An Ontology-based Method for Discovering Specific Concepts from Texts via Knowledge Completion

Céline Alec, Chantal Reynaud-Delaître, Brigitte Safar
LRI, Univ. Paris-Sud, CNRS, Université Paris-Saclay, Orsay cedex, F-91405, Email: firstname.lastname@lri.fr

***Abstract*—Heterogeneity in user's queries and data sources can easily cause problems in perceiving sufficient information to form correct answers. In this paper, we address this issue when data sources are unstructured short texts describing only key characteristics of concerned individuals but when keywords in user's queries are customized concepts. To bridge the gap between texts and user's concepts, we propose an ontology-based approach, named SAUPODOC (Semantic Annotation Using Population of Ontology and Definition of Classes), to discover formal definitions of specific concepts via population of property assertions. Property assertions are extracted from texts but the texts under our consideration are incomplete, i.e., information about the target concepts is missing. To solve this problem, we further propose a method to extract property assertions by exploiting LOD (Linked Open Data) datasets to deal with missing and multiple values. Experiments have been carried out in two application domains, whose results show a clear benefit of SAUPODOC over well-known classifiers.**

Keywords–discovering concepts from texts; ontology enrichment; ontology population.

I. INTRODUCTION

A lot of information is accessible over the Internet but discovering relevant information for users still poses today research challenges. This is mostly caused by heterogeneity in user's queries and in data sources. In this paper, we address this issue when data sources are unstructured documents describing individuals only by their key characteristics and when keywords in user's queries are customized concepts. We have to handle format heterogeneity but, most of all, content heterogeneity. One frequently used approach to semantic heterogeneity is to rely on Semantic Web techniques, particularly on ontologies. We are adopting this direction of research.

Ontologies, which are formal specifications of a domain of interest, specify the meaning of concepts in a semantically well-founded way and can be processed by machines. They are a key component to address our problem, which can be defined this way: "Given a customized concept C_c only defined as being a specific concept of C, and given a set of individuals I instances of C, find the subset of I that groups instances of C_c. Moreover, if that set is empty, find the subset of I that fulfills only partial properties of C_c". Solving this problem requires knowing definitions, i.e., the property restrictions of each customized concept and the property assertions for each considered individual. There are two reasons explaining the need for discovering precise definitions of customized concepts. The first is that it will find out individuals satisfying the definitions by exploiting the values of their properties. Secondly, definitions can be used by reasoning to provide users with partial satisfactory proposals. This point is crucial in our work. Consequently, the approach presented in the paper aims at enriching/populating a domain ontology both with formal definitions of concepts and with property assertions, which can be solved by ontology learning techniques at first glance.

However, no single approach in the ontology learning state-of-the-art addresses our problem. First, approaches dealing with generating formal definitions of concepts from texts describing generic domains [1] are not applicable because the content of our texts is not adapted. Second, property assertions in our texts are incomplete with respect to those required for defining customized concepts. Thus, extraction of property assertions from given texts only would not be enough to apply an approach at the instance level as in [2][3]. Third, the majority of ontology learning tools aiming at building a lightweight ontology extract only ontological elements [4] recognized by terms in texts. These techniques are inapplicable too. Our texts do not include names of concepts or instances. Finally, some ontology learning work deals with texts close to ours [5][6] in the sense that they involve properties of instances without naming the underlying concept. However, they assume that the precise definitions of concepts corresponding to individuals described in texts are known in advance. It is not the case in our work either.

In this paper, we investigate how several approaches can be combined in order to jointly contribute to address our problem. The proposed approach, called SAUPODOC, relies on a domain ontology relative to the field under study, which has a pivotal role, on its population by property assertions, and on automatic generation of formal concept definitions from the enhanced ontology. Our contributions are the following. We first design the SAUPODOC approach combining various tasks. Second, as property assertions extracted from texts are incomplete with respect to those required for defining customized concepts, we propose techniques to exploit LOD datasets in order to obtain further property assertions, which deal in particular with missing and multiple values. Obtaining appropriate values for all properties required to define customized concepts is indeed a critical issue solved in SAUPODOC. Finally, we experiment our approach in two application domains. We analyze the results and demonstrate the relevance of such a combined approach compared to well-known classifiers.

