Department of
Information Engineering
and Computer Science
DISI

UNIVERSITY
OF TRENTO - Italy

DISI - Via Sommarive 5 - 38123 Povo - Trento (Italy)
http://disi.unitn.it

IMPOSING A SEMANTIC SCHEMA FOR THE DETECTION OF POTENTIAL MISTAKES IN KNOWLEDGE RESOURCES

Vincenzo Maltese

June 2013

Technical Report # DISI-13-034
Version 1.0

Published also: ODBASE 2013

Imposing a semantic schema for the detection of potential mistakes in knowledge resources

Vincenzo Maltese

DISI – University of Trento, Trento, Italy

Abstract. Society is becoming a complex socio-technical ecosystem requiring novel ICT solutions to provide the basic infrastructure for innovative services able to address key societal challenges and to assist people in their everyday activities. To tackle such complexity, there is a pressing need for very accurate, up-to-date and diversity-aware knowledge resources which can guarantee that results of automatic processing can be trusted enough for decision making processes. As the maintenance of such resources turns out to be very expensive, we argue that the only affordable way to address this is by complementing automatic with manual checks. This paper presents a methodology, based on the notion of *semantic schema*, which aims to minimize human intervention as it allows the automatic identification of potentially faulty parts of a knowledge resource which need manual checks. Our evaluation showed promising results.

Keywords: knowledge quality, knowledge evaluation, knowledge selection

1 Introduction

Society is becoming a complex socio-technical ecosystem where novel ICT solutions are needed to provide the basic infrastructure for innovative services able to assist people in their everyday activities and to address key societal challenges. In order to tackle such complexity and provide prompt and reliable services, there is a pressing need for very accurate, up-to-date and diversity-aware knowledge resources [1] which can guarantee that results of automatic processing can be trusted enough to take decisions on top, both at individual (e.g. for daily planning) and societal levels (e.g. by policy makers for city planning). Such knowledge is fundamental for a better understanding of the world and to address such challenges. For instance, by providing live information about local transportation means and touristic attractions of a city, a knowledge resource can be employed by automatic tools to provide real-time information about optimized (e.g., in terms of personal interests, as well as price and time required) sightseeing routes to tourists and suggestions for better (e.g., in terms of reduced traffic and pollution) transportation planning to policy makers.

Unfortunately, so far attempts to achieve positive outcomes from such systems often failed to meet the expectations [33]. Above all, the Semantic Web [34] has been only marginally successful because not being yet capable of providing usable semantics, namely something up-to-date and reliable enough for decision making processes. While automatically built resources (e.g., developed by extracting knowledge from the Web) can scale up to millions of entities, they can hardly compete in accuracy w.r.t. manually built resources. In general, there is also a fragmentation of such knowledge resources which turn out to be hardly reusable and interoperable [26, 33]. Moreover, even very accurate knowledge would be inadequate if not constantly updated. General problems that need to be tackled with knowledge resources include:

- ***Incompleteness***. When they lack of important information, e.g., in terms of missing terminology, missing entities and relations between them, attributes and their values, or their time validity and provenance information. For instance, in terms of attributes they may lack of the latitude and longitude coordinates of a certain location; in terms of time validity they may lack of the time period in which *Ronald Regan* was the president of *USA*.
- ***Inconsistency***. When the information they provide about the individuals (the ABox) is in contrast with the constraints established (in the TBox), e.g., in terms of values out of the range defined for the attributes. For

instance, the altitude of a location might be wrongly set to 10,000 meters when there are actually no places higher than *Mount Everest* (8,848 meters).

- ***Incorrect information***. When despite the information they have is consistent, it is actually wrong. For instance, they might specify that the altitude of *Mount Everest* is only 848 meters.
- ***Outdated information***. When despite the information they have is consistent and correct, it is actually obsolete. For instance, they might specify that the president of *USA* is *Ronald Regan* while it is currently *Obama*.

This paper presents a methodology to the automatic detection of potential mistakes in knowledge resources, based on the notion of *semantic schema.* The schema takes into account the meaning of the terms in the resource and allows enforcing higher level constraints on the data contained in the resource. They include explicit disjointness between classes, and constraints on the domain and range of the attributes. For example, the schema might specify that an entity cannot be a person and a location at the same time or that while it is perfectly fine for a person to have a birthdate, it is clearly inappropriate to have an ISBN code. Violations to the schema are automatically detected and directed to manual checks. This approach is in line with the recent trend in complementing automatic processing with human intervention, which is leading to very promising research in social computing (e.g., FoldIt[1]), human computation (e.g., ESP games [31]) and crowdsourcing (e.g., Amazon Mechanical Turk[2]). We evaluated the approach on the YAGO *ontology* [3] which was selected because it does not have a fixed schema, and because its 2009 version has been never evaluated before. A similar experiment, performed on the GeoNames *database*, is described in [35].

