in an external resource.

Once the necessary information is acquired, a description can then be labeled as an instance of the concept *DWW* or not. Automating the process requires clarifying the formal definition in an appropriate language, which can be difficult to achieve: for instance, what does a place warm enough in winter explicitly mean? However, a formal definition can be learned by an automatic tool if the system designer provides a number of descriptions that are already labeled as positive or negative examples of the concept. Once the definition has been learned, new descriptions of destinations satisfying this definition can be automatically labeled as instances of the concept (positive labels) or as not instances of the concept (negative labels).

Explicitly knowing the definitions gives more flexibility to the system. If no destination complies with a given definition, some constraints on the definition can be deleted in order to propose at least a few destinations. For example, a user seeking *DWW* will eventually be satisfied with a destination where the temperatures are slightly lower than what is expected.

The contribution of this paper is an original approach for ontology enrichment that combines different text analysis tools, LOD (Linked Open Data) extraction, machine learning and reasoning, in a context constrained by the three statements outlined above. The rest of the article is organized as follows. Section 2 presents the

[1] http://www.wepingo.com/.

state of the art. Section 3 describes the approach and Sect. 4 details its various tasks. Section 5 presents the experiments. We conclude in Sect. 6 on future work directions.

2 Related Work

Ontology enrichment is a vast field of research in which we distinguish three categories of works that focus on the extraction of semantic knowledge from more or less structured texts.

The first category concerns works on expressive ontologies and generation of concept definitions. Some approaches work on texts describing concepts. For example, Lexo (Völker et al. 2007) applies syntactic transformation rules on natural language definitions for generating axioms in Description Logic (DL). Ma and Distel (2013b) uses an approach based on the extraction of relations and relies on formal constraints to ensure the quality of the learned definitions (Ma and Distel 2013a). These approaches are not applicable on the documents that we deal with because they contain only descriptions of instances. Others, such as Chitsaz (2013) and Lehmann and Hitzler (2010), have only descriptions of instances, just like us. They rely on inductive logic programming to find new logical descriptions of concepts from the assertions of an ontology. Lehmann and Hitzler (2010) applies to expressive ontologies in DL, whereas Chitsaz (2013) applies to lightweight ontologies and both require a large number of assertions related to the instances. In comparison, our inputs are incomplete and weakly structured texts, from which assertions have to be extracted.

The second category of works deals with the generation of lightweight ontologies like taxonomies. They study how to extract different ontological elements from textual resources (Cimiano 2006). For the extraction of concepts, the key step is the extraction of the relevant terminology of the domain (Cimiano et al. 2006) using different measures of term weighting. Classification techniques are then applied to detect synonyms and an ontological class can be derived for each group of similar terms. Others are interested in learning concept hierarchies. They mainly use unsupervised hierarchical classification techniques to simultaneously learn the concepts and their subsumption relations (Cimiano 2006). Finally, supervised methods can be used in the case where an existing concept hierarchy has to be extended with new concepts. Classifiers must be trained for each of the existing ontology concepts which must not be very large. Appropriate similarity measures are then used to compare a new concept with the existing ones (Cimiano and Völker 2005). All these works are designed to recognize words denoting concepts (or instances) in the texts and then extract them. However, sometimes the texts only mention the properties of the instances without naming the underlying concept, as in our work. Other approaches, such that those described below, are then necessary.

The third category includes works that use reasoning to partially replace traditional extraction techniques. In the BOEMIE system (Petasis et al. 2013), concepts are divided into primitive and composite concepts, the latter ones being defined from the first ones. The primitive concepts are populated with standard extraction tools.

Instances of the composite concepts are not explicitly present in the texts but their properties are. The composite concepts are populated by reasoning on the extracted properties and on instances of primitive concepts. Yelagina and Panteleyev (2014) extracts facts from texts thanks to natural language processing tools and an ontology. From these facts, background knowledge and inference rules introduced beforehand, new facts not mentioned in the text can be derived. Our work is close to these two works but differs in the fact that we do not have definitions of the concepts to be populated.

This state-of-the-art shows that none of the approaches taken in isolation is a solution to our problem. The next section describes the approach SAUPODOC that combines some processes to fit our context.

3 The Principle of the SAUPODOC Approach

3.1 Description of the Approach

Our approach, called SAUPODOC (Semantic Annotation Using Population of Ontology and Definition Of Classes) is automatic and generic. This means that the same approach can be used for several domains, provided that all the inputs of the framework are adapted to the concerned domain. The approach aims at labeling input documents with so called target concepts. For each target concept, a document must be labeled either as an instance of this concept or as not an instance of this concept.

The SAUPODOC approach is composed of two workflows. The first one, performed once, consists in learning definitions from example documents, manually annotated. The second one consists in applying the learned definitions to new documents that need to be labeled. These two workflows rely on four tasks which are guided by a domain ontology. The first two tasks, used by both workflows, consist in populating the ontology by extracting the data associated with the described entities. This extraction is performed from the textual documents (Task 1) and external resources (Task 2). The other tasks are reasoning tasks on the populated ontology: the discovery of formal definitions of target concepts in the first workflow (Task 3) and the population of these target classes on the second workflow leading to the labeling of the documents (Task 4).

The structure of both workflows is illustrated by Figs. 1 and 2. During the whole process the ontology, initially denoted by O is progressively enriched.

The first workflow aims at constructing the definitions of target concepts. The first task populates the ontology with property assertions. These are extracted from the information contained in documents of an input corpus. Then, the second task extends these assertions using information extracted from an external resource. The populated ontology (O^+) is then enriched. The documents of the input corpus are manually labeled, in order to obtain positive and negative examples for each target concept.