The remainder of the paper is organized as follows. Section II presents a motivating example of the approach. Section III exposes some related work. Section IV describes our approach. Section V presents experiments to evaluate the approach. Section VI concludes and outlines future work.

II. A MOTIVATING EXAMPLE

Let us consider a Business to Consumer (B2C) company in the holiday destination domain, accepting products from several suppliers and proposing to users the most appropriate products according to their needs. Usually, the needs are expressed from a point of view significantly different from that

Copyright (c) IARIA, 2016. ISBN: 978-1-61208-457-2

of product suppliers and change over time. These companies must design applications making searching products easy, i.e., offering totally satisfactory products or, if that is impossible, partially satisfactory products. Such applications have to be designed for a wide range of products. These application requirements have motivated our work in the context of a collaboration with the Wepingo start-up.

In this settings, textual documents are descriptions of destinations extracted from advertising catalogs or websites. They describe the main features of destinations, praise their virtues and include hardly any negative expressions. Users may be interested in destinations where they can do water sports during winter (DWW), with nightlife (DWN), or in cultural destinations (CD). DWW, DWN and CD are what we call customized concepts. They are specific in regard to the general concept Destination. There is no in-line travel catalogs, or knowledge bases indexing Dominican Republic as DWW. As an illustration, here are two excerpts of the description of Dominican Republic coming from Thomas Cook website: "[...] especially loved by scuba divers. Over 20 existing diving sites and 3 old shipwrecks are waiting to be discovered" and "[...] get active with a range of water sports". These two description parts do not reveal whether Dominican Republic is a DWW or not. One part includes the terms "scuba divers" and "diving", the other one mentions the term "water sport" but is it possible to practise water sport during winter? What is the weather in winter? There is nothing about that. Information in these texts is incomplete and can not be used to infer if the described entities are instances of DWW.

The SAUPODOC approach aims at discovering the specific concepts whose instances are described in textual documents. It requires property assertions, pa, for all considered individuals. Some pa as practised activities can be extracted from textual documents and other required pa as temperatures or precipitations about which the texts are silent will be extracted from another source. Based on the enhanced ontology, definitions of specific concepts can be then discovered. An example of definition for DWW is "a destination hot enough in winter (a mean temperature exceeding 23°C) and with little precipitation (less than 70 mm) to practise water sport activities". Dominican Republic totally satisfies that definition. More generally, all destinations satisfying the formal DWW definition will be part of the answer to a query with the keyword DWW. By contrast, Bali, which does not satisfy the constraint about precipitations, will be a partial satisfactory proposal delivered in the absence of totally satisfactory answers.

III. RELATED WORK

We distinguish three categories of work focusing mainly on semantic knowledge extraction from unstructured texts.

The first addresses learning expressive ontologies in favor of applications based on ontology reasoning. Some work in this category applies on texts containing concepts but not instances. For instance, LExO [1] applies syntactic transformation rules to generate DL axioms from definitory natural language sentences. Ma and Distel investigate a way to learn concept definitions via a relation extraction based approach [7] and propose formal constraints in [8] to ensure the quality of the definitions. It is not appropriate to apply those approaches in our work because the content of the texts is not adapted. By contrast, two research works [2][3] seek to generate a logical description from instances. They rely on the inductive logic programming technique to find a new concept description from assertions of an ontology. [3] applies to expressive DL ontologies, where [2] propose a system for light-weight DL ontologies. The main drawback of those approaches is that they require a large amount of facts about individual entities when applying to real world ontologies. Compared to [2][3], our inputs are texts, not assertions in an ontology about individuals. Needed assertions are expressed in unstructured texts but incompletely.

The second category addresses generating lightweight ontologies limited in their expressiveness, which often consists of taxonomies. Research work investigates how to extract various ontological elements that are learned from textual resources [4]. In regards to concept extraction, a main step is extracting the relevant domain terminology [9] using different term weighting measures. Clustering techniques can then be applied to detect synonyms. An ontological class can be derived from each group of similar terms. Researchers have also investigated learning concept hierarchies from texts. They mainly apply unsupervised hierarchical clustering techniques in order to learn subclass relations and concepts at the same time [4]. Approaches where patterns are identified in the texts are other applied techniques [10] although they discover lexical relations between terms, not between concepts. Finally, when the ontology has not to be built from scratch and when a concept hierarchy already exists and has to be extended with new concepts, supervised methods become possible as well. Classifiers need to be trained for each concept in the existing ontology. The number of those concepts cannot be very large. Unsupervised approaches applied in this settings use an appropriate similarity measure to compare a new concept with those already in the ontology [11]. All these research work seeks to recognise terms denoting concepts (or instances) in texts, and then extract them. However, sometimes texts involve properties of instances without naming the underlying concept, as in our work. Other approaches (3rd category) are then necessary.