The rest of this paper is organized as follows. Section 2 provides relevant state of the art. Section 3 introduces the notion of semantic schema and explains how to enforce it for the detection of potential mistakes. Section 4 briefly describes the YAGO ontology. Section 5 focuses on the definition of the semantic schema for YAGO. Section 6 explains how the dataset was prepared in order to enforce the schema as described in Section 7. Section 8 provides the evaluation. Finally, Section 9 concludes the paper by summarizing the work done and outlining the future work.

2 State of the art

Large-scale knowledge resources. In the recent years, several linguistic and knowledge resources have been built, either manually or automatically. WordNet[3], Cyc[4] and SUMO[5] are examples of manually built resources. WordNet is by far the most widespread. Though, among its drawbacks we can mention that it is not tailored for any particular domain and it is often considered too fine grained to be really useful [29]. Cyc is distributed along three levels from general to domain specific knowledge. SUMO is a formal general ontology whose extension, called MILO, covers individual domains. Resources of this kind tend to be accurate, but quite small in size. Domain specific resources are offered by Library Science communities, but they typically lack of explicit semantics [30]. Among (semi-)automatically generated resources we can mention DBPedia[6], YAGO and Freebase[7] offering millions of entities and facts about them. Resources of this kind tend to be much bigger in size, but their accuracy cannot be always considered satisfactory according to the task they need to serve [1] and therefore they may require some refinement.

Tools for the evaluation and improvement of knowledge resources. According to Gomez-Perez [4], the goal of ontology evaluation is to determine what it defines correctly, does not define, or even defines incorrectly. This should be done in two steps: *verification* and *validation*. The purpose of verification is to check the syntactic correctness, i.e. that the ontology comply with the syntax of the representation language used. The purpose of validation is mainly to check its *consistency* (when contradictory conclusions cannot be obtained), *completeness* (when it fully captures what it is supposed to represent of the real world) and *conciseness* (when it does not contain redundancies). Several tools have been developed at the purpose of evaluating ontologies. Syntax and consistency checks are typically performed

[1] http://fold.it/portal/
[2] http://mechanicalturk.blogspot.com/2007/02/you-can-help-find-jim-gray-from-home.html
[3] http://wordnet.princeton.edu/
[4] http://www.cyc.com/
[5] http://www.ontologyportal.org/
[6] http://dbpedia.org/
[7] http://www.freebase.com/

by ontology development toolkits, such as Protégé [21], OilED [22], OntoEdit [23], NeOn [28] and Swoop [24]. Preece and Shinghal [14] provide a survey of the programs used to anomaly detection, where an anomaly is the sign of probable errors. Several diagnostic tools have been developed. The Chimaera suite [15] offers checks for incompleteness, taxonomic analysis, and semantic evaluation. Ceuster et al. [17] present a semi-automatic approach to the detection and repair of anomalies in medical ontologies. It mainly focuses on incompleteness in terms of potentially missing relations, classes and entities. ODEval [5] detects potential taxonomical errors in terms of inconsistency, incompleteness and conciseness.

Specifically for OWL ontologies, [12, 13, 19] concentrate on inconsistency, i.e. on the detection, explanation and resolution of unsatisfiable concepts. Arpinar et al. [20] present a rule-based approach to define and automatically detect conflicts on property (like transitivity or asymmetry), class (like disjoint classes) and statement assertions (conflicting statements). In [16], a *symptom ontology* is proposed as a common language for describing errors and warnings that may indicate inconsistencies. Qadir & Noshairwan [18] concentrate on incompleteness and partition omissions.

Specifically for RDF ontologies, ODEval [5] allows checking for circularity and redundancy. Ding & Finin [25] and Hogan et al. [26] present two studies on large corpora of RDF data showing that the quality of available resources is far from being acceptable. The latter provide recommendations about how to avoid typical mistakes. The TW OIE tool [6] allows checking for syntax errors, logical inconsistencies, type mismatch in property ranges, logical redundancy and violation of number restrictions. These tools do focus on automatic detection and fixing of mistakes, but they allow enforcing only those constraints which can be expressed by the same representation language of the ontology under evaluation. They are instead of limited use for the identification of mistakes for which the representation language employed would not be sufficient to express the constraints.

3 Defining and enforcing the semantic schema

The quality of the data contained in a knowledge resource can heavily depend on the strategy employed for the data representation [32]. Databases ensure certain levels of data quality by enforcing integrity constraints, but it is not possible to directly codify domain knowledge in them. For instance, it is not possible to specify all the possible more specific classes of a certain class (e.g. the fact that what is valid for generic locations is also valid for lakes and mountains). On the other hand, the constraints that ontologies can specify depend on the expressiveness of the language used. While OWL is extremely powerful, the RDFS model has well-known limitations: even if it distinguishes between classes and instances, a class can be potentially treated as an instance [11]; it is not possible to explicitly represent disjointness between classes; transitivity cannot be enforced at the level of instances (see also [2]).