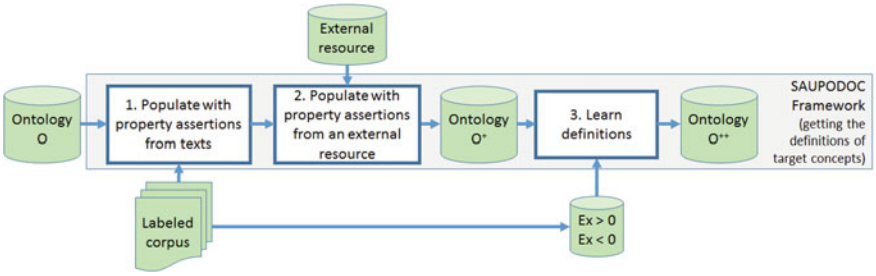

Fig. 1 Workflow 1 of SAUPODOC: getting the definitions of target concepts

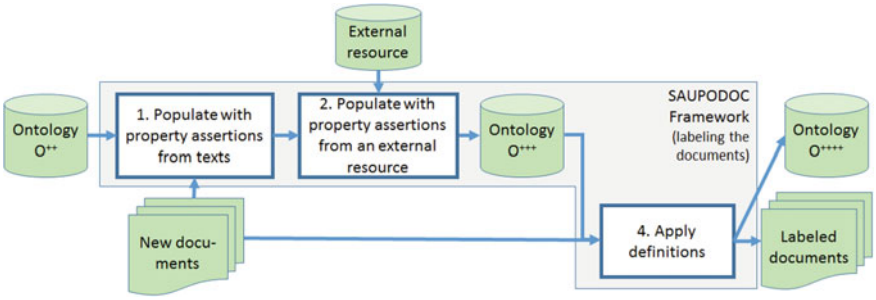

Fig. 2 Workflow 2 of SAUPODOC: labeling the documents

Definition of those target concepts are learned automatically from the examples and then inserted as classes into the ontology (O^{++}).

The O^{++} ontology can then be used for labeling new documents, cf. Figure 2. To do that, Tasks 1 and 2 have to be done, in the same way as for the labeled corpus, in order to populate the ontology with property assertions describing the entities of the new documents (O^{+++}). Finally, the definitions of the target concepts are applied, populating the classes corresponding to the target concepts (O^{++++}) and labeling the new documents with the target concepts.

The next subsections describe the inputs of the approach: (1) a domain ontology, assumed to be previously defined, enriched by the properties that the system designer believes to be involved in the future definitions of target concepts, and (2) a corpus describing entities in the form of textual descriptions, labeled as positive or negative instances of each target concept.

3.2 The Input Ontology

The input ontology defines the domain under consideration. It is a guide to the analysis of documents, the search for additional information and the reasoning on the obtained definitions and assertions. This means that the ontology has to contain

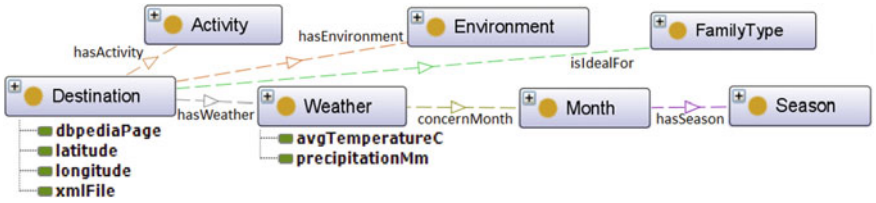

Fig. 3 Structure of the ontology about touristic destinations

all the elements defining the domain entities. It is domain-specific but not approach-specific. Its constitution is not the focus of the paper.

More formally, the ontology O is an OWL (Ontology Web Language) ontology defined as a tuple (C, P, I, A, F) where C is a set of classes, P a set of properties (datatype, object and annotation) characterizing the classes, I a set of instances and their types, A a set of axioms and F a set of facts.

C contains two types of classes:

- ***the main class***, which corresponds to the general type of the entities described in the documents. For example, Destination in Fig. 3.
- ***descriptive classes***, which are all other classes, used to define the main class. For example, Activity or Environment in the Fig. 3.

Figure 3 presents an excerpt of an ontology about touristic destinations where the main class is Destination. Descriptive classes Activity, Environment, FamilyType and Season are respectively hierarchy roots. For example, Environment represents the natural environment (Aquatic, Desert, etc.) or its qualities (Beauty, View). Some of the inter-class properties have sub-properties, which are not shown here. Datatype properties appear under the classes.

A contains constraints on the set of classes C or on the set of properties P such as subsumption, equivalence, disjunction, domain/range as well as characteristics (functional, transitive, etc.).

At first I contains only instances of descriptive classes. For instance, _rainForest is an instance of Forest (descendant of Environment) and dense forest is one of its labels.

The set of facts F is initially empty. It will contain property assertions, i.e., triples like <Dominican_Republic hasEnvironment _rainForest> that will be introduced by Tasks 1 and 2.

The set of target concepts used by the system designer to label documents will be introduced by Task 3 as a set of specializations of the main class. This set will not form a partition, as a product may be an instance of several target concepts.

3.3 The Labeled Corpus

The labeled corpus consists of a set of XML (Extensible Markup Language) unstructured documents. Every document describes a particular instance of the main class, i.e., a product, or more generally an entity. The structure reveals both the name of the instance and the textual description. The textual description contains labels of instances of descriptive classes from the ontology but no label of the target concepts, which can be complex sentences like "Destination where you can practice Watersports during Winter" (*DWW*). Each document is manually labeled by the system designer, as a positive or negative instance of each target concept. Note that in our context, the documents are either extracted from advertising catalogs (and they praise the virtues of the entities described) or very short descriptions. In all cases, few negative expressions are present and they mention all the characteristics of entities. However, some specific information may be missing like numerical values.

For the second workflow, the new documents have the same features as those of the labeled corpus (XML documents describing all the main characteristics of entities and with few negative expressions). But, they are not labeled.

