A Multiple-Domain Ontology Builder

Sara Salem **Samir AbdelRahman**

Computer Science Department
Faculty of Computers and Information - Cairo University
{s.salem,s.abdelrahman@fci-cu.edu.eg}

Abstract

The interpretation of a multiple-domain text corpus as a single ontology leads to misconceptions. This is because some concepts may be syntactically equal; though, they are semantically lopsided in different domains. Also, the occurrences of a domain concept in a large multiple-domain corpus may not gauge correctly the concept significance. This paper tackles the mentioned problems and proposes a novel ontology builder to extract separate domain specific ontologies from such a corpus**.** The builder contribution is to sustain each domain specific concepts and relations to get precise answers for user questions. We extend a single ontology builder named Text2Onto to apply our thought. We fruitfully enhance it to answer, more precisely, questions on a subset of AQUAINT corpus.

1 Introduction

Domain ontology is a knowledge representation of the domain as a set of concepts and relations. Ontology notion always presents handy semantic solutions for various hot research areas such as Semantic Web, Informational Retrieval, and Question Answering.

Currently, automatic ontology builders presume that the given corpus has a single domain. When used with a multiple-domain corpus, these builders generate 1 large ontology for the whole corpus. Dramatically, this causes 2 domain misconception problems. First, the ontology conceptual model becomes imprecise for the common concepts in various domains having different semantics. Second, the relevance weights assigned to the concepts do not measure precisely their significance in specific domains.

This paper presents a promising solution for the mentioned problems. The proposed solution is an integrated 2-layer ontology builder. The ontology layers are: 1) the conceptual layer, which has the key concepts and relations of each separate domain, and 2) the general layer, which maintains the general domain information regarding related persons, organizations, locations, and dates. Our proposed 2-layer ontology improves the extracted answers for single-domain and cross-domain questions. We successfully prove our thought against Text2Onto builder.

Ontology extraction from a domain corpus has been targeted by many researchers. The core extraction approaches can be classified into 3 approaches. The first approach is to build the ontology from scratch (Buitelaar et al., 2004; Cimiano and Völker, 2005). The second approach is to extend a predefined general ontology, such as WordNet, with possible application domain concepts and relations (Navigli and Velardi, 2004). The last approach is to build ontology as a composition of other predefined ontologies (Cimiano et al., 2006). Moreover, as an ontology building design decision, the resultant ontology is either a single layer ontology or a multi-layered ontology (Benslimane et al., 2000; Dumontier and Villanueva-Rosales, 2007).

The paper is organized as follows: Section 2 introduces some related systems; Section 3 explains the misconceptions due to extracting a single ontology from a multiple-domain corpus; Section 4 describes our proposed builder; Section 5 illustrates our Question Answering system, which is used for the evaluation; Section 6 states our evaluation results; and Section 7 draws our conclusion and directions for the future work.

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 967–975,
Beijing, August 2010

2 Related Work

There are 3 main approaches for ontology building, namely building from scratch, extending a general ontology, or building an ontology as a composition of other predefined ontologies.

Text2Onto (Cimiano and Völker, 2005) applies the first approach. It is a framework for learning ontologies automatically from textual data. It implements diverse linguistic and statistical techniques to extract domain concepts and relations. It combines results from different techniques, and it represents the extracted ontology elements in a so called Probabilistic Ontology Model (POM), which assigns a confidence value for each learnt element.

OntoLT (Buitelaar et al., 2004) is another example of building from scratch. It is a Protégé[1] plug-in that extracts ontology from text by defining a set of mapping rules. The rules map certain linguistic structures in an annotated text into ontological elements. The extracted elements are validated by the user before being inserted into the ontology.

OntoLearn (Navigli and Velardi, 2004) employs the second approach. It is a framework for trimming and extending general purpose ontologies, like WordNet, with specific domain terminologies and taxonomies. It extracts domain terminologies, and it uses a relevance measure to keep out non-relevant terms. OntoLearn uses a novel technique, called SSI, to assign a domain specific term to the correct sense in a general ontology.