We compensate for the limitations of databases and RDF(S) ontologies by defining additional constraints that constitute what we call a *semantic schema*. *Higher level* quality control software modules are implemented to guarantee that the facts are consistent w.r.t. the additional constraints, even though these constraints cannot be expressed by the representation language employed (i.e., in the case of databases and RDFS). We build such schema on the data model described in [27] that provides the corresponding *language*. In the language, we identify the following sets:

- C is a set of *classes*;
- E is a set of *entities* that instantiate the classes in C;
- R is a set of binary *relations* relating entities and classes, including *is-a* (between classes in C), *instance-of* (associating instances in E to classes in C) and *part-of* (between classes in C or between entities in E) relations. We assume *is-a* and *part-of* to be transitive and asymmetric;
- A is a set of *attributes* associating entities to the data type values.

We then define a semantic schema as a set of constraints:

- on the domain and range of the attributes in A, such that the domain is always constituted by the entities in E which are instances of one or more classes in C and the range is a standard data type (e.g. Float, String);

- on the domain and range of the relations in R, such that both the domain and range are always constituted by the entities in E which are instances of one or more classes in C;
- about known disjoint classes.

We call *types* the subset of those classes in C which are explicitly mentioned as the domain or range of a relation in R or an attribute in A. Entities in E and facts about them constitute what in [27] is called the *knowledge*. It can be easily observed that the above addresses all the limitations we described for RDFS. In fact, the schema enforces a clear split between entities and classes, thus preventing a class to be used as source or target of a relation; it enforces disjointness between classes and transitivity between entities when it is the case.

The semantic schema is defined according to the expected content of the ontology. This may require an initial inspection of its content in terms of kinds of entities, their attributes and relations. For instance, if it is supposed to contain knowledge about locations and people we might define the language where C contains *location*, *person* and their more specific subclasses (e.g., *city*, *river* and *hill* for location; *professor*, *student* and *scientist* for person); E contains actual locations and persons (e.g., *Rome* the city and *Albert Einstein* the scientist); R contains *is-a*, *instance-of*, *part-of* and *birthplace* relations; A contains *latitude*, *longitude* and *birthdate* attributes. We might then define the schema where:

- **locations** can have *latitude* and *longitude* which are Floats, *part-of* relations between them, and nothing else;
- **persons** can have a *birthdate* which is a Date, a *birthplace* which is a *location*, and nothing else;
- locations and persons are disjoint.

Once the semantic schema is defined, the content of the knowledge resource is processed by enforcing the schema in two steps. With the first step each entity in the resource is assigned exactly one type X from the schema. The selection of X is performed by checking that:

1. ALL the classes associated to the entity have at least a candidate sense (a possible disambiguation) which is more specific or more general than X
2. ALL the attributes of the entity are allowed for the type X
3. X is the only type exhibiting properties 1 and 2

Entities failing this test (i.e., they do not fit in any X or in more than one X) are considered to violate the semantic schema and are spotted as potential mistakes. With the second step, and for those entities passing the test, corresponding classes, attributes and relations are disambiguated accordingly. For instance, an entity with classes *bank* and *institution* would be associated to the type *organization* (despite bank may also mean riverbank), while an entity with class *printer* and attribute *bornOnDate* would be associated to *person* (despite printer may also mean a device).

4 The YAGO ontology

The YAGO ontology [3] is automatically built by using WordNet noun synsets and the hypernym\hyponym relations between them as backbone and by extending it with additional classes, entities and facts about them extracted from Wikipedia infoboxes and categories. The YAGO model is compliant with RDFS. Entities are therefore described in terms of facts of the kind <source, relation, target>. Overall, 95 different relation kinds are instantiated in YAGO 2009 version generating around 29 million facts about 2.5 million entities. Quality control is guaranteed by ensuring that facts are consistent with the domain and range defined for the relations. For instance, for the entity *Elvis Presley* it includes the following facts:

Elvis_Presley	isMarriedTo	Priscilla_Presley
Elvis_Presley	bornOnDate	1935-01-08
Elvis_Presley	type	wordnet_musician_110340312
Elvis_Presley	type	wikicategory_Musicians_from_Tennessee

where *isMarriedTo* corresponds to a relation between entities, *bornOnDate* is a data attribute, and *type* connects an entity to a class. Classes are of three different kinds:

- ***WordNet classes***, with prefix “wordnet_”, correspond to WordNet synsets;

- ***Wikipedia classes***, denoted with the prefix "wikicategory_", correspond to Wikipedia categories which are linked to WordNet classes;
- ***YAGO classes***, such as "YagoInteger", are additional classes introduced to enforce type checking on the domain and range of the relations.

The linking of Wikipedia with WordNet is computed by automatically extracting and disambiguating the head of the sentence from Wikipedia categories. In most of the cases, as senses in WordNet are ranked, the first sense is assigned. E.g., the head of the category *Musicians from Tennessee* is *musician* that is disambiguated as wordnet_musician_110340312. The linking is maintained via the *subClassOf* relation.