4 Tasks Implemented in the SAUPODOC Approach

This section details each task involved in the approach. Note that, because it is successively enriched by the different tasks, the ontology here plays a central role, both as unifying language supporting some kind of task cooperation, as well as a guide for each of them, since these are driven by its terminology and structure.

4.1 Preliminary Task

In a preliminary task, entities described by each document of the corpus are added to the ontology as individuals of the main class. For example, if the main class of the tourist destinations ontology is Destination and if the corpus contains a document "Dominican_Republic.xml", then the ontology is populated with an individual Dominican_Republic such as <Dominican_Republic, rdf:type, Destination>.

4.2 Task 1: Population with Property Assertions from Texts

In the first task, data is extracted from textual descriptions of entities by a document annotation process according to the domain ontology. We chose to use the GATE annotation software (Bontcheva et al. 2004; Cunningham et al. 2011) because it

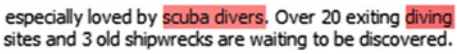

Fig. 4 Extract from the document about the Dominican Republic annotated by GATE

performs different text analysis tasks allowing the user to choose the ontology which will guide the process. Other tools like Open Calais[2] are not able to do that. Figure 4 is a snippet of the document describing the Dominican Republic destination, the terms scuba divers and diving are associated by GATE with the individual _diving from the ontology, instance of a concept specializing WaterSport.

Property assertions for each entity that are instances of the main class are then populated thanks to GATE annotations. The JAPE language (Java Annotation Patterns Engine), usable with GATE, is able to define rules transforming annotations into property assertions in the ontology. We have built a generic pattern, suitable for any ontology, which is automatically converted into as many JAPE rules as ontology properties to be populated. The system designer has to specify in the ontology what properties should be populated by the JAPE rules. For example, he or she can state that hasActivity must be considered but not hasWeather. For each property, the process is guided by the constraints on ranges expressed in the ontology. For example, the constraint <Destination, hasActivity, Activity> requires the range value taken by the property hasActivity to belong to the extension of the class Activity. From this constraint, if the description of an entity e contains an annotation corresponding to an instance a of Activity, then the assertion <e, hasActivity, a> is built. In the example of Fig. 4, as _diving is an instance of a concept specializing WaterSport (a subclass of Activity) the assertion <Dominican_Republic, hasActivity, _diving> is added.

In our context, where documents praise the advantages of the described entities and therefore contain no (or few) negative expressions, this simple population process appears to be sufficient.

4.3 Task 2: Population with Property Assertions from an External Resource

Task 2 aims at completing the property assertions found in Task 1. Indeed, the documents are often incomplete. They usually do not contain all the information needed to define a target concept. For example, intuitively the definition of a *DWW* takes into account the weather of destinations in winter. The temperature and precipitation per season or per month do not appear in the descriptions. The collection must be enriched by exploiting resources available on the Web.

To perform this task, the system designer needs an RDF resource containing entities of the domain of the corpus and to identify in this resource the properties that are required by the ontology. Currently, the resource used is DBpedia.

[2] http://www.opencalais.com/.

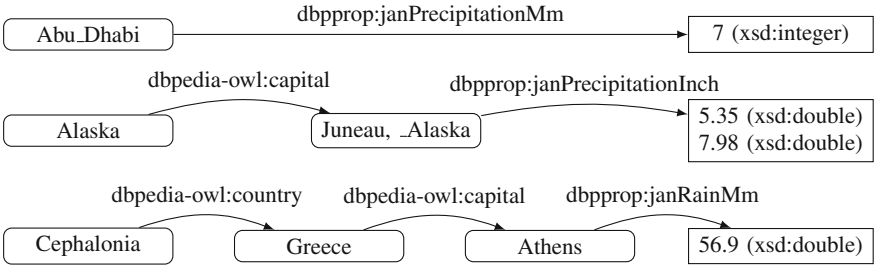

Fig. 5 Exploration paths in DBpedia

The task has two steps. First, we use DBpedia Spotlight (Mendes et al. 2011). This tool can automatically annotate the references to DBpedia entities in a text. We apply it to the name of the instance of each document of the corpus. It allows us to have a direct access to the DBpedia page representing the entity of the document. Other tools could have been used like Wikifier (Cheng and Roth 2013; Ratinov et al. 2011) or AIDA (Yosef et al. 2011), but the access would not have been direct since they return Wikipedia pages instead of DBpedia pages. The second step consists in automatically generating SPARQL queries to add property assertions to the ontology. These queries are applied via the DBpedia SPARQL endpoint.[3] A summary of this task, detailed in Alec et al. (2016) is given below.

We face a correspondence problem. Indeed, the vocabularies of the ontology and of the external resource like DBpedia may differ. Mechanisms must be conceived to establish mappings between the required ontology properties and those of the resource. Various mechanisms have been set up to deal with cases where, for example, a source property (from the ontology) corresponds to several equivalent target properties with different syntaxes. For instance, the property precipitation_in_January is represented by six different properties in DBpedia (janPrecipitationMm, janRainMm, janPrecipitationInch, janRainInch, janPrecipitationIn and janRainIn). Other mechanisms not listed here allow us to match with target properties that are calculated or obtained following an aggregation process.

We also take into account the incompleteness of DBpedia and therefore the fact that there could be no known value for some property. Indeed, in our context of definition learning from examples, the data extracted from examples should be as complete as possible, because it impacts the quality of the learned definitions. We have therefore developed a mechanism for exploring nearby pages in the resource graph in order to obtain a value that can serve as an acceptable approximation of a missing property. Figure 5 shows two examples (Alaska and Cephalonia) where the value of the property for the January precipitation is not on the destination page itself but where an approximate value can be found on a nearby page (the page of the capital city).

[3] http://dbpedia.org/sparql.

