# Ontology Population for Open-Source Intelligence (Discussion Paper)

Giulio Ganino, Domenico Lembo, Massimo Mecella, and Federico Scafoglieri

DIAG, Sapienza Università di Roma, Italy
{lembo,ganino,mecella,scafoglieri}@diag.uniroma1.it

**Abstract.** We present an approach based on GATE (General Architecture for Text Engineering) for the automatic population of ontologies from text documents. We describe some experimental results, which are encouraging in terms of extracted correct instances of the ontology. We then focus on a phase of our pipeline and discuss a variant thereof, which aims at reducing the manual effort needed to generate pre-defined dictionaries used in document annotation. Our additional experiments show promising results also in this case.

## 1 Introduction

Open-Source INTelligence (OSINT) is intelligence based on publicly available resources, such as news sites, blogs, forums, etc. OSINT is nowadays used in many application scenarios, like, for instance, security, market intelligence, or statistics. A major issue in using *Internet as a data source* is that Web data come mainly in the form of free text, thus with no structure and formal semantics. This therefore requires that two problems have to be faced, that is, how to derive structured information from unstructured text, and how to interpret the derived information according to a precise semantics.

Towards a comprehensive solution to this problem, in this paper we investigate how to populate a domain ontology with the information extracted from documents crawled from the Web. Ontologies are indeed nowadays recognized as the best means to represent domain knowledge at the conceptual level, and thus are particularly suited to define the concepts and relationships of interest for an application. Thus, structuring Web data according to the predicates and axioms defined in an ontology turns out to be particularly effective to our aims, even in the light of the reasoning abilities ontologies allow for [2]. It is worthwhile however to point out that we do not consider here the task of extracting (the intensional/schema level of) ontologies in an automatic way from text documents. Domain ontologies indeed have usually a very complex intensional structure, and their design is typically carried out manually, since automatic ontology construction approaches are in general not able to deal with all the needs of the specific application domain at hand [7, 1]. In this work we instead pursue an approach for information extraction from text that assumes that a domain ontology is already available, and that text should be mined to extract instances of ontology predicates.

SEBD 2018, June 24-27, 2018, Castellaneta Marina, Italy. 

To solve this task, *Named Entity Recognition* (NER) [3] can be initially adopted, but this approach is mainly focused on instantiating specific concepts (persons, places, or organizations), without considering relationships among them. *Relational Information Extraction* attempts in addition to identify and instantiate the relationships existing among concepts. We will adopt both approaches, by relying on open-source technologies.

Specifically, we make use of the GATE system[1], given its ability to customize the underlying architecture components and to incorporate other external components developed by third parties. Information extraction in GATE is carried out through different stages, each depending on the contingent needs of the user, who can either adopt existing dictionaries (a.k.a. Gazetteers) or create different extraction rule sets through the adoption of Java Annotation Patterns Engine (JAPE) language [4]. GATE has become popular in the last years, especially in relation to information extraction from English documents. To some extent, it also supports other languages, primarily thanks to the dictionaries created and then shared on the platform by its many users.

We have tested our techniques within the XASMOS and RoMA projects, involving the Leonardo company and the Sapienza research center on Cyber Intelligence and Information Security. In the projects, we have considered the case study "Mafia Capitale", from the name of an important inquiry of 2015 that received a lot of attention by the Italian media, and thus turned out to be a valid testbed (for both number of available Web documents and significance of the domain). For this work, we have created specific dictionaries and JAPE rules for the Italian language to instantiate, with the information acquired from text documents, an ontology designed for our case study. We present the case study successfully applied to more than 2600 documents crawled from the Web.

We have experienced that some of the tasks we conducted, such as dictionary or JAPE rule definition, are rather domain specific and time-consuming. We investigated how to refine our approach so that the time needed for these tasks is reduced and the solution adopted can be more easily reused in different contexts. In particular, after describing our approach and the use case, we propose a general technique for dictionary construction, which relies on the extraction of gazetteers lists from the open KB Wikidata[2].

This paper is an extended abstract of [6].