5 Definition of the semantic schema on YAGO

As YAGO is a large-scale ontology not targeted to any specific domain, it is pretty hard to define a schema to capture the whole content of YAGO. Therefore, for demonstrative purposes, we decided to define a schema covering only a portion of YAGO. This proves the applicability of the approach, still requiring extending the schema in the future. We decided to focus on locations, organizations and people. To come up with a meaningful schema, we inspected its content and analyzed the definition of the relation kinds as they are given in the YAGO documentation. In doing so, we faced the following issues:

- **Lack of explicit semantics:** no explicit meaning of the attribute and relation names is given thus we found them to be ambiguous. For instance, the YAGO relation *hasHeight* can be interpreted as *height* (in the sense of stature) in case of persons, *altitude* in case of locations and *tallness* for buildings. They correspond to three different senses in WordNet. In fact, they are three different attributes. We address this problem by assigning a different sense from WordNet to each of the attributes and relations in our schema. Wikipedia classes also lack of explicit definition, thus they are ambiguous too. For instance, consider the class *Cemeteries in Canada*. There are several locations in the world named Canada, not necessarily the country. For this reason, we decided to only focus on WordNet classes.
- **Too broad domain and range**: in some cases, the domain and range of the defined relation kinds are too broad. For instance, the relation *isAffiliatedTo* is defined between any two generic entities; we rather believe that the domain should include only persons and organizations while the range should only include organizations. We address this by refining them in our schema.
- **Lack of latitude and longitude coordinates**: locations lack of latitude and longitude. As they would be extremely useful to determine whether an entity is a location, we extracted them from Wikipedia. Among available ones, we took the closest Wikipedia dump to the one used by the YAGO version used. The heuristics we used allowed us extracting 440,687 latitude\longitude pairs.
- **Lower than expected accuracy of the linking of Wikipedia to WordNet classes.** By analyzing 500 Wikipedia classes randomly taken from YAGO we found out that the accuracy of the linking is 82% (in the worst case) which is lower than the 95% found for the other YAGO versions. Additional details are given in the appendix. We address this problem by recomputing the linking.

After the initial inspection, we defined a meaningful language for the locations, organizations and persons in YAGO as follows:

- C contains *location*, *person, organization*, their more specific subclasses and their more general super-classes (e.g. *entity*, *physical object*) from WordNet. As we assume *facilities*, *buildings*, *bodies of water*, *geological formations*, *dry lands* and *geo-political entities* (such as countries and cities) to be locations, corresponding more specific subclasses are also contained in C. The set C does not contain Wikipedia classes.
- E is initially empty and it is later populated with entities from YAGO.
- R contains *is-a*, *instance-of*, *part-of* (the YAGO *locatedIn*) relations and the subset of *YAGO* relations whose domain and range intersects with the classes in C and where such relations were refined, disambiguated and renamed in order to identify corresponding synsets for them in WordNet. For instance, the YAGO relation *isAffiliatedTo* was renamed as *affiliation* defined in WordNet as "a social or business relationship", the domain was restricted to the union of person and organization and the range to organization.

- A contains the subset of *YAGO* relations whose domain intersects with the classes in C, the range is a standard data type and where such relations were refined, disambiguated and renamed in order to identify corresponding synsets for them in WordNet. For instance, *hasHeight* (being ambiguous) is mapped to the attribute *height* (in the sense of stature) in case of persons, *altitude* in case of generic locations and *tallness* for buildings. A is extended with *latitude* and *longitude* whose domain is location and range is Float.

We then defined the semantic schema where:

- **Persons, locations and organizations** can all have the attributes\relations corresponding to the following YAGO relations: {hasWebsite, hasWonPrice, hasMotto, hasPredecessor, hasSuccessor}
- **Persons** can also have: {hasHeight, hasWeight, bornOnDate, diedOnDate, bornIn, diedIn, originatesFrom, graduatedFrom, isAffiliatedTo, isCitizenOf, worksAt, livesIn, hasChild, isMarriedTo, isLeaderOf, interestedIn, influences, isNumber, hasAcademicAdvisor, actedIn, produced, created, directed, wrote, discovered, madeCoverOf, musicalRole, participatedIn, isAffiliatedTo, politicianOf}
- **Organizations** can also have: {hasRevenue, hasBudget, dealsWith, produced, created, hasNumberOfPeople, isAffiliatedTo, musicalRole establishedOnDate, hasProduct, isLeaderOf, createdOnDate, influences, participatedIn, isOfGenre}
- **Locations** can also have the following: {latitude, longitude, hasHeight, hasUTCOffset, establishedOnDate, hasArea, locatedIn, inTimeZone}
- **Geo-political entities** are more specific locations that can also have: { hasGini, hasPoverty, hasCapital, imports, exports, hasGDPPPP, hasTLD, hasHDI, hasNominalGDP, hasUnemployment, isLeaderOf, has_labour, dealsWith, has_imports, has_exports, has_expenses, hasPopulation, hasPopulationDensity, participatedIn, hasCurrency, hasOfficialLanguage, hasCallingCode, hasWaterPart, hasInflation, hasEconomicGrowth}
- **Facilities and buildings** are more specific locations that can also have: {hasNumberOfPeople, createdOnDate}
- Locations, persons and organizations are pairwise disjoint.