This mechanism is based on the composition of properties and allows the designer to establish alternative access paths of the DBpedia graph, until the pages containing the required information are reached. SPARQL queries (with the CONSTRUCT query form) based on these specifications collect the data and create property assertions that will be inserted into the ontology. Note that, for object properties, property assertions are not the only thing to be added. The ranges of these assertions (individuals) are added to the ontology as instances of the range of the object property under consideration. For example, let us suppose that we have an ontology about films with an object property hasLanguage such as its range is the class Language. Suppose Task 2 adds the assertion < A_Crime_in_Paradise, hasLanguage, _French_language > regarding a film named "A_Crime_in_Paradise" described in a document. In this case, an individual _French_language is created such as < _French_language, rdf:type, Language >.

4.4 Task 3: Definition Learning

Task 3 is a learning task that is performed only once, in the first workflow. First, the target concepts are inserted in the ontology as subclasses of the main class, that we call target classes. Then, definitions of each target concept are learned using the tool DL-Learner (Lehmann 2009). To learn definitions, this tool uses the ontology and positive and negative examples of the different target concepts. These definitions are added to the ontology via axioms of equivalence between a target class and its definition.

DL-Learner has been chosen for this task because it is an open source tool using an input ontology. It is able to learn class definitions expressed in Description Logic. To the best of our knowledge, two other existing tools perform the same task, YinYang (Esposito et al. 2004) and DL-FOIL (Fanizzi et al. 2008), but they are not open source. DL-Learner has many advantages in our context compared to most machine learning tools. First, it allows us to obtain the explicit definitions of each target class, which is an important asset in concrete applications. Second, most machine learning tools do not take into account the relations explained in an ontology (subsumption, object properties across classes) in their representations of the examples.

The definitions built by DL-Learner are conjunctions or disjunctions of items. An item can be a class (Destination), an expression about an object property at the class level (hasActivity some Nightlife), an expression about an object property at the instance level (hasActivity value _diving), an expression about a datatype property (avgTemperatureC some double[>= 23.0]), or a cardinality constraint (hasCulture min 3 Culture). The ranges can also be conjunctions or disjunctions of items. Hence, with a sufficient number of labeled examples given as inputs, a definition can be learned. For example, the definition of the class *DWW* can be something like this:

```
(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])))).
```

To setup DL-Learner, we have used the CELOE (Lehmann et al. 2011) (Class Expression Learning for Ontology Engineering) algorithm announced as the best for learning classes, and the default reasoner that uses the closed-world assumption (CWA). However, we have disabled the negation (NOT) and universal restriction (ONLY) operators since learned definitions should be introduced in the OWL ontology and reasoning in it is under the open-world assumption (OWA). In addition, in order to learn and operate with minimum cardinality constraints such as (hasActivity min 3 Activity), instances have automatically been expressed as disjoint (Unique Name Assumption) not to be assumed likable via an owl:sameAs. However, as the reasoning cannot work under OWA with definitions containing maximum cardinality restrictions (MAX, EXACTLY), they have been ignored, i.e., we automatically retain the best definition that does not contain such restrictions. In addition to the basic configuration, corresponding to the parameters described above, we have defined a complex configuration. A search heuristic is activated so that we can learn definitions that are really complex and far from easy to learn, such as the one expected for *DWW*.

A final important parameter of DL-Learner is the noise (called noisePercentage in DL-Learner), i.e., the percentage of positive examples that we accept to be not covered by the definitions. We proceeded by trial and error to adjust it and have established a methodology based on the conducted experiments. For every target class, 10 configurations are tested: the basic and complex configurations, each with 5 different noise values (5-15-25-35-45%). For each configuration, the highest ranked solution in terms of accuracy and size was chosen, then for each target class, the definition chosen is the best out of the 10.

At the end of this step, we have an ontology enriched by the definitions of target concepts. These definitions will be used to label new documents of the same domain.

4.5 Task 4: Definition Application

The output of Task 3 is an ontology including the definitions of target concepts. As described in Fig. 2, when we have some new documents of the same domain, we perform Tasks 1 and 2 on these documents in order to have property assertions for the entities. Then, Task 4 can be performed. It consists in applying the definitions to populate the target classes in the ontology. For this task, we chose to use FaCT++ (Tsarkov and Horrocks 2006), an available OWL-DL reasoner. We also tried to use HermiT (Shearer et al. 2008) and Pellet (Sirin et al. 2007) but they failed when tested on an ontology with lots of individuals (more than 10,000 individuals). FaCT++ applies the definitions of target classes thanks to the property assertions known about the entities. Hence, it is able to identify the entities that correspond to a

given definition. This task populates the target classes with the individuals complying with their definitions. For a given target class *tc*, if the entity described in a document *d* complies with the learned definition for *tc*, then it is recognized as an instance of *tc*. Hence, the document *d* is labeled by *tc*, otherwise it is labeled by *not tc*.

By doing this, we simulate a CWA while OWL reasons under OWA. Our particular context allows us to simulate the CWA at all steps. Hence, as we indicated in Sect. 4.4, definitions are built under CWA, from property assertions extracted from the examples. For an entity, when a property cannot be extracted, we assume it does not hold. For example, if a description of destination does not mention any beaches, we are sure that there are no beaches, since the documents are supposed to mention all the characteristics of the places. Similarly, to deal with the incompleteness of DBpedia data, we defined a path model to replace missing values by approximations in order to have all the required data.

5 Experiments

We have compared the approach SAUPODOC with two classification approaches, one based on SVM (Support Vector Machine) and the other on a decision tree. The experiments have been made on two application domains described below.

5.1 Materials

5.1.1 The Domain of Touristic Destinations

The corpus of touristic destinations contains only 80 documents. It is quite small, which makes a manual check of the found property assertions possible. Each document is automatically extracted from the catalog of the Thomas Cook website[4] and describes a particular destination (country, region, city or island). Documents are promotional, i.e., they highlight the qualities of destinations and contain very few negative expressions. Geolocation and weather data are missing. This information will be extracted from DBpedia through Task 2.