## 2 Background

**Natural Language Processing (NLP).** NLP aims at solving problems related to the automatic generation and understanding of human language [9]. The process carried out by NLP is made up of several steps:

- *Tokenization*: This task breaks down the raw text in tokens, which can be words, spaces or dots.
- *Part of Speech (POS)*: This phase aims at labeling each word with a unique tag indicating its syntactic role, i.e. Noun, Verb, or Pronoun.
- *Named Entity Recognition (NER)*: NER labels atomic elements in a sentence into categories (such as "Person" or "Location") through the application of specific rules or statistical machine learning techniques [8].

[1] `https://gate.ac.uk/`

[2] `https://www.wikidata.org/`

– *Semantic Role Labelling (SRL)*: SRL gives a semantic role to a syntactic constituent of a sentence, and add further labels to words in the documents. SRL aims at understanding the meaning of an entire sentence starting from the meaning of each word taken in isolation and the relationship existing among the words [11].

**General Architecture for Text Engineering (GATE).** GATE, is an architecture, a framework and a development environment for Language Engineering (LE) [5]. It has a component-based model, allowing for easy combination of Processing Resources (PRs), thereby facilitating comparison of alternative configurations of the system or different implementations of the same module (e.g., different parsers). GATE comprises a core library and a set of reusable LE modules, which perform basic language processing tasks, such as POS and semantic tagging. This provides a good starting point for new applications. The modules used in this work are described in Section 3.

**Ontologies.** An ontology is a formal description of an abstract, simplified view of a certain portion of the world [7]. Ontologies can be naturally used to represent knowledge on the web, where they are mainly adopted to add semantics to data. This also enables the usage of the reasoning mechanisms ontologies are equipped with [2]. The importance of ontologies to interpret and structure Web data is testified also by the standardization effort carried out by the W3C, which led to the definition of OWL[3].

As usual in ontologies, OWL distinguishes between *intensional* and *extensional* knowledge. Intensional knowledge is given in terms of logical axioms involving Classes (a.k.a. concepts) and properties, which are of two types, ObjectProperties (a.k.a. binary relationships or roles) and DataProperties (a.k.a. attributes). Classes denote sets of objects, ObjectProperties denote binary relations between objects, whereas DataProperties denote binary relations between objects and values from predefined datatypes. Person, livesIn and personAge are examples, of Class, ObjectProperty, and DataProperty, respectively. At the extensional level, an OWL ontology is a set of assertions about its instances. For example, ClassAssertion(Person John) indicates that the individual John is an instance of Person, whereas ObjectPropertyAssertion(livesIn John NY) specifies that the pair of individuals (John, NY) is an instance of livesIn.

## 3 Approach

Our approach comprises two phases: *Semantic Annotation* and *Ontology Population*.

**Semantic annotation.** In this stage we create annotations, i.e., metadata that indicate properties of the text contained in the analyzed documents. At the end of this phase, the annotations will allow us to identify those entities that are indeed instances of the classes and properties of the ontology given as input to our pipeline. In our GATE-based approach, this phase relies on several PRs, which are available as GATE plugin-ins, possibly provided by third-party organizations[4]. Such resources are described below.

1. *GATE Unicode Tokeniser*: this is a PR of the GATE core library used to split the text in *Tokens* and *SpaceTokens*, where the latter denote spaces among single terms, whereas the former are numbers, punctuations, symbols, and words.

[3] Ontology Web Language (`https://www.w3.org/TR/owl2-primer/`)