The third approach is proposed in (Cimiano et al., 2006). It presents a system that integrates several heterogeneous semantic sources into 1 ontology, which is used to extract answers for user queries from various knowledge sources.

As a design decision, the ontology may consist of a single layer or of multiple layers. Benslimane et al. (2000) apply the multiple-layer approach for manually generating a set of interrelated ontology layers; each layer models a spatial domain specific function. Also, Dumontier and Villanueva-Rosales (2007) suggest a 3-layer ontology design. The first layer (primitive layer) defines the basic domain concepts and relations. The second layer (complex layer) imposes more complex domain restrictions on the primitive layer. The top layer (application layer) maintains application specific restrictions.

Our builder constructs a layered ontology from scratch. Its main distinguished features are: 1) generating separate domain specific ontologies from a multiple-domain corpus, 2) extracting general domain information, in addition to core domain conceptual information, and 3) it is an automatic multi-layered ontology builder, unlike other automatic builders, which generate single layer ontologies.

Our system can extend current builders, which extract ontologies from textual data, allowing them to handle a multiple-domain corpus. We selected Text2Onto because it is an automatic ontology builder, and it implements a variety of algorithms to extract many types of ontology elements. We use a news corpus as a multiple-domain corpus since it contains documents from different domains like Politics, Sports, Arts, and Finance.

3 Ontology Misconceptions

Building a single ontology for a given corpus is a familiar method. However, when dealing with a multiple-domain corpus, the builder usually suffers from the following 2 problems:

First, the ontology conceptual model becomes imprecise in the definition of common concepts that are semantically lopsided in different domains. For example, the concept "wall street" in the Finance domain is defined as a financial institution, and it is in the Arts domain defined as a movie. It is inaccurate to define the concept with 2 totally different meanings in 1 ontology. It is also incorrect to ignore a definition of them. When using Text2Onto for that example, it generates only 1 definition for "wall street" as a subclass-of "institution".

Second, when weighing concepts in a multiple-domain corpus, the relevance weights assigned to the concepts do not indicate the significance of each concept in a certain domain. As a result, some core domain specific concepts may have low weights with respect to the whole corpus. For example the concept "trading" has a low weight in a multiple-domain corpus; although, it is a main concept in the Finance domain (Section 6.2). This gives wrong indication of the concept importance to the user.

[1] http://protege.stanford.edu/

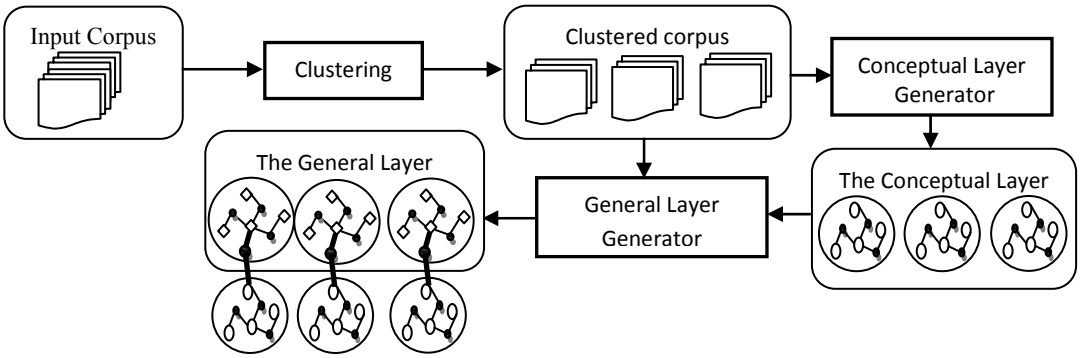

Figure 1. The Multiple-Domain Ontology Builder.