The domain ontology is well-structured. It includes a main class Destination and 161 descriptive classes. These are used to characterize the nature of the environment (46 classes), the possible activities (102 classes), the type of concerned families, e.g., with children, couples, etc. (6 classes) and classes for the weather (7 classes). Descriptive classes contain instances and their terminology forms for an easy identification in the texts. For example, the terms archaeology, archaeological, acropolis, roman villa, excavation site, mosaic are associated with the instance archaeology. 39 target concepts are under consideration.

[4] http://www.thomascook.com/.

5.1.2 The Domain of Films

The corpus of films contains 10,000 documents. It is quite large, which allows us to verify the applicability of the approach with many individuals. This corpus has been automatically built from DBpedia. Each document corresponds to a DBpedia page of a film, and contains the film name, the URI of this page (the page is already known, DBpedia Spotlight will not be used for the film study case) and a summary of the film (with very few negative expressions). The duration of the film, its languages and countries of origin will be extracted from DBpedia. Indeed, the duration of the film is not mentioned in the descriptions while the languages and countries of origin may be mentioned, but a misinterpretation is possible. For example, the word "French" in the summary may have various meanings: the film may be French (country) or in French (language), or it can tell the story of a French. We therefore prefer to use the information from DBpedia, which is clearly stated as language or country of origin, rather than the information from texts.

The ontology dealing with films is very simple, with little structure. It contains the main class Film and only the 5 descriptive classes useful for the target concepts of our experiments. If new target concepts appear, the ontology would have to be adapted w.r.t. them. Both descriptive classes Language and Country initially have no instances, since Task 2 is able not only to add assertions but in addition to instantiate descriptive classes. The 12 chosen target concepts correspond to categories of DBpedia, given by the property dcterms:subject, which allows us to automatically obtain the labeled examples of each target concept.

5.2 Experimental Scenario

The positive and negative examples of each target concept must be given as input for each tested approach. They are given by the designer of the application in the case of destinations and automatically generated for films: a film f is a positive example for a target concept corresponding to the class c when it has the property $< f$ dcterms:subject $c >$, and a negative example otherwise.

SAUPODOC is based on an ontology, in contrast to the classifiers we are comparing with (SVM and Decision tree). We use the terminology of the ontology as a domain dictionary. What we call the terminology of the ontology is the set of individuals that are instances of descriptive classes, including all of their labels. Thus, one word from this dictionary corresponds to one individual from the ontology. All of its labels are taken into consideration as if they were a unique same word.

Each document is modeled by a vector (Vector Space Model), with a bag-of-words representation where each element of the vector corresponds to a word in the dictionary (that can be one or several keywords). After a lemmatisation phase, when a document contains a word, the value of the corresponding element of the vector is the TF-IDF (Term Frequency-Inverse Document Frequency) value, otherwise the value is 0. The resulting vector representation is used as inputs of the two classifiers.

The classifiers are tested with several parameters and the best results are kept. For the evaluation, the set of annotated documents is split into two sets: the training set which contains two thirds of documents and the test set containing the remaining documents. This means that the learning is done on the two thirds of documents and the results are assessed on the remaining documents. Several metrics are calculated.

5.3 Results on the Test Set

Table 1 shows the accuracy, F-measure, precision and recall of our approach and of the two classifiers. Let us focus first on the accuracy. The three approaches work well, even if SAUPODOC is a bit better. Accuracy is important to see the overall correct labeling but it is not the only measure to be considered in our context because each target class has mainly negative examples. For example, over the film target classes, the average percentage of negative examples is 91.76%. This means that a simple classifier predicting only negatively for all inputs has a high accuracy. Other measures such as precision, recall and F-measure are thus needed to make an assessment of the positive prediction, which is important in our context. We can see in Table 1 that our approach is better than the two classifiers on these measures.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN} \qquad Precision = \frac{TP}{TP + FP}$$

$$F - measure = \frac{2 \times precision \times recall}{precision + recall} \qquad Recall = \frac{TP}{TP + FN}$$

Our results combine the performance of the different tasks performed by SAUPODOC. The learning task (Task 3) allows a good classification, but the previous tasks (Tasks 1 and 2) impact the results as well. Indeed, they play a role in the quality of the data used to learn the definitions. Since the touristic destination corpus is small, we performed a manual assessment of these two first tasks.

In Task 1, property assertions are created thanks to the information in the textual descriptions. 2,375 property assertions are created. Among them, 52 are false (false positives). This means the precision of this task is 97.81%. We do not calculate

Table 1 Average results for destinations (39 target classes) and films (12 target classes). "Us" describes the results for SAUPODOC whereas "SVM" and "Tree" respectively describe the results for the SVM classifier and the Decision tree classifier

Metric	Accuracy (%)			F-measure (%)			Precision (%)			Recall (%)		
Corpus	Us	SVM	Tree	Us	SVM	Tree	Us	SVM	Tree	Us	SVM	Tree
Dest.	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

the recall, because if a property assertion is not mentioned in the text, then this property does not characterize the instance described since all the important features are assumed to be included in the descriptions. This means our context supposes the recall to be equal to 1, since there are no false negatives (missing assertions). Task 1 does not bring much nose to the ontology.

For Task 2, the techniques proposed to address the multiple or multi-valued or missing properties have proved their utility. In our tests, performed on DBpedia 2014, only 29 of the 80 touristic destinations have the desired weather data. The specification of access paths has allowed approximate values to be found: for example, temperatures for Boston have been obtained from the page Quincy,_Massachusetts.

Moreover, our approach has another benefit over the classifiers. These ones do not provide any explicit definitions. SVM classifiers generate a model that is not understandable by a human being. Decision trees are a little more understandable since they are sets of rules, but here these rules refer to TF-IDF values, so their human interpretation is not obvious. In SAUPODOC, definitions are directly understandable by both a human and a tool. This means a refinement work can be performed as a post-process (see future works in Sect. 6).