[4] `https://gate.ac.uk/gate/doc/plugins.html`

2. *RegEx Sentence Splitter*: it is another PR of the GATE core library. It is used to divide documents into *sentences*. It is essentially language-independent, and thus can be used as is for very many languages, including Italian. It is implemented in Java, and is based on regular expressions that define syntactic rules for sentence identification. At the end of this phase two new annotations are added to the document, i.e., *Sentence*s and *Split*s, the latter indicating separation between sentences.
3. *TreeTagger POS*[5]: it is a component for document annotation with POS and lemma information, developed at University of Stuttgart. It is a Markov Model tagger which makes use of a decision tree.
4. *Gazetteer* [4]: this GATE built-in component annotates the documents on the basis of a set of lists of words, such as names of cities or organisations, or abbreviations for types of companies (e.g., ltd., Corp., Inc.). Each list can be associated with so-called major and minor types. These types correspond to categories, such that minor types are more specific than the corresponding major types. If the document contains a string matching with an element of a Gazetteer list, the component annotates the string with the major and minor type of this list. In case the string has more than one match, major and minor types of all the matching lists are added.
5. *JAPE Transducer*: this component is used to import user-written JAPE rules into the GATE platform. A JAPE program is constituted by a set of pattern/action rules, such that the left-hand side of a rule consists of an annotation pattern description, and the right-hand side consists of an annotation manipulation statement. The JAPE language allows to recognize regular expressions among the annotations produced from the PRs that run before the JAPE Transducer. Once the expression is found, a further annotation referring to the searched patterns/entities is added to the document [4].

**Ontology population.** It is the process of inserting instances into a domain ontology. To this aim we used three PRs, two of which are JAPE-based modules that essentially convert annotations created in the semantic annotation phase so that they are compatible with the format used by the third PR, called *OwlExporter*[6], which is the PR that concretely extracts ontology instances in the form of OWL assertions from the annotated text [6]. The OwlExporter manages two ontologies: a domain specific ontology, like the one we developed for our use case, and a domain-independent NLP ontology, which contains concepts commonly used in LE, like paragraphs, sentences, or noun phrases [10].

## 4 Case Study

We used Web Content Extractor[7] to retrieve articles concerning the "Mafia Capitale" inquiry and appeared on the Web portals of some major Italian newspapers between June 16, 2015, and February 29, 2016. The crawling phase generated 2657 articles.

We have then defined a domain ontology to represent some aspects of interest for the domain at hand. Our ontology is defined on a alphabet of 21 Classes, 9 ObjectProperties and 14 DataProperties. We have then manually created specific Gazetteer lists and JAPE

[5] `http://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger`
[6] `http://www.semanticsoftware.info/owlexporter`
[7] `http://www.newprosoft.com/web-content-extractor.htm`

**Table 1.** Results of test #1

| Words | Annotations | CA | OPA | Correct | Missing | Precision | Recall |
|---|---|---|---|---|---|---|---|
| 6.084 | 414 | 404 | 10 | 403 | 98 | 97% | 78% |

rules for document annotation through the GATE PRs described in Section 3. We remark that the both Gazetteer lists and JAPE rules are specific for the Italian language.

To evaluate our approach, we carried out two tests, both performed on a MacBook Pro 9.2, with Intel Core i7 2,90 GHz and 8GB RAM. In the first test, we have chosen 10 documents, in a random way, among those produced by the crawling phase, and have asked a domain expert to annotate them, to identify all and only the instances of the ontology predicates that can be extracted by these documents. We have then compared the annotations produced by our pipeline with the annotations of the domain expert, by computing the *Precision* ($P$) and the *Recall* ($R$). We remind that $P = C/(C + S)$, where $C + S$ is the number of annotations produced through our pipeline, some of which are indeed correct ($C$), if compared with the annotations identified by the expert, whereas others are spurious ($S$) (i.e., false positive), whereas $R = C/(C + M)$, where $M$ denotes the annotations missed by our pipeline, with respect to those identified by the expert (i.e., false negative). The results of this first test are given in Table 1, where *Words* indicates the number of words contained in the analyzed documents, *Annotations* is the total number of annotations produced by our pipeline, i.e., $C + S$, *CA* and *OPA*, denote the number of Class and ObjectProperty assertions extracted, respectively (to keep the expert annotation task simple, we did not consider DataProperty annotations).

We got excellent Precision and good Recall, a significant number of class assertions, but few ObjectProperty assertions. As for the precision, the errors we obtained were only due to the wrong annotations associated to the word "Marino", which is both a city and the last name of a Rome ex-mayor, since our JAPE rules were not able to disambiguate this situation. We notice however, that such kind of problem can be solved by adding other PRs to our pipeline, like the Pronomial Coreference (PC)[8], which provides an annotation on pronouns that refer to already annotated entities. This PR is currently available in GATE only for the English language, and some extension is needed to make it work also for the Italian.