The third category includes work using reasoning as a partial replacement of the traditional techniques of information extraction. In the BOEMIE system [5], concepts are divided into primitive and composite concepts. Primitive concepts are populated by classical extraction tools. Instances of composite concepts can not be found in texts but rather their properties. Consequently, composite concepts, defined in terms of primitive ones in the ontology, are populated by reasoning over primitive instances. In [6], the authors extract facts from texts thanks to an ontology and natural language processing tasks. New facts, not explicitly mentioned in texts, are then derived from extracted facts and ontology knowledge. Reasoning is based on background knowledge and inference rules given in advance. Compared to [5][6], we work on texts with a close content but the point is that we have no definitions of concepts that have to be populated.

This state-of-the-art shows that none of these approaches taken in isolation is the solution. However, some can help provided they are adapted to our requirements as outlined in the following section.

IV. THE SAUPODOC APPROACH

The tasks performed in SAUPODOC cooperate via an ontology defining the domain knowledge and populated/enriched little by little by property assertions, definitions and then individuals of customized concepts. As textual descriptions of

Copyright (c) IARIA, 2016. ISBN: 978-1-61208-457-2

entities are often incomplete in regards to required property assertions, collected data has to be complemented. We propose to exploit LOD datasets. The knowledge engineer has in charge to choose which properties can be populated from texts and those which can be populated from other sources. Definitions of customized concepts are then derived based on the populated ontology and applied to obtain their individuals. We present the general approach and each of the tasks that it comprises.

A. An ontology-based approach

The ontology, an input of SAUPODOC, contains all elements defining entities in the application field. It is domain specific but approach independent. Indeed, the only constraints imposed by the approach are described in this subsection. The ontology can therefore be largely reused, or (semi-) automatically built, its creation is not the focus of the paper. More formally, the ontology $\mathcal{O}$ is an OWL ontology defined as a tuple ($\mathcal{C}$, $\mathcal{P}$, $\mathcal{I}$, $\mathcal{A}$) where $\mathcal{C}$ is a set of classes, $\mathcal{P}$ a set of (datatype, object and annotation) properties characterizing the classes, $\mathcal{I}$ a set of individuals and property assertions, and $\mathcal{A}$ a set of axioms including constraints on classes and properties: subsumption, equivalence, type, domain/range, characteristics (functional, transitive, etc.), disjunction. Figure 1 shows an excerpt of an ontology in the holiday destination domain. The classes Activity, Environment, FamilyType and Season are respectively the roots of a hierarchy, e.g., Environment expresses the natural environment (Aquatic, Desert, etc.) or its quality (Beauty, View). Some object properties represented on the figure have subproperties. Datatype properties are represented under their domain class. Individuals are not represented on the figure.

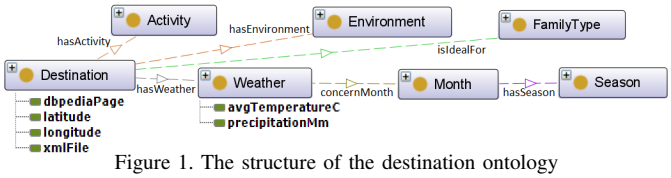

Figure 1. The structure of the destination ontology

Our approach is based on the fact that the names of customized concepts are known by the knowledge engineer but he has no precise definitions. The only thing that he knows is that they are more specific than more general ones including numerous individuals. What we are proposing in our approach is to introduce that knowledge in the ontology. Classes in the ontology are then distinguished as follows:

- ***the main class*** (MC) corresponds to the general type of entity considered.
- ***the target classes*** (TC) represent customized concepts. They have a name but no definition.
- ***the descriptive classes*** (DC) are all the other classes included in the ontology.

We add a subclass relationship between each TC and MC. $\mathcal{I}$ initially contains individuals being instances of DC, and annotation property assertions.