5.4 Validation of the Approach

The labeled corpus has been divided into two thirds of documents for the training set and one third for the test set. In our approach, as well as for classifiers, a number of parameters are tested and we only keep the best results, that is, those that generate the best accuracy on the test set. Thus, a slight bias is introduced into the results as they represent the models generating the highest accuracy on the test set, and then potentially induces a drop on all measures with a new set (accuracy, precision, recall, F-measure).

Since we do the same in the three approaches (SAUPODOC, SVM, decision tree), we consider that the differences between the obtained results are sufficiently characteristic. To verify this, we have used a new labeled sample in the domain of films. This sample, called validation set, contains 10,000 new documents labeled with the 12 film target concepts. It has been obtained in the same manner as our training and test sets, i.e., using DBpedia. We only do this experiment with the film domain because a validation set is easy to create on this domain thanks to DBpedia.

On this validation set of 10,000 documents, we apply our approach with the definitions obtained in the experiments of Sect. 5.3 and we apply the classifiers obtained too. In Fig. 6, we can observe that the same trends emerge for the test set and the validation set. Therefore, the experiments of Sect. 5.3 make sense.

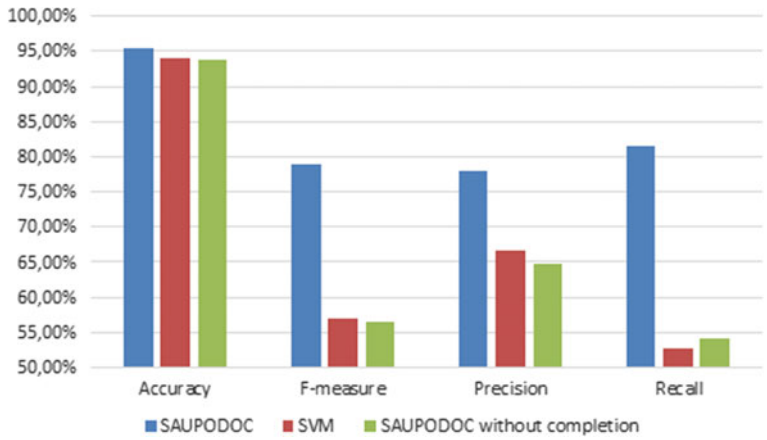

Fig. 6 Results on the validation set (10,000 films)

5.5 Experiments and Discussion on Completion

A big advantage of our approach is the exploitation of DBpedia to supplement the information of the documents. This completion corresponds to:

- geolocation and weather data for destinations,
- essentially a treatment of the possible misinterpretations in the language and country for films.

Apart from completion, the main difference between the use of data by the classifiers and SAUPODOC is the way of representing the texts:

- For classifiers, the use of a bag-of-words with TF-IDF provides a frequency notion that is not in the ontology. Indeed, in the ontology, the presence or absence of a property assertion is a binary notion, much less fine than the TF-IDF.
- For SAUPODOC, the advantage of using an ontology compared to a bag-of-words is the structure. In a bag-of-words, there is no notion of proximity between words, unlike in an ontology where similar individuals are instances of common classes, and similar classes are subclasses of common super-classes.

Figures 7 and 8 show the results on the four measures for the three approaches as well as the SAUPODOC approach without completion of the assertions with data from DBpedia (without Task 2).

For destinations, without completion of geolocation and weather data, cf. Fig. 7, SAUPODOC is less efficient, as it can be expected. However, it keeps surpassing the two classifiers on the four measures. For example, for a target concept where we intuitively think of a nice weather in winter, like *DWW*, SAUPODOC without completion cannot use weather data, since this data is obtained during completion task. Nevertheless, it is able to get a definition partly based on the environment of this type of destinations,

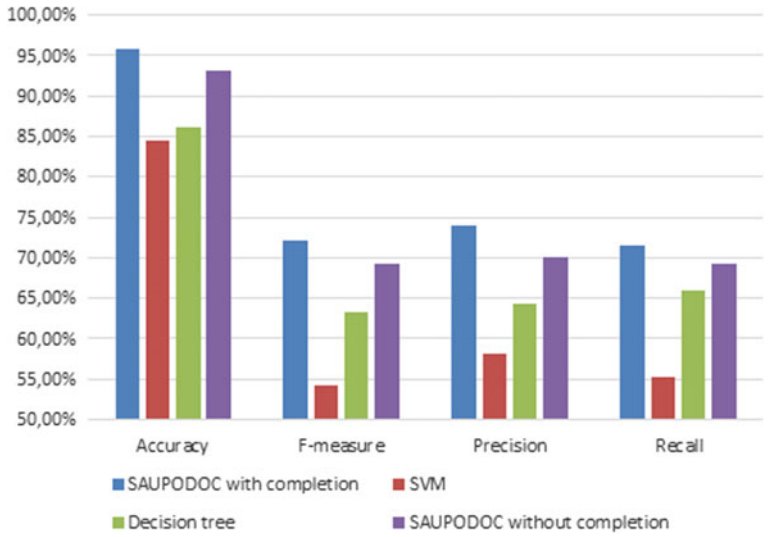

Fig. 7 Results on the corpus of destinations (carried out on the test set)