6 Preparation of the dataset

In order to be able to enforce the semantic schema, relevant facts about locations, organizations and persons from YAGO were extracted and preliminary imported into a relational database. As the database does not yet comply with the semantic schema defined in the previous section, we call this database the *intermediate schema*. In particular, classes are not yet linked to WordNet, attributes\relations are still ambiguous and their values are not yet checked for consistency w.r.t. the schema.

The selection of relevant knowledge was performed by following principles at the basis of *ontology modularization* techniques. d'Aquin et al. [8] define ontology modularization as the task of partitioning a large ontology into smaller parts each of them covering a particular sub-vocabulary. Doran et al. [10], define an ontology module as a self-contained subset of the parent ontology where all concepts in the module are defined in terms of other concepts in the module, and do not refer to any concept outside the module. They reduce module extraction to the traversal of a graph given a starting vertex that ensures in particular that the module is transitively closed w.r.t. the traversed relations. Cuerca Grau et al. [9] stress that the partitioning should preserve the semantics of the terms used, i.e. the inferences that can be made with the terms within the partition must be the same as if the whole ontology had been used. We understand modularization as the process of identifying self-contained portions of the ontology about specific entity types. In our work locations, organizations and persons were taken from YAGO by selecting all those entities whose WordNet class (identified through the *type* relation) is equivalent or more specific (identified through the *subClassOf* relation) than one of those in Table 1. The table also shows the amount of entities and Wikipedia classes found in each sub-tree.

Overall, we identified 1,568,080 entities that correspond to around 56% of YAGO. Selected entities, corresponding classes, alternative names in English and Italian and other related facts codifying their attributes and relations were then imported into the *intermediate schema*. Table 2 provides corresponding statistics. Notice that entities can belong to more than one class and this explains why the mere sum is bigger than 1,568,080.

WordNet Class	Entities	Wikipedia classes
wordnet_location_100027167	412,839	16,968
wordnet_person_100007846	771,852	67,419
wordnet_organization_108008335	213,952	19,851
wordnet_facility_103315023	83,184	8,790
wordnet_building_102913152	49,409	6,892
wordnet_body_of_water_109225146	36,347	1,820
wordnet_geological_formation_109287968	19,650	1,978
wordnet_land_109334396	8,854	805

Table 1. Number of entities and Wikipedia classes for the WordNet classes

Kind of object	Amount
Classes	3,966
Entities	1,568,080
instance-of relations	3,453,952
Attributes/Relations	3,229,320
Alternative English names	3,609,373
Alternative Italian names	220,151

Table 2. Kind and amount of objects in the intermediate schema

Notice that the classes in Table 2 do not correspond to the original Wikipedia classes as we recomputed them. In doing so, we directly associated entities to classes likely to correspond to those in C because syntactically matching with words in WordNet synsets. The class extraction was performed through the use of NLP tools, and specifically of a POS tagger developed and trained with the work presented in [7], and a BNF grammar generated to work on POS tagged Wikipedia classes. From our experiments the grammar turns out to be able to process from 96.1 to 98.7% of them according to the different sub-tree in which they are rooted, where the roots are the WordNet classes listed in Table 2. For the uncovered cases, we reused the YAGO linking. The final grammar, able to recognize *class names* and *entity names*, is as follows:

```
wikipedia-class ::= classes IN [DT] [pre-ctx] entity {post-cxt}* | classes
classes ::= class [, class] [CC class]
class ::= {modifier}* class-name
class-name ::= {NNS}+ | NNS IN {JJ}* NN [^NNP]
modifier ::= JJ | NN | NNP | CD | VBN
entity ::= {NNP}+ | CD {NNP}*
pre-ctx ::= ctxclass IN
post-ctx::= VBN IN {CD | DT | JJ | NNS | NN | NNP}* | CD | , entity | ctxclass |
            (ctxclass) | (entity [ctxclass])
ctxclass ::= {NN}+
```

For instance, from the Wikipedia class *City, towns and villages in Ca Mau Province* the grammar allows extracting the three classes *city*, *town* and *village* (while YAGO extracts *city* only), while from *Low-power FM radio stations* the grammar allows extracting *radio station* (while YAGO extracts *station* only). When multiple classes are extracted from a Wikipedia class, modifiers of the first class are assumed to apply to all classes. For instance, *Ski areas and resorts in Kyrgyzstan* means *Ski areas and ski resorts in Kyrgyzstan*. Some modifiers can explicitly (with NNP) or implicitly (with JJ) denote a named entity and are therefore filtered out. An example for the first kind is *Hawaii countries*, while an example of the second kind is *Russian subdivisions*. Less frequent POS tags found (e.g. NNPS and VBG) were not included in the grammar.