4 The Proposed Ontology Builder

Our builder aims to extract precise ontologies, which model possible knowledge in a multiple-domain corpus. A domain corpus, mostly, not only contains information about the core domain concepts and their relations, but it also contains general domain information such as dates of events and names of persons, locations, or organizations participating in the domain. Existing ontology builders either ignore this general information or they provide a limited implementation to extract it.

4.1 System Overview

The input to our builder (Figure 1) is a multiple-domain corpus. The first step is the ***clustering operation***, which divides the given corpus documents into clusters that are different among each other with high internal similarity. The next step is the ***conceptual layer generation***. In this step, we use Text2Onto to extract a separate ontology for each domain. Finally, the ***general layer generator*** uses each domain corpus and the conceptual layer ontology to extract relations among the concepts and the Named Entities in that domain.

4.2 The Conceptual Layer

The first step in constructing the conceptual layer is the clustering operation. We separate a multiple-domain corpus into various domain specific corpora such that the domain concepts are weighted based on their significance in that domain; also, the common concepts in different domains are separated. We favored a hierarchical clustering technique over a flat clustering one. That was because the number of resulting clusters should be known as a parameter in the latter. However, the number of corpus domains might be unknown in our case.

We employ the agglomerative hierarchical clustering technique (Manning et al., 2008). The technique starts with each document as a singleton cluster, and then it successively merges pairs of similar clusters until all clusters are merged into 1 cluster. We use the vector space model (Manning et al., 2008) to represent each document as a vector of terms' weights. The weight of a term *w* in a document *d* is calculated using the TF-IDF measure (Equation 1).

$$TFIDF(w,d) = TF(w,d) * log\frac{N}{DF(w)} \qquad (1)$$

Where *N* is the corpus size, *TF (w,d)* is the number of occurrences of the term *w* in the document *d*, and *DF (w)* is the number of documents containing the term *w*.

The similarity between 2 documents is calculated using the Cosine Similarity measure (Equation 2).

$$cosine(d1,d2) = \frac{V(d1)\,.\,V(d2)}{||V(d1)|| * ||V(d2)||} \qquad (2)$$

Where *V(d)* is the terms' weights vector for the document *d*, *||V(d)||* is the Euclidean length of the vector *V(d)*, and the numerator is the dot product of the 2 vectors.

The similarity between 2 clusters is calculated using the UPGMA measure (Steinbach et al., 2000) (Equation 3).

$$sim(C1, C2) = \frac{\sum_{d1 \in C1} \sum_{d2 \in C2} cosine(d1, d2)}{size(C1) * size(C2)} \quad (3)$$

We use the UPGMA measure to cluster a subset of DMOZ[2] data (1322 documents, in 4 domains), and it performs F-Measure of 0.86. Steinbach et al. (2000) describe how to calculate F-Measure for a hierarchy of clusters.

The combination similarity is the similarity of 2 merged clusters. We use this measure to cut the clusters hierarchy into M clusters by grouping ones having a minimum combination similarity of the threshold value €[3]. After clustering, we use Text2Onto to generate an ontology for each cluster (domain).

4.3 The General Layer

Text2Onto performs well in extracting ontology elements such as concepts, sub-class-of relations, instance-of relations, and part-of relations. Unfortunately, it performs inaccurately in extracting general domain information such as Named Entities and numeric information. There are 3 reasons for such misconception. First, proper nouns are not extracted as concepts. Second, numeric data is ignored. Third, restricted patterns are applied for the relations of Named Entities, that include only verb relations like [(NP |PNP) *verb* (NP|PNP)] and instance-of relations like [NP *such as* PNP], [*such* NP *as* PNP], and [PNP (NP)].

Because of the above reasons, we propose a highly flexible pattern based relation extractor. In our system, a pattern is a sequence of tags in the form of a regular expression. The possible tags are the normal POS tags like NN, VB, JJ, IN besides the following 5 tags CONCEPT, PERSON, LOCATION, ORGANIZATION, and DATE. This criterion is called Mixed Tagging. Currently, dates are the only data containing numbers extracted by our builder, but we can easily extend it to handle more numeric data.