The SAUPODOC approach will gradually complete the ontology as follows (cf. activity diagram Figure 2):

- by individuals of MC representing the entities in the corpus, an individual being created for each considered document.
- by property assertions: either properties of individuals extracted from texts or from other (possibly several) sources;
- by definitions of TC specifying their specificities in regard to all the properties of MC, this step being only necessary the first time documents of a given domain are addressed;
- by individuals of TC.

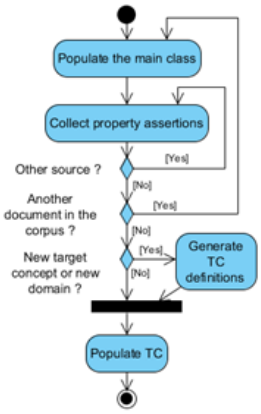

Figure 2. Activity diagram

B. Extraction of property assertions from texts

This step consists in annotating texts given $\mathcal{O}$ where all concerned properties are represented and then in representing those annotations in $\mathcal{O}$ as property assertions. We assume that texts do not include negative expressions that could disrupt the process. Thus, a simple information extraction system is appropriate provided the system can take $\mathcal{O}$ as input.

A property assertion states that an individual is connected by a property to another individual or a literal. Property assertions added in $\mathcal{O}$ have to follow this format. The extraction step is guided by $\mathcal{O}$, which includes terms corresponding to labels of individuals and allows extraction of property assertions related to individuals. For instance, property assertions connected to "snorkeling" can be added because "snorkeling" is an individual in $\mathcal{O}$, associated with labels that are used in texts to refer to it. The introduction of property assertions into $\mathcal{O}$ is guided by object properties and their associated range constraints expressed in $\mathcal{O}$. For example, based on the statement <Destination, hasActivity, Activity>, which asserts that the range values of the property hasActivity must belong to the extension of the class Activity, if the description of an entity e contains a match with an individual a instance of Activity, then <e, hasActivity, a> is introduced in $\mathcal{O}$.

In our work, we use GATE [12], an open source software performing a lot of text processing tasks. The GATE resource OntoRoot Gazetteer can produce annotations over textual documents w.r.t. an ontology given as input, in combination with other generic GATE resources. The JAPE transducer applies JAPE rules in order to transform annotations to property assertions. Rules are automatically created from one pattern, a rule per object property having to be populated. The same pattern is used whatever the ontology.

C. Extraction of property assertions from the LOD

Exploiting datasets coming from the LOD is interesting because number of those datasets can easily be queried thanks to SPARQL endpoints. Our goal is to build Construct SPARQL queries to get data from the LOD and add property assertions in our ontology. At first, the knowledge engineer has to find the most relevant datasets including information relative to the entities in the corpus. Then he has to recognize within those datasets data that corresponds to what is required. As the

Copyright (c) IARIA, 2016. ISBN: 978-1-61208-457-2

number of concerned properties is not too large, this task can be done manually. However, vocabularies differ in $\mathcal{O}$ and in the LOD datasets and correspondences may be very complex. We analyze this mapping process by giving first a few examples.

The first example illustrates a *1:n* correspondence. A property $prop$ characterizing c in $\mathcal{O}$ may be associated to several equivalent properties but with a different syntax and having possibly a value expressed with a different unit of measurement. For example, the property "precipitationMm" for January in a given place in $\mathcal{O}$ is represented in DBpedia amongst others by dbp:janPrecipitationMm, dbp:janRainMm, dbp:janRainInch. The second example is also a *1:n* correspondence but the properties are not equivalent to each other and are all needed for calculating the value of $prop$. For example, the temperature for any month in a given location can be obtained in DBpedia with the average between the highest and the lowest temperatures during that month. The third example is about missing values. While DBpedia is a huge knowledge base, it is also incomplete. A lot of property assertions are missing. For example, a given destination may have no value for the highest temperature during a given month.

We have to consider all these various situations including multiple properties, missing values together with multivalued properties, i.e., one property with possibly different values. Although incompleteness is not a problem in Linked Data under assumptions of many research works, having complete information is essential in our case in order to achieve the best results as output of the entire process. Consequently, we define a model to specify all the possible correspondences between a property $prop$ characterizing c in $\mathcal{O}$ and related properties in a RDF dataset. We define also a model to specify alternative access paths to properties in case of missing values.