7 Enforcing the schema on YAGO

As reported in Table 3, by enforcing the defined schema we could unambiguously assign a type to 1,389,505 entities corresponding to around 89% of the entities in the intermediate schema; 20,135 entities were categorized as ambiguous because more than one type X is consistent with the classes and the attributes of the entity; 158,441 entities were not categorized because of lack of or conflicting information.

Type	Amount
Person	719,551
Organization	154,153
Location	284,267
Geological formation	14,426
Body of water	34,958
Geo-political entity	100,910
Building and facilities	81,240
Total:	**1,389,505**

Table 3. Type assignment to the entities in the intermediate schema

Entity classes were disambiguated and assigned elements in C according to the type X associated to them. Notice that this means that we use classes and attributes to restrict candidate senses (i.e., they disambiguate each other). Specifically, we always assigned to them the WordNet sense more specific (or more general) than the type X. In case more than one with such property was available (a few cases), we assigned the sense with highest rank among them. Similarly, attributes and relations were mapped to the corresponding attribute in A or relation in R according to the type X assigned. Values are considered to be correct only if they are consistent with the corresponding range constraints. The remaining values would be again target of the crowdsourcing.

8 Evaluation

With the initial selection 1,568,080 entities and related facts were extracted from YAGO and imported into the intermediate schema. By enforcing the schema:

- **CASE I:** 1,389,505 entities (around 89% of the imported entities) were assigned exactly one type X;
- **CASE II:** 20,135 entities were assigned as ambiguous, i.e. more than one type is consistent with the classes and the attributes of the entity;
- **CASE III:** 158,441 entities were not categorized because of lack of or conflicting information.

We then evaluated the quality of our class disambiguation w.r.t. the one in YAGO 2009. Notice that as we recomputed the entity classes by extracting them from the Wikipedia classes, they might differ, also in number, w.r.t. those in YAGO. Notice that classes were disambiguated only in case I. For this case, the accuracy of the type assignment was also evaluated.

CASE I. Over 100 randomly selected entities our assignment of the type is always correct, while our disambiguation of their 250[8] classes is **98%** correct (5 mistakes). By checking the Wikipedia classes of these entities in YAGO, we found out that the corresponding linking of their 216 classes is **97.2%** correct (6 mistakes). The mistakes tend to be the same, for instance we both map crater to volcanic crater instead of impact crater; manager to manager as director instead of manager as coach; captain to captain as military officer instead of captain as group leader.

[8] Notice that the accuracy of YAGO versions was evaluated on a much smaller sample. As it can be clearly seen from http://www.mpi-inf.mpg.de/yago-naga/yago/statistics.html the YAGO *type* relation was evaluated on 55 instances only.

CASE II. 50 random entities were selected among those that we categorized as ambiguous. We found out that the accuracy of the linking of their Wikipedia classes in YAGO is **72.3%** (18 mistakes over 65). Mistakes include for instance bank as river slope instead of institution; ward as person instead of administrative division; carrier as person instead of warship; division as military division instead of geographical. Yet, **10%** of these entities (5 over 50) are neither locations, nor organizations, nor persons. They are in fact reports about the participation of some country to some competition while they are treated as countries (e.g., *Norway in the Eurovision Song Contest 2005*). Further **4%** of them (2 over 50) are not even entities. One is actually a list of categories and the other a list of countries. Overall **14%** of them look therefore wrong. In addition, among the 72.3% of the cases considered as correct there are actually 18 controversial cases where it is not clear if the class is really meant as geographical or political division (e.g. subdivision); it is not clear if the class is meant as geographical or political entity (e.g. country); it is not clear if the class is meant as organization or building (e.g. hospital). We believe that these cases might be due to the phenomenon of *metonymy* which generates very fine grained senses in WordNet.

CASE III. 50 random entities were selected among those that we preferred to do not assign any type because of lack of information (e.g., the entity has only one class and no attributes) or presence of conflicting information (i.e., classes or attributes of different types). We found out that the accuracy of the linking in YAGO for these cases is **86.14%** (14 mistakes over 101). Mistakes include unit as unit of measurement instead of military unit; model as fashion model instead of mathematical model. Though, **72%** of the candidates (36 over 50) contain mistakes or they are not even entities. They include for instance entities which are both animals and persons (e.g., we found 137 persons as fishes and 4,216 as dogs); entities which are both organizations and persons; or even sex and political positions marked as locations.

Thus, the evaluation confirms the need to manually inspect entities falling in case II and case III as their quality is significantly lower than those in case I.

9 Conclusions

Starting from the observation that individual and societal decision making processes require very accurate, up-to-date and diversity-aware knowledge resources, we presented an automatic semantic schema-based approach for the identification of those parts of a knowledge resource which are particularly noisy and need, with higher priority w.r.t. other parts, to be manually inspected and fixed. Dually, it allows identifying those parts of higher quality that can be already trusted enough. In this way, human involvement - very costly in general - can be reduced by directing the attention to the lower quality parts.