The Mixed Tagging operation inputs are a document and the related conceptual ontology (Figure 2). The operation output is a mixed tagged document. The tagged text is then provided to the Relations Extractor to take out all

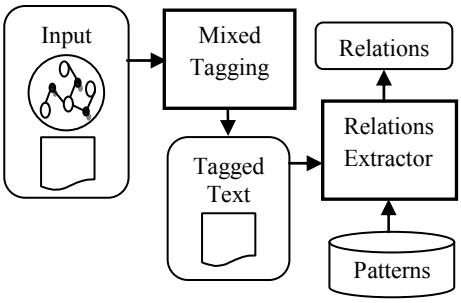

Figure 2. The General Relations Extraction.

relations matching our current predefined patterns. Example patterns are listed in Table 1; the first 2 patterns are verb relations, and the last 2 are noun relations.

The regular expression ([.{1,12}]){0,5} is used to limit the maximum number of tokens between the subject, the object, and the relation to 5 tokens. The expression [NN.?.?] matches any noun tag, and [VB.?] matches any verb tag.

After extracting the relations in all domain documents, the domain general ontology is created. It imports the corresponding conceptual ontology to model the relations among Named Entities and concepts.

([PERSON]) ([.{1,12}]){0,5}**([VB.?])+** ([.{1,12}]){0,5}**([CONCEPT])**
([ORGANIZATION])([.{1,12}]){0,5} **([DATE])**([.{1,12}]){0,5}**([VB.?])+**
([PERSON])([.{1,12}]){0,5} **([NN.?.?])+**([.{1,12}]){0,5}**([DATE])**
([NN.?.?])+([.{1,12}]){0,5}**([PERSON])** ([.{1,12}]){0,5}**([ORGANIZATION])**

Table 1. Sample Relation Patterns.

5 Question Answering System

Based on (Brank et al., 2005), a generated ontology can be evaluated using 4 different ways: 1) by a human who assesses it based on specific criteria, 2) by a comparison with the source data, 3) by a comparison with a golden standard, or 4) by using the ontology in an application and measuring the application performance. We chose the last option because the manual human assessment and the comparison with the source data are time consuming. Also, there is no golden standard ontology for a multiple-domain corpus.

Recently, researchers have studied the use of ontologies to extract answers to the user questions. AquaLog (Lopez et al., 2007) and

[2] http://www.dmoz.org/

[3] For clustering 600 AQUAINT documents, we use €=0.55 resulting in 7 Clusters (Secion 6.4).

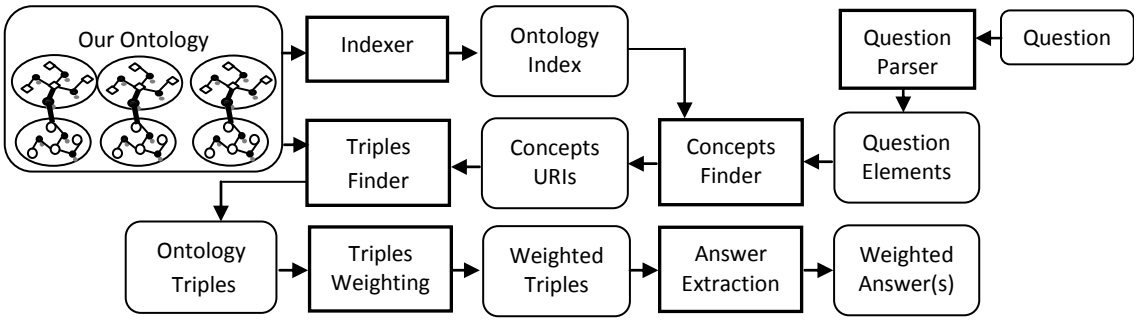

Figure 3. The Question Answering System.