1) Specification of correspondences between properties: The correspondences are the result of a matching process between a source ontology $\mathcal{O}_s$ and a target ontology $\mathcal{O}_t$, more precisely between an OWL ontology and another one providing access to a RDF data source. The correspondences are relationships between Property Expressions (PE) expressed as triples (id, PE_s, PE_t) where id is the correspondence identifier, PE_s is a property expression in $\mathcal{O}_s$ and PE_t is a property expression in $\mathcal{O}_t$.

A property expression in $\mathcal{O}_s$ (PE_s) is either an object property (op) or a datatype property (dp) possibly including restriction constraints on its domain ($PE_s.Constr(d)$) using any ontological constraint from the time it can be expressed in OWL DL, cf definition 1.

Definition 1. $PE_s = op \mid dp \mid PE_s.Constr(d)$

Example The datatype property "precipitationMm" with its domain "Weather" constrained by <Weather concernMonth January> is a PE_s that matches the property dbp:janPrecipitationMm in O_t, i.e., precipitationMm.<Weather concernMonth January> $\in PE_s$.

A property expression in $\mathcal{O}_t$ (PE_t cf definition 3) is either an elementary property (p_e) in $\mathcal{O}_t$, a mathematical, set-theoretic, transformation or agregation expression (f) using property expressions in $\mathcal{O}_t$. A PE_t can include domain or range typing constraints ($PE_t.Constr$). An elementary property p_e (cf definition 2) is either a property in $\mathcal{O}_t$ or its inverse.

Definition 2. $p_e = op \mid dp \mid op^{-1}$

Definition 3. $PE_t = p_e \mid f(PE_t) \mid f(PE_t, PE_t) \mid PE_t.Constr$

Example AVG(UNION(dbp:janPrecipitationMm, dbp:janRainMm)) is a PE_t whose value is the average of all the values of dbp:janPrecipitationMm and dbp:janRainMm.

The knowledge engineer defines correspondences between properties in $\mathcal{O}_s$ and $\mathcal{O}_t$ based on this model and on knowledge in $\mathcal{O}_s$. That way, if PE_s in $\mathcal{O}_s$ is related to a functional property and its correspondence PE_t in $\mathcal{O}_t$ is multivalued, an aggregation function is applied to obtain a single value.

2) Specification of access paths: The correspondence model requires access to p_e values in the target dataset but some of them may be missing. For instance, no precipitation data exists in the Kefalonia page. To solve this problem, we define the notion of *i^{th}-Order property* (cf definition 4).

Definition 4. *An i^{th}-Order property ($p^i.p^{i-1}...p^1$) is a property that can be reached from the initial page through a path of length i in the data graph.*

Example "janPrecipitationMm", a property of Abu Dhabi with the value 7, is a 1st-Order property (p^1) w.r.t. Abu Dhabi.

Example "country.capital.janRainMm" with "janRainMm" a property of Athens with the value 56.9, Athens being the capital of Greece, which is the country of Kefalonia, is a 3rd-Order property ($p^3.p^2.p^1$) with respect to Kefalonia.

Based on this notion, we define two kinds of access paths for p_e w.r.t. a PE_t: *direct* if p_e is accessed by a 1st-Order property, *combined* if p_e is accessed by a nth-Order property (n>1).

Combined access paths are alternatives to access to property values. They allow to obtain approximate values, which are more or less good values obtained by composing properties. The knowledge engineer has to define all access paths (a maximum) for each p_e involved in correspondences with properties in $\mathcal{O}_s$. In case of multiple paths of a given order, they must be ordered w.r.t. their relevance. Parts of combined access paths independent of p^1 can be reused.

In our work, we chose to work with DBpedia. We applied DBpedia Spotlight [13] on each document of the corpus to have an access to the DBpedia page corresponding to the described entity. Then, the model of correspondences and the specification of access paths are used as a support to write SPARQL queries in order to access DBpedia.

D. Deriving definitions of target concepts

Discovering *target definitions* of TC is a reasoning task based on the populated ontology, i.e., on all property assertions of individuals of MC. The knowledge engineer may not be able to express precise definitions of TC but we assume that he is able to manually annotate a subset of documents describing instances as positive and negative examples of each class from TC. Manual annotations of entity descriptions can therefore be used by Machine Learning (ML) tools as positive and negative examples for each TC, in order to induce their definition. This will allow us to get an explicit formal definition for all TC. These definitions are then applied to get instances of TC.