The approach was evaluated on the YAGO ontology in its 2009 version, a large-scale knowledge resource not targeted to any specific domain, with no fixed schema and never evaluated previously. As proved by the final figures, enforcing the schema allowed identifying portions of the ontology which are particularly noisy and that would benefit from further (manual) refinement. Higher quality portions (knowledge falling in case I) have been selected and imported into Entitypedia [1], a knowledge base under development at the University of Trento.

The future work will focus on (a) the extension of the schema to have a higher coverage on YAGO (e.g., by defining types for books, movies, events) and (b) the design of crowdsourcing tasks necessary to refine the potentially noisy parts.

Acknowledgements. The research leading to these results has received funding from the CUbRIK Collaborative Project, partially funded by the European Commission's 7th Framework ICT Programme for Research and Technological Development under the Grant agreement no. 287704. Thanks to Fausto Giunchiglia for his suggestions, Biswanath Dutta for the early work in analyzing YAGO, Aliaksandr Autayeu for his support for the NLP tools, Feroz Farazi, Mario Passamani and the other members of the KnowDive group for the technical assistance.

References

1. Giunchiglia, F., Maltese, V., Dutta, B. (2012). Domains and context: first steps towards managing diversity in knowledge. Journal of Web Semantics, 12-13, 53-63.
2. Maltese, V., Farazi, F. (2011). Towards the Integration of Knowledge Organization Systems with the Linked Data Cloud. UDC seminar.
3. Suchanek, F. M., Kasneci, G., Weikum, G. (2008). YAGO: A Large Ontology from Wikipedia and WordNet. Journal of Web Semantics, 6 (3), 203-217.
4. Gomez-Perez, A. (2001). Evaluation of ontologies. International Journal of Intelligent Systems, 16 (3), 36.
5. Corcho, O., Gomez-Perez, A., Gonzalez-Cabero, R., and Suarez-Figueroa, C. (2004). ODEVAL: a tool for evaluating RDF(S), DAML + OIL, and OWL Concept Taxonomies. In Proceedings of the 1st Conference on AI Applications and Innovations, 369–382.
6. Tao, J., Ding, L., McGuinness, D.L. (2009). Instance data evaluation for semantic web-based knowledge management systems. In Proceedings of the 42th Hawaii International Conference on System Sciences (HICSS-42). 5-8.
7. Autayeu, A. Giunchiglia, F., Andrews, P. (2010). Lightweight parsing of classifications into lightweight ontologies. In Proceedings of the European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2010).
8. d'Aquin, M., Schlicht, A., Stuckenschmidt, H., Sabou, M. (2007). Ontology modularization for knowledge selection: experiments and evaluations. In Proceedings of the 18th International Conference on Database and Expert Systems Applications, 874-883.
9. Cuenca Grau, B., Horrocks, I., Kazakov, Y., Sattler, U. (2008). Modular reuse of ontologies: Theory and practice. Journal of Artificial Intelligence Research, 31, 273–318.
10. Doran, P., Tamma, V., Iannone, L. (2007). Ontology module extraction for ontology reuse: an ontology engineering perspective. In Proceedings of the 16th ACM Conference on Information and Knowledge Management (CIKM), 61-70.
11. Brickley, D., Guha, R. V. (2004). RDF Vocabulary Description Language 1.0: RDF Schema, (Editors), W3C Recommendation.
12. Parsia, B., Sirin, E., Kalyanpur, A. (2005). Debugging OWL ontologies. In Proceedings of WWW, 633-640.
13. Kalyanpur, A., Parsia, B., Sirin, E., Hendler, J.A. (2005). Debugging unsatisfiable classes in OWL ontologies. Journal of Web Semantics 3(4), 268–293.
14. Preece, A. D., Shinghal, R. (1994). Foundation and application of knowledge base verification. International Journal of Intelligent Systems, 9 (8), 683–701.
15. McGuinness, D.L., Fikes, R., Rice, J., Wilder, S. (2000). An environment for merging and testing large ontologies. In Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR2000), 483–493.
16. Baclawski, K., Matheus, C.J., Kokar, M.M., Letkowski, J., Kogut, P.A. (2004). Towards a symptom ontology for semantic web applications. In Proceedings of ISWC, 650–667.
17. Ceusters, W., Smith, B., Kumar, A., Dhaen, C. (2003). Mistakes in Medical Ontologies: Where Do They Come From and How Can They Be Detected? In Pisanelli DM (ed) Ontologies in Medicine. In Proceedings of the Workshop on Medical Ontologies, 2003. IOS Press, Studies in Health Technology and Informatics, 2004, 145-164.
18. Qadir, M.A., Noshairwan, W. (2007). Warnings for Disjoint Knowledge Omission in Ontologies. 2nd Int. Conf. on Internet and Web Applications and Services, IEEE, 45-49
19. Ovchinnikova, E., Wandmacher, T., Kuhnberger, K. (2007). Solving terminological inconsistency problems in ontology design. Interoperability in Business Information Systems, 2(1), 65–80.
20. Arpinar, I.B., Giriloganathan, K., Aleman-Meza, B. (2006). Ontology quality by detection of conflicts in metadata. 4th International EON Workshop.
21. Noy, N., Sintek, M., Decker, S., Crubezy, M., Fergerson, R., Musen, M. (2000). Creating semantic web contents with Protégé-2000, IEEE Intelligent Systems.
22. Bechhofer, S., Horrocks, I., Goble, C., Stevens R. (2001). OilEd: a reason-able ontology editor for the Semantic Web. Advances in Artificial Intelligence, 396-408.
23. Sure, Y., Erdmann, M., Angele, J., Staff, S., Studer, R., Wenke, D. (2002). OntoEdit: Collaborative ontology development for the Semantic Web. In Proceedings of ISWC