PowerAqua (Lopez et al., 2009) are both ontology based Question Answering systems. PowerAqua extracts answers from various ontologies available on the web, unlike AquaLog, which extracts answers from 1 configurable ontology.

5.1 System Description

We implemented our simple Question Answering system handling who, when, where, and what questions. In the following, we describe the components of the system (Figure 3).

The Indexer: to make it easier for the system to locate the question concepts, an index is generated for our layered ontology. All concepts in different ontologies containing a certain stem are grouped in an index entry in the index file. The form of an index entry is as follows:

Stem,(Concept URI)+

The Question Parser: this component parses the user question, and it extracts 4 elements from it. First, the answer type; it can be PERSON, LOCATION, ORGANIZATION, DATE, or ANY based on the question type such as who, where, when, or what. Second, the answer restriction; it is used to limit the answers of *what* questions. For example, the answers for "what sport …?" question are restricted only to the sport types. Third, the question target; it defines the thing in which the question is interested. The fourth element is the relation; it contains the main verb(s) in the question. As an example, the elements of the question "*What sport does Jennifer Capriati play?*" are: the answer type (ANY), the restriction (sport), the question target (Jennifer Capriati), and the relation (play).

For a compound (2-clause) question such as *"What countries have Rhodes Scholars come from **and** has the Hale Bopp comet visible?*", each question clause is parsed as a separate question; finally, the answer extraction step intersects the answers of both clauses.

The Concepts Finder: using the ontology index, it locates concepts containing the stems of the question target and the restriction (if exists).

The Triples Finder: it extracts the triples which contain the question target concepts either as subjects or as objects. If the question is a definition question like "What is something?", the triple finder extracts only the sub-class-of triples.

The Triples Weighting: the triples are weighted based on their similarity to the question using our similarity criterion (Equation 4):

$$sim(Q,T) = \frac{\sum_{a \in Q} \underset{b \in T}{Lin(a,b)}}{L(Q) * L(T)} \quad (4)$$

Where Q and T are sets of the bag-of-words for the question relation and the triple relation respectively, $Lin(a,b)$ is a measure for the semantic similarity between a and b based on WordNet (Lin, 1998), and $L(x)$ is the number of elements in the set x.

The Answer Extraction: this component first filters out the triples mismatching the expected answer type. Then, if there is no restriction element, it extracts the answer from the weighted triples by considering the triple object if the question target is the subject, and vice versa. The extracted answer from a triple is assigned the same triple weight. If the question has a restriction element, the answer(s) will be limited to the sub concepts of the restriction element. A weight (Equation 5) is assigned to each sub concept s based on its similarity to the extracted triples as follows:

$$W(s) = \frac{\sum_{T \in R} sim(S,T)}{L(R)} \quad (5)$$

Where *R* is the set of extracted triples, *S* and *T* are the sets of bag-of-words for the sub concept and the triple relation respectively, *sim(S,T)* is calculated using Equation 4, and *L(R) is* the number of elements in *R*.

For a compound question, the list of resulting answers contains only the common answers extracted for the 2 clauses.

6 Evaluation and Discussion

In our evaluation, we assess: 1) the enhancement of the concepts' weights in a specific domain corpus, 2) the enhancement of modeling common concepts in different domains with different semantics, and 3) the performance of our Question Answering system. The assessment is done through a comparison between our approach and Text2Onto.

In the development of our builder, we used Text2Onto [4], Stanford Part-Of-Speech Tagger (POS Tagger)[5], Stanford Named Entity Recognizer (NER)[6], and Jena[7]. In the Question Answering system, we also used the Java WordNet Similarity Library (JWSL)[8]; it implements the Lin measure.

6.1 Data Set

Our evaluation is based on the AQUAINT[9] corpus (Graff, 2002). It is an English news corpus containing documents from the New York Times News Service, the Xinhua News Service, and the Associated Press Worldstream News Service. The Question Answering track in TREC[10] (The Text REtrieval Conference) provides a set of questions on AQUAINT corpus along with their answers.