We need ML tools capable of learning definitions of classes expressed in Description Logics from expert-provided examples. Specifically, definitions of TC must be learned from (i) a populated OWL ontology including all the property assertions of all individuals of MC and from (ii) positive and negative examples. Moreover, explicit specifications of relations (subsumption, object/datatype properties) between

features as it is expressed in an OWL ontology have to be taken into account. For example, it could be able to learn what a DWW is, given which destinations from the corpus are one (e.g., Dominican Republic) and which are not (e.g., Alaska).

We chose to use DL-Learner [14], an open source tool using inductive logic programming on Description Logics. The DL-Learner definitions are conjunctions and disjunctions of elements. An element can be a class (Destination) or an expression using object properties (hasActivity some Nightlife), numerical datatype properties (avgTemperatureC some double[>= 23.0]), or cardinality constraints (hasCulture min 3 Culture). Ranges are conjunctions and disjunctions of elements. For example, (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])))) is a definition for DWW that can be learned by DL-Learner.

We developed a methodology from the conducted experiments. For each TC, 10 configurations are tested, each one with a different set of parameters, tuned using test experiments. For each test, we keep the highest ranked solution, which is the best one in terms of Accuracy and length. For each TC, we then choose the best definition from the 10 tests.

E. Ontology enrichment with individuals of target classes

The discovered definitions are applied to retrieve all individuals that instantiate TC. This task has to be performed any time new entity descriptions are provided. We use FaCT++ [15], an efficient OWL-DL reasoner applicable to a large number of individuals unlike HermiT [16] and Pellet [17], according to our experiments. If there is no instances for a TC, some restrictions in the definition can be relaxed so that we obtain partial satisfactory proposals.

In conclusion to this section, it may be noted that adjustments simulating the Closed-World Assumption (CWA) were made for ensuring all tasks cooperate with each other. As the open world assumption is made for OWL ontologies, we disabled negation (NOT) and universal restrictions (ONLY) in the definition learning task. We adopted the Unique Name Assumption (UNA) in the ontology specifying that all individuals are different from each other. Otherwise, they are supposed to be possibly connected by owl:sameAs links and this can lead to problems when reasoning with definitions containing minimum cardinality restrictions generated by learning methods under a CWA. Furthermore, as inferences cannot be made on OWL with a definition having a maximum cardinality restriction, this type of restriction is ignored, i.e., we keep the highest ranked definition such that it does not contain a maximum cardinality restriction. The extraction tasks have also been adjusted. If one property assertion is not extracted, we presume that it is unlikely. For example, if a document about a destination does not mention beaches, it is assumed that there are no beaches, as documents are supposed to describe all the assets of destinations. Conversely, if properties are relevant for some types of individuals, values must be found for each of them. This explains why techniques presented in Section IV-C complete the missing property assertions in RDF data sources.

V. EXPERIMENTAL EVALUATION

We compare SAUPODOC with classification approaches, which are tools to discover concepts from texts even if they do not generate any definitions.

A. Materials

We experiment our approach in two application domains chosen for their different characteristics.

1) The holiday destinations field: The corpus of holiday destinations is small (80 documents), which makes it possible to manually verify collected assertions. Each document has been automatically extracted from the Thomas Cook catalog (http://www.thomascook.com/) and describes a specific place (country, region, island or city). The documents are promotional, i.e., describe the qualities of destinations on a comprehensive basis and have hardly any negative expressions. The ontology includes one main class, Destination. Descriptive classes (161) characterize the nature of the environment (46 classes), the activities that can be done (102 classes), the kind of family that should go there, e.g., people with kids, couples, etc. (6 classes) and classes to define the weather like the seasons (7 classes). These descriptive classes contain individuals associated with terminological forms for facilitating their identification in texts. For example, the terms "archaeology, archaeological, acropolis, roman villa, excavation site, mosaic" are associated to the individual archaeology. 39 TC are addressed.

2) The film field: The film corpus contains 10,000 documents, a significant number in order to check the applicability of the learning step with many individuals. It has been automatically built using DBpedia. Each document corresponds to a DBpedia page about a film. A document contains the DBpedia URI (no need to use DBpedia Spotlight to get the page) and its abstract describing the film (with hardly any negative expressions). The film ontology is basic (5 descriptive classes). It only contains the needed classes w.r.t. TC of our experiments. 12 TC corresponding to DBpedia categories (values of the property dcterms:subject) are addressed.