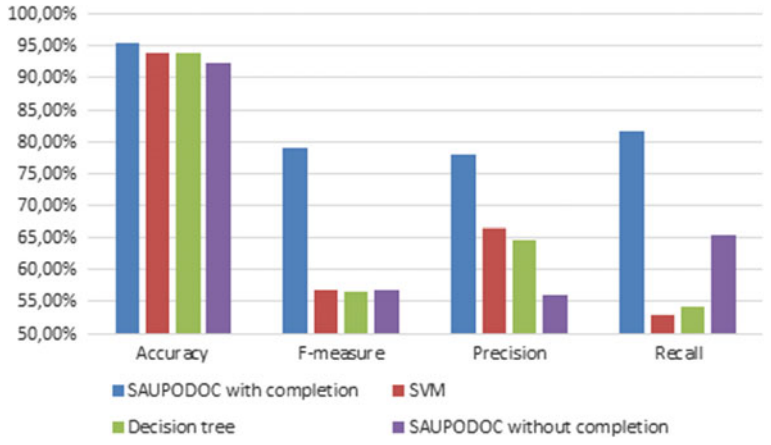

Fig. 8 Results on the corpus of films (carried out on the validation set)

e.g., (hasEnvironment some (Jungle or Vegetation)). Indeed, descriptions of destinations where the weather is good in winter often mention a jungle nearby or at least some vegetation that do not mention the other destinations. Thanks to the structure of the ontology, individuals that are instances of the class Jungle (or of the class Vegetation) are automatically seen as close individuals, unlike in the classifiers. Hence, the structure of the ontology enables to find more correct labels with SAUPODOC without Task 2 than with the classifiers in the case of touristic destinations.

Table 2 Contribution of completion in SAUPODOC. The delta is the difference between the given measure for SAUPODOC with and without completion

Corpus	Delta accuracy (%)	Delta F-measure (%)	Delta precision (%)	Delta recall (%)
Destination	2.82	2.97	3.80	2.26
Film	2.98	22.15	22.08	16.00

In the case of films, without completion of languages and countries, the four measures are considerably lower (cf. Fig. 8). Accuracy is a bit worse than for classifiers. The precision of SAUPODOC without Task 2 is clearly lower while the recall is better, creating roughly the same F-measure as classifiers. Overall, the performance of SAUPODOC without completion is very close to the one of classifiers but a little worse. This is due to the fact that the ontology is weakly structured. It connects the films to 5 concepts, but has no internal structure of concepts, and has only 18 initial individuals. Therefore, the power of ontology, residing in its ability to combine similar individuals by associating them with the same classes, is not really present here. Thus, SAUPODOC can benefit from neither the structure of the ontology, nor the notion of frequency that classifiers benefit from thanks to the bag-of-words TF-IDF.

From these experiments, we can deduce two conclusions:

- Completion has an interest for the approach. Indeed, it generates a certain contribution. For destinations, the added numerical values allow a gain between 2 and 4% in our measures, see Table 2. For movies, given the higher risk of misinterpretation in textual documents, taking into account external data allows for a large gain: around 3% for accuracy and 20% for the other measures.
- The advantage of using an ontology compared to a conventional bag-of-words resides in the exploitation of its structure. Indeed, here the bag-of-words contains notions of similarity (several labels for the same individual). It is a kind of "bag of extracted relations". However, it ignores the concepts linking the extracted relations between themselves, unlike an ontology. Thereby, the structure of the ontology allows an important contribution in the learning task. Table 3 shows

Table 3 Contribution of the structure of the ontology: case of the destinations. The delta is the difference between the given measure for SAUPODOC without completion and the classifiers

Corpus of destinations	Delta accuracy (%)	Delta F-measure (%)	Delta precision (%)	Delta precision (%)
Delta SAUPODOC without Task 2 compared to SVM	8.56	15.11	12.04	14.00
Delta SAUPODOC without Task 2 compared to decision tree	6.85	6.04	5.91	3.42
Average delta	7.70	10.58	8.98	8.71

Table 4 Absence of contribution without structure of the ontology: case of the films. The delta is the difference between the given measure for SAUPODOC without completion and the classifiers

Corpus of films	Delta accuracy (%)	Delta F-measure (%)	Delta precision (%)	Delta recall (%)
Delta SAUPODOC without Task 2 compared to SVM	−1.52	−0.02	−10.58	12.74
Delta SAUPODOC without Task 2 compared to decision tree	−1.45	0.30	−8.66	11.31
Average delta	−1.49	0.14	−9.62	12.03

that with the same information (only from documents) represented in a bag-of-words or in a well-structured ontology, using an ontology allows a gain in the labeling approach. On the contrary, without structure in the ontology, see Table 4, the ontological approach has no interest in the labeling.

6 Conclusion and Future Works

We have proposed an original approach for automatically labelling of documents describing entities with specific concepts not explicitely mentioned in documents. This approach combines steps of ontology population and enrichment. Tasks cooperate via an ontology under the closed-world assumption, which is quite original for an ontology-based approach. It implements also innovative mechanisms to exploit the LOD without being penalized by its incompleteness.

Experiments show the relevance of such a combined approach. SAUPODOC performs better than classifiers thanks to the structure of the ontology that highlights the semantic links between different individuals, and thanks to completion of information from external resources.

Future work will address experiments on new domains, such as books and music. We are also interested in a semi-automatic refinement of the definitions. Indeed, when the definition of a target concept only generates negative labels, the Wepingo company is not able to propose any entity that fit the user need corresponding to this target concept. The definition could be refined a bit in order to have some positive labels. To do this in a semi-automatic way, we could use the structure of the ontology, for example, by replacing a concept in a definition by one of its super-concept. New definitions generating positive labels would then be proposed to the designer to be validated.

Acknowledgements We acknowledge the Wepingo startup, which has funded this work in the settings of the PORASO project.

References

Alec, C., Reynaud-Delaître, C., & Safar, B. (2016). A model for linked open data acquisition and SPARQL query generation. In *Graph-based Modeling of Conceptual Structures. 22nd International Conference on Conceptual Structures, ICCS* (pp. 237–251). Annecy, France: Springer.

Bontcheva, K., Tablan, V., Maynard, D., & Cunningham, H. (2004). Evolving GATE to meet new challenges in language engineering. *Natural Language Engineering*, *10*(3/4), 349–373.