6.2 Concepts Weights Enhancement

For this experiment, we generated a corpus for the 3 domains, namely Finance, Sports, and Movies, from AQUAINT documents, such that each domain has equal number of documents. We measured the concepts' significance weights when using Text2Onto to generate a single ontology for the whole corpus and when using our builder to generate 3 different domains ontologies. We consider 3 measures implemented in Text2Onto, namely the Relative Term Frequency (RTF), the Entropy, and the TF-IDF.

The RTF for a concept *w* is the probability of the concept occurrence in the corpus (Equation 6).

$$p(w) = \frac{No.\,of\ occurences\ of\ w}{No.\,of\ all\ corpus\ concepts} \quad (6)$$

The entropy and the normalized entropy for a concept *w* are calculated as follows (Equations 7 and 8 respectively):

$$Ent(w) = p(w) * \log p(w) \quad (7)$$

$$N\big(Ent(w)\big) = \frac{Ent(w)}{Min\ Ent - Max\ Ent} \quad (8)$$

In Section 4.2, we mention how to calculate the TF-IDF value for a term *w* in a document *d* (Equation 1). The TF-IDF weight and the normalized TF-IDF weight for a concept *w* in the whole corpus are calculated as follows (Equations 9 and 10 respectively):

$$TFIDF(w) = \frac{\sum_{d \in D} TFIDF(w,d)}{N} \quad (9)$$

$$N\big(TFIDF(w)\big) = \frac{TFIDF(w)}{\sqrt{\sum_{ci \in C} TFIDF(ci)^2}} \quad (10)$$

Where *D* is the set of documents containing *w*, *N* is the corpus size, and *C* is the set of all concepts in the corpus.

Since the concept weight is proportional to its occurrences in the corpus with respect to the other concepts, the fair distribution of the occurrences leads to precise weight calculation. In the specific domain corpus, the distribution is more reasonable than in multiple-domain corpus.

[4] http://code.google.com/p/text2onto/
[5] http://nlp.stanford.edu/software/tagger.shtml
[6] http://nlp.stanford.edu/software/CRF-NER.shtml
[7] http://jena.sourceforge.net/
[8] http://grid.deis.unical.it/similarity/
[9] http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2002T31
[10] http://trec.nist.gov/data/qa.html

Domain	Concept	Entropy		TF-IDF		RTF	
		Text2 Onto	Our Builder	Text2 Onto	Our Builder	Text2 Onto	Our Builder
Finance	Stock	0.181	0.999	0.053	0.103	0.001	0.020
	Trading	0.155	0.670	0.044	0.139	0.001	0.010
	Shares	0.100	0.670	0.036	0.139	0.000	0.010
	Economy	0.100	0.670	0.026	0.051	0.000	0.010
Sports	Sport	0.822	0.974	0.344	0.379	0.012	0.019
	Baseball	0.321	0.389	0.147	0.190	0.003	0.006
	League	0.299	0.363	0.134	0.174	0.003	0.005
	Football	0.205	0.251	0.085	0.111	0.002	0.003
Movies	Actor	0.525	0.613	0.150	0.194	0.007	0.022
	Movie Industry	0.230	0.362	0.098	0.263	0.002	0.011
	Music	0.205	0.326	0.085	0.230	0.002	0.009
	Home Video	0.038	0.066	0.012	0.032	0.000	0.001

Table 2. Concepts Weights Comparison between Our Builder and Text2Onto.

This fact can be verified easily from Table 2. The 3 measures give higher weights in the domain specific ontologies than in a single ontology for the whole corpus.

6.3 Modeling Common Concepts

To study the enhancement in modeling common concepts having different meaning in different domains, we chose 5 concepts as samples (Table 3). For each concept, we selected documents from AQUAINT and from the Wikipedia concerning the concepts in 2 different domains.