B. Evaluation of the SAUPODOC approach

1) Experimental scenario: Positive and negative examples for each TC have to be given as input for all tested approaches. They are manually given by the knowledge engineer for destinations and automatically generated for films: a film f is a positive example for a TC corresponding to a category c if <f dcterms:subject c>, otherwise it is a negative example.

SAUPODOC relies on an ontology but classifiers do not. The idea is then to considerate the domain terminology given by the knowledge engineer as a domain dictionary. Each document is represented as a vector (Vector Space Model). We use a bag-of-words method, where each element of the vector represents a word in the dictionary, which can be one or several keywords or keyphrases. Basically, if a document contains a word (lemmatization is performed), the value for its element is TF-IDF, otherwise 0. These vector representations are used as input of (i) a SVM classifier and (ii) a decision tree classifier. Both classifiers are tested with several parameters. We keep the best results.

For evaluation, documents are split into 2/3 for the training set and 1/3 for the test set. This means that learning is performed on 2/3 of data and that results are given on the rest of data. Several metrics are computed.

2) Results: First, we observe (see Table I) that the three approaches give satisfactory results in terms of Accuracy although slightly better results are obtained with ours. However, Accuracy is not the measure the best appropriate to our

Copyright (c) IARIA, 2016. ISBN: 978-1-61208-457-2

problem because each TC has lots of negative examples and few positive ones: if a classifier predicts negative on all inputs (no instances of TC found) then Accuracy is high (91.76% on average for film TC). True negatives and true positives are not of equal importance. Alternative metrics such as Precision, Recall and F-measure are needed to evaluate the prediction of positives, which is central in our problem. Table I shows the results with respect to those metrics. We can observe that our approach is the best in terms of Precision, Recall and F-measure on both domains.

$$\textit{Acc.} = \frac{TP+TN}{TP+FP+TN+FN} \qquad \textit{F-measure} = \frac{2\times Precision\times Recall}{Precision+Recall}$$

$$\textit{Precision} = \frac{TP}{TP+FP} \qquad \textit{Recall} = \frac{TP}{TP+FN}$$

TABLE I. Average results for TC

Metric (%)	Accuracy			F-measure		
Corpus	Us	SVM	Tree	Us	SVM	Tree
Destination (39 TC)	95.89	84.52	86.23	72.23	54.14	63.22
Film (12 TC)	95.46	94.41	94.32	75.65	61.74	61.40
Metric (%)	Precision			Recall		
Corpus	Us	SVM	Tree	Us	SVM	Tree
Destination (39 TC)	73.95	58.10	64.23	71.58	55.32	65.89
Film (12 TC)	76.27	69.90	67.72	77.76	57.59	58.99

These results evaluate the combined performance of the various tasks performed by SAUPODOC. The definition learning task allows a good classification but the tasks upstream of the learning process have an impact on the results too in the sense that they affect the quality of data used to learn definitions. In the following, we analyze the two extraction tasks on the destination domain. The corpus contains few documents. A manual validation can be conducted.

We analyze first the extraction of property assertions from texts. We notice 52 wrong property assertions (false positives) out of 2,375 (2.19% of noise). Precision reaches 97.81%. Recall is assumed to be close to 1. Indeed, if a property assertion is not mentioned in a text, then that property does not characterize the described individual since all main features are supposed to be given in textual descriptions. This way, the number of false negative assertions (missing assertions) should be extremely limited. This clearly shows that the quality of the extraction task from texts is good. Relevant assertions are introduced in the ontology with minimal noise.

With respect to the extraction task from the LOD, the proposed techniques dealing with multiple or multivaluated properties and missing values proved to be very useful. Only 29 from 80 destinations have weather data (tested on DBpedia 2014). Specification of access paths provided approximated values. For example, the weather for Boston has been given from the page Quincy_Massachusetts. Moreover, complex correspondences have been defined for all properties (26) having to be valued from DBpedia. Property expressions in DBpedia corresponding to properties in the ontology were quite complex, never elementary properties.