Cheng, X., & Roth, D. (2013). *Relational inference for wikification. Empirical Methods in Natural Language Processing (EMNLP)* (pp. 1787–1796), Seattle, Washington, USA.

Chitsaz, M. (2013). Enriching ontologies through data. In *Doctoral Consortium Co-located with International Semantic Web Conference (ISWC)* (pp. 1–8), Sydney, Australia.

Cimiano, P. (2006). *Ontology learning and population from text: Algorithms. Evaluation and applications*. Secaucus, NJ, USA: Springer New York Inc.

Cimiano, P., & Völker, J. (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* (pp. 227–238). Alicante, Spain: Springer.

Cimiano, P., Völker, J., & Studer, R. (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., Maynard, D., Bontcheva, K., Tablan, V., Aswani, N., Roberts, I., Gorrell, G., Funk, A., Roberts, A., Damljanovic, D., Heitz, T., Greenwood, M. A., Saggion, H., Petrak, J., Li, Y., & Peters, W. (2011). *Text Processing with GATE*. ACM Digital Library.

Esposito, F., Fanizzi, N., Iannone, L., Palmisano, I., & Semeraro, G. (2004). Knowledge-intensive induction of terminologies from metadata. In *Third International Semantic Web Conference (ISWC), Hiroshima, Japan, November 7–11* (pp. 441–455).

Fanizzi, N., d'Amato, C., & Esposito, F. (2008). DL-FOIL concept learning in description logics. *18th International Conference Inductive Logic Programming, (ILP)* (pp. 107–121). Prague, Czech Republic.

Lehmann, J. (2009). DL-Learner: Learning concepts in description logics. *Journal of Machine Learning Research*, *10*, 2639–2642.

Lehmann, J., Auer, S., Bühmann, L., & Tramp, S. (2011). Class expression learning for ontology engineering. *Journal of Web Semantics*, *9*, 71–81.

Lehmann, J., & Hitzler, P. (2010). Concept learning in description logics using refinement operators. *Machine Learning*, *78*(1–2), 203–250.

Ma, Y., & Distel, F. (2013a). Concept adjustment for description logics. *7th International Conference on Knowledge Capture, K-CAP'13* (pp. 65–72). Banff, Canada: ACM.

Ma, Y., & Distel, F. (2013b). Learning formal definitions for snomed CT from text. In *Proceedings of Artificial Intelligence in Medicine (AIME)* (pp. 73–77). Murcia, Spain: Springer.

Mendes, P. N., Jakob, M., García-Silva, A., & Bizer, C. (2011). DBpedia spotlight: Shedding light on the web of documents. *7th International Conference on Semantic Systems, I-Semantics'11* (pp. 1–8). NY, USA: ACM.

Petasis, G., Möller, R., & Karkaletsis, V. (2013). BOEMIE: Reasoning-based information extraction. *12th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR)* (pp. 60–75), A Corunna, Spain.

Ratinov, L., Roth, D., Downey, D., & Anderson, M. (2011). Local and global algorithms for disambiguation to wikipedia. In *49th Annual Meeting of the Association for Computational Linguistics (ACL)* (pp. 1375–1384).

Shearer, R., Motik, B., & Horrocks, I. (2008). HermiT: A highly-efficient OWL reasoner. In *Fifth Workshop on OWL (OWLED), Co-located with the 7th International Semantic Web Conference*, volume 432 of *CEUR Workshop Proceedings*.

Sirin, E., Parsia, B., Grau, B. C., Kalyanpur, A., & Katz, Y. (2007). Pellet: A practical OWL-DL reasoner. *Journal of Web Semantics*, *5*(2), 51–53.

Tsarkov, D., & Horrocks, I. (2006). FaCT++ description logic reasoner: System description. In *Third International Joint Conference Automated Reasoning (IJCAR)* (pp. 292–297), Seattle, WA, USA.

Völker, J., Hitzler, P., & Cimiano, P. (2007). Acquisition of OWL DL axioms from lexical resources. In *4th European Semantic Web Conference (ESWC)*, pp. 670–685. Innsbruck, Austria: Springer.

Yelagina, N., & Panteleyev, M. (2014). Deriving of thematic facts from unstructured texts and background knowledge. *5th International Conference Knowledge Engineering and the Semantic Web (KESW)* (pp. 208–218). Kazan, Russia: Springer.

Yosef, M. A., Hoffart, J., Bordino, I., Spaniol, M., & Weikum, G. (2011). AIDA: An online tool for accurate disambiguation of named entities in text and tables. In *Proceedings of the 37th International Conference on Very Large Databases, (VLDB)* (pp. 1450–1453).

Author Biographies

Céline Alec obtained her PhD in 2016 from the Université Paris-Sud, Paris-Saclay, France. Her supervisors were Pr. Chantal Reynaud-Delaître and Dr. Brigitte Safar, members of the LaHDAK team (Large-scale Heterogenous DAta and Knowledge) in the LRI (Laboratoire de Recherche en Informatique). Her research interests are situated within the domain of ontology enrichment and population applied to semantic annotation of documents.

Chantal Reynaud-Delaître is a professor of Computer Science in the Laboratory for Computer Science (LRI) at the University of Paris-Sud. For several years, she was the head of the LaHDAK (Large-scale Heterogeneous Data and Knowledge) team. Her areas of research are Ontology Engineering and Information Integration. In particular, she works on the following topics: integration of semantically heterogeneous information sources, linked open data integration and maintenance of semantic annotations impacted by the evolution of ontologies. She is involved in several projects, as the Labex DigiCosme at the University Paris-Saclay and is the author of more than 120 refereed journal articles and conference papers.

Brigitte Safara is an Assistant Professor of Computer Science in the Laboratory for Computer Science (LRI) at the University of Paris-Sud. She is a member of the LaHDAK (Large-scale Heterogeneous Data and Knowledge) team. Her research interests are Ontology Engineering and Information Integration.