In this experiment, the single ontology generated by Text2Onto contains only 1 definition for each concept namely wall_street *is_a* institution, marijuana *is_a* drug, bear *is_a* mammal, jaguar *is_a* cat, and world_war *is_a* war. On the other hand, our builder maintains both concept definitions in different ontologies.

Concept	Definition 1	Definition 2
Wall Street	A financial Institution	A movie
Marijuana	A drug	A song
The bear	A Mammal	A movie
Jaguar	A big cat	A car
World War	A war	A museum

Table 3. Sample of Lopsided Concepts.

6.4 Question Answering Enhancement

The experiment includes common concepts definition questions, single-domain questions, and cross-domain questions.

To illustrate the effect of the common concepts misconception problem solved by our builder against Text2Onto, we generated 5 definition questions for the 5 concepts in Table 3, like "what is wall street?", "what is marijuana?"…etc.

For the single-domain questions, we used a subset of AQUAINT corpus composed of 600 documents clustered into 7 domains using combination similarity threshold value of 0.55. We selected 60 factoid questions from TREC 2004 questions having their answers in these documents. Examples of single-domain questions are:

- *Who discovered prions?*
- *When was the IFC established?*

In addition to factoid questions, TREC 2004 also includes list questions. The answers of each question are aggregated from multiple documents. We used these questions in generating 10 cross-domain questions. Each question combines 2 of TREC list questions such that the 2 list questions are in different domains. Examples of these questions are:

- *What cities have an Amtrak terminal and have Crip gangs?*
- *What countries are Burger King located in and have IFC financed projects?*

Evaluation Criteria: the accuracy (A) (Equation 11) is used for evaluating single-domain questions because each factoid question has only 1 correct answer.

$$A = \frac{No.of\ correct\ answers}{No.of\ questions} \quad (11)$$

The definition and cross-domain questions have multiple correct answers. The average Precision (P), Recall (R), and F-Measure (F) (Equations 12, 13, and 14 respectively) of all questions are used for our evaluation.

$$P = \frac{No.of\ correct\ answers}{No.of\ retrieved\ answers} \quad (12)$$

$$R = \frac{No.of\ correct\ answers}{No.of\ actual\ answers} \quad (13)$$

$$F = \frac{2 * P * R}{P + R} \quad (14)$$

Table 4 shows that, in the definition questions, we achieve F-Measure of 1, while Text2Onto achieves 0.5. This is because our builder maintains the 2 different definitions of each concept, unlike Text2Onto, which contains only one.

Questions Type	Our Ontology	Text2Onto Ontology
Definition Questions	P=1.0 R=1.0 F=1.0	P=0.5 R=0.5 F=0.5
Single-Domain	A=68%	A=0.05%
Cross-Domain	P=0.49 R=0.59 F=0.44	P=0 R=0 F=0

Table 4. Question Answering Evaluation.

In the single-domain questions, using our ontology, we could answer 41 questions while using Text2Onto ontology we could answer only 3 questions ("what particle is a quark?", "what are prions made of?", and "What is the treatment of cataract?"). The low coverage of Named Entities in Text2Onto hinders it from answering correctly any question of types Who, When, and Where. This indicates the enhancement introduced by the proposed general layer for modeling accurately more domain information. In the cross-domain questions, we achieve F-Measure of 0.44. None of the cross-domain questions are answered using Text2Onto ontology due to the mentioned Named Entity coverage problem.

Although our results are better than Text2Onto, there is a room for more improvements. There are 4 main sources for retrieving wrong or incomplete answers (Table 5). Some relations are not extracted because their elements (subject, relation, and object) are not near enough from each other in the text, so none of our patterns or Text2Onto patterns could match them. This is the source of 65% of the errors. Missed Named Entities or wrongly tagged ones cause 16% of the errors. Some relations are not extracted because co-reference has not been handled yet. That leads to 12% of the total errors. Finally, in the factoid questions, we consider the answer with the highest weight to be the correct answer; 7% of the answers are extracted but with lower weights.