Finally, let us note that well-known classifiers do not result in explicit definitions. SVM classifiers create a model, which is not comprehensive for human. Decision tree classifiers are a bit more intelligible since trees can be seen as sets of rules. However, these rules deal with the TF-IDF number associated with a dictionary word, which is hard to interpret by humans. In SAUPODOC, definitions are comprehensive and could be refined if needed.

VI. CONCLUSION AND FUTURE WORK

We proposed an ontology-based approach, SAUPODOC, to solve an issue of heterogeneity between customized concepts without a priori definitions and unstructured documents describing individuals in an incomplete way. The approach combines several tasks operating at different abstraction levels and exploits the LOD. We also proposed a model to specify complex correspondences between an ontology and LOD data sources and mechanisms to deal with incompleteness and multiple properties or values. Finally, experiments have been carried out. Results show the relevance of SAUPODOC and a better Precision, Recall and F-measure than well-known classifiers. Future work will address automatic generation of SPARQL construct queries to query the LOD based on the two models described in Section IV-C.

ACKNOWLEDGEMENTS

This work has been funded by the PORASO project, in the setting of a collaboration with the Wepingo company.

REFERENCES

[1] J. Völker, P. Hitzler, and P. Cimiano, "Acquisition of OWL DL Axioms from Lexical Resources," in ESWC. Innsbruck, Austria: Springer, 2007, pp. 670–685.

[2] M. Chitsaz, "Enriching Ontologies through Data," in Doctoral Consortium co-located with ISWC, Sydney, Australia. CEUR-WS.org, 2013, pp. 1–8.

[3] J. Lehmann and P. Hitzler, "Concept learning in description logics using refinement operators," Machine Learning, vol. 78, no. 1-2, 2010, pp. 203–250.

[4] P. Cimiano, Ontology learning and population from text - algorithms, evaluation and applications. Springer, 2006.

[5] G. Petasis, R. Möller, and V. Karkaletsis, "BOEMIE: Reasoning-based Information Extraction," in LPNMR. A Corunna, Spain: CEUR-WS.org, 2013, pp. 60–75.

[6] N. Yelagina and M. Panteleyev, "Deriving of Thematic Facts from Unstructured Texts and Background Knowledge," in KESW, vol. 468. Springer, 2014, pp. 208–218.

[7] Y. Ma and F. Distel, "Learning Formal Definitions for Snomed CT from Text." in Proc. of Artificial Intelligence in Medicine. Springer Berlin Heidelberg, 2013, pp. 73–77.

[8] ——, "Concept Adjustment for Description Logics," in K-CAP. New York, NY, USA: ACM, 2013, pp. 65–72.

[9] P. Cimiano, J. Völker, and R. Studer, "Ontologies on Demand? - A Description of the State-of-the-Art, Applications, Challenges and Trends for Ontology Learning from Text," Information, Wissenschaft und Praxis, vol. 57, no. 6-7, Oct. 2006, pp. 315–320.

[10] P. Cimiano, S. Handschuh, and S. Staab, "Towards the Self-annotating Web," in WWW. New York, NY, USA: ACM, 2004, pp. 462–471.

[11] P. Cimiano and J. Völker, "Text2Onto - A Framework for Ontology Learning and Data-driven Change Discovery," in NLDB, vol. 3513. Alicante, Spain: Springer, 2005, pp. 227–238.

[12] H. Cunningham et al., Text Processing with GATE. University of Sheffield Department of Computer Science, 2011.

[13] P. N. Mendes, M. Jakob, A. García-Silva, and C. Bizer, "DBpedia Spotlight: Shedding Light on the Web of Documents," in I-Semantics. NY, USA: ACM, 2011, pp. 1–8.

[14] J. Lehmann, "DL-Learner: Learning Concepts in Description Logics," Journal of Machine Learning Research, vol. 10, 2009, pp. 2639–2642.

[15] D. Tsarkov and I. Horrocks, "FaCT++ Description Logic Reasoner: System Description," in IJCAR. Berlin, Heidelberg: Springer, 2006, pp. 292–297.

[16] R. Shearer, B. Motik, and I. Horrocks, "HermiT: A Highly-Efficient OWL Reasoner." in OWLED, vol. 432. CEUR-WS.org, 2008.

[17] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A practical OWL-DL reasoner." Journal of Web Semantics, vol. 5, no. 2, 2007, pp. 51–53.

Copyright (c) IARIA, 2016. ISBN: 978-1-61208-457-2