24. Kalyanpur, A., Parsia, B., E.Sirin, Cuenca-Grau, B., Hendler, J. (2006). Swoop: A web ontology editing browser. Journal of Web Semantics, 4 (2), 144-153.
25. Ding, L., Finin, T. (2006). Characterizing the Semantic Web on the Web. Proc. of ISWC.
26. Hogan, A., Harth, A., Passant, A., Decker, S., Polleres, A. (2010). Weaving the pedantic web. In the 3rd International Workshop on Linked Data on the Web, WWW.
27. Giunchiglia, F., Dutta, B., Maltese, V., Farazi, F. (2012). A facet-based methodology for the construction of a large-scale geospatial ontology.Journal of Data Semantics,1(1),57-73.
28. Haase, P., Lewen, H., Studer, R., Tran, D., Erdmann, M., d'Aquin, M., Motta, E. (2008). The neon ontology engineering toolkit. WWW Developers Track.
29. Mihalcea, R., Moldovan, D. I. (2001). Automatic generation of a coarse grained wordnet, NAACL Workshop on WordNet and Other Lexical Resources.
30. Soergel, D., Lauser, B., Liang, A., Fisseha, F., Keizer, J., Katz, S. (2004). Reengineering thesauri for new applications: the agrovoc example, Journal of Digital Information, 4.
31. Von Ahn, L. (2006). Games With A Purpose. IEEE Computer Magazine, 96-98.
32. Martinez-Cruz, C., Blanco, I., Vila, M. (2011). Ontologies versus relational databases: are they so different? A comparison. Artificial Intelligence Review, 1–20.
33. Jain, P., Hitzler, P., Yeh, P. Z., Verma, K., Sheth, A. P. (2010). Linked data is merely more data. Linked Data Meets Artificial Intelligence, pp. 82-86.
34. Berners-Lee, T., Hendler, J., Lassila, O. (2001). The Semantic Web. Scientific American.
35. Maltese, V., F. Farazi (2013). A semantic schema for GeoNames. INSPIRE Conference.

Appendix: manual evaluation of the accuracy of the YAGO linking

The evaluation was conducted on 500 Wikipedia-WordNet pairs randomly selected and by incrementally computing the figures at blocks of 100 classes. Final findings are summarized in Table 4. The first column shows the amount of classes analyzed. The second column shows the number of mistakes found in the corresponding block. The third column shows how the percentage varies after each block of categories is analyzed. The final accuracy we found is 87%. For instance, the class *Indoor arenas in Lithuania* is wrongly linked to the first WordNet sense of *arena* that is "a particular environment or walk of life" while we believe that the correct one should be the third sense "a large structure for open-air sports or entertainments". However, as reported in the fourth and fifth columns, there are some cases in which, despite the proximity of the right sense, a more general (MG) or a more specific (MS) sense would be more appropriate. The last two columns show the percentage of mistakes updated taking into account such cases with accuracy varying from 85% to 82%. For instance, *Coal-fired power stations in Uzbekistan* is linked to *station* defined as "a facility equipped with special equipment and personnel for a particular purpose", while a more appropriate class is clearly *power station* defined as "an electrical generating station".

# classes	# mistakes	overall % mistakes	# MG senses	# MS senses	% mistakes MG	% mistakes MG + MS
100	11	0.11	2	1	0.13	0.14
200	7	0.09	2	3	0.11	0.13
300	12	0.10	0	2	0.11	0.13
400	16	0.12	2	3	0.13	0.15
500	20	0.13	4	3	0.15	0.18

Table 4. Manual evaluation of the YAGO linking

We also found 4 mistakes due to lack of senses in WordNet. For instance, *Eredivisie derbies* was mapped to the only sense of WordNet available for derby, i.e., "a felt hat that is round and hard with a narrow brim", while we believe that it refers more to football derby. They were not counted as mistakes in the table above.