Error Type	Error percentage
No matching pattern	65%
NER Error	16%
Co-Reference	12%
Low answer weight	7%

Table 5. Answer Error Sources.

Based on the mentioned experiments, our builder outperforms Text2Onto in Question Answering. In addition, it can be used skillfully to enhance other Natural Language Processing applications such as Information Retrieval from multiple-domain data. Our initial results using 220 queries on 600 AQUAINT documents records 0.35 F-Measure against Lucene[11], which achieves 0.18.

7 Conclusion and Future Work

This paper presents the misconception problems when interpreting a multiple-domain corpus in a single ontology. A novel ontology builder is presented handling these problems by generating separate domain ontologies describing core and general domain information.

Currently, we hand on improving our builder relation extractor to answer more TREC questions by automatically learning patterns from text and by handling co-reference. Moreover, we are working to enhance the performance of our Information Retrieval system.

[11] http://lucene.apache.org/

References

Benslimane, D., E. Leclercq, M. Savonnet, M.-N. Terrasse, and K. Yétongnon. 2000. *On the Definition of Generic Multi-layered Ontologies for Urban Applications.* In the International Journal of Computers, Environment, and Urban Systems, volume 24: 191-214.

Brank, Janez, Marko Grobelnik, and Dunja Mladenić. 2005. *A Survey of Ontology Evaluation Techniques.* In the Proceedings of the 8th International Multiconference on Information Society: 166-169.

Buitelaar, Paul, Daniel Olejnik, and Michael Sintek. 2004. *A Protégé Plug-In for Ontology Extraction from Text Based on Linguistic Analysis.* In the Proceedings of the 1st European Semantic Web Symposium: 31-44.

Cimiano, Philipp, and Johanna Völker. 2005. *Text2Onto - A Framework for Ontology Learning and Data-driven Change Discovery*. In the Proceedings of the 10th International Conference on Applications of Natural Language to Information Systems: 227-238.

Cimiano, Philipp, Peter Haase, York Sure, Johanna Völker, and Yimin Wang. 2006. *Question Answering on Top of the BT Digital Library.* In the Proceedings of the 15th International Conference on World Wide Web: 861-862.

Dumontier, Michel, and Natalia Villanueva-Rosales. 2007. *Three-Layer OWL Ontology Design.* In the Proceedings of the *2nd* International Workshop on Modular Ontologies. CEUR Workshop Proceedings, volume 315.

Graff, David. 2002. *The AQUAINT Corpus of English News Text*. Linguistic Data Consortium, Philadelphia.

Lin, Dekang. 1998. *An Information-Theoretic Definition of Similarity*. In the Proceedings of the 15th International Conference on Machine Learning: 296-304.

Lopez, Vanessa, Victoria Uren, Enrico Motta, and Michele Pasin. 2007. *AquaLog: An Ontology-driven Question Answering System for Organizational Semantic Intranets*. In the Journal of Web Semantics, volume 5: 72-105.

Lopez, Vanessa, Victoria Uren, Marta Sabou, and Enrico Motta. 2009. *Cross Ontology Query Answering on the Semantic Web: An Initial Evaluation*. In the Proceedings of the 5th International Conference on Knowledge Capture: 17-24.

Manning, Christopher D., Prabhakar Raghavan, and Hinrich Schütze. 2008. *Introduction to Information Retrieval*. Online edition. http://nlp.stanford.edu/IR-book/information-retrieval-book.html. Cambridge University Press.

Navigli, Roberto, and Paola Velardi. 2004. *Learning Domain Ontologies from Document Warehouses and Dedicated Web Sites.* In the Journal of Computational Linguistics, volume 30: 151-179.

Steinbach, Michael, George Karypis, and Vipin Kumar. 2000. *A Comparison of Document Clustering Techniques.* Technical Report #00-034, University of Minnesota.