The value we obtained for the Recall is mainly affected by the fact that we were not able to identify many ObjectProperty assertions. Whereas for this aspect there is much space for improvement, e.g., by adding further JAPE rules, we believe that the usage of the PC mentioned above can be helpfull also to increase the Recall, since many links between entities can be discovered by correctly annotating pronouns.

In the second test we have processed in our pipeline all the 2657 articles produced by the crawling phase. The results are shown in Table 2, where also the overall running time needed to process all documents is given. In the table, *DPA* denotes the number of DataProperty assertions we were able to extract. Due to the large number of documents used in this test, we could not measure overall precision and recall, but we mainly used this test to evaluate the impact of our approach on a large text corpus. Our results are

[8] `https://gate.ac.uk/sale/tao/splitch6.html#sec:annie:pronom-coref`

**Table 2.** Results of test #2

| Words | CA | OPA | DPA | Exec.time (sec.) |
|---|---|---|---|---|
| 1.285.290 | 38.452 | 419 | 99.145 | 1.948,88 (∼30 mins) |

encouraging, in particular for the number of CA and DPA extracted. Furthermore, to get an idea of the quality of the result, we measured the precision on the 1% of the data in the output (involving a domain expert) and we have got a value of 94%.

## 5 Simplifying Gazetteer Lists Generation

As said in Section 3, the Gazetteer PR annotates the documents on the basis of a set of lists of words, used to find occurrences of these words in the text, (e.g., for the NER task). In our use case, it was crucial to create all the Gazetteer lists we used, because all the articles we analyzed are in Italian, and when we started our project no reusable support for Gazetteer lists for this language was available. To create the lists related to the domain entities of "Mafia Capitale", we use two methods:

- based on *Open Data Sources (ODS)*: Lists have been downloaded (and then manually refined) from open data Web sites. In particular, we did this for the first and last name of individuals belonging to a specific category, e.g., the members of the Government of the city of Rome. To this aim we have downloaded specific lists of interest from, e.g., OpenPolitici[9], for the lists of Politicians currently in charge, and also from other data sources, such as DBpedia.
- based on *Human-Knowledge (HK)*: When we could not find a specific list in an open data source, we resorted to a completely manual construction of the dictionary, by exploiting personal knowledge on the topic and web searches. This has been in particular done to construct lists of words that identify certain categories (e.g., containing all phrases used to mean "Lawyer").

The construction of such lists through this approach was non-trivial and required a significant amount of time. We thus investigated alternative methods for Gazetteer lists generation that could reduce the manual effort required, and could be easily replicable in different domains, thus augmenting the generality of our approach. To this aim, we focused on Wikidata as an informative source for Gazetteer lists production.

Wikidata is a collaboratively edited knowledge base, which organizes a large amount of data in a structured way according to a general reference ontology. It is an openly accessible resource, following the Semantic Web standards for exporting, interconnecting and querying data, which can be edited and read by both machine and humans. We have downloaded a Wikidata RDF dump (specifically we used the version of March 3, 2017), and have loaded it on a graph database management system with RDF/SPARQL support, namely Blazegraph[10]. This has enabled us to access Wikidata information

[9] `http://politici.openpolis.it/`
[10] `https://www.blazegraph.com/`

**Table 3.** Annotations through Gazetteer lists based on ODS/HK vs. those based on Wikidata

| List | Entries | Annotations | Spurious Ann. | Missing Ann. | Precision | Recall |
|---|---|---|---|---|---|---|
| First Names ODS | 8913 | 322 | 83 | 0 | 74% | 100% |
| First Names Wikidata | 2517 | 262 | 31 | 5 | 88% | 98% |
| Criminal Organizations HK | 25 | 37 | 21 | 0 | 43% | 100% |
| Criminal Organizations Wikidata | 68 | 20 | 18 | 14 | 10% | 12% |
| Criminal Organizations Wikidata Mod. | 273 | 30 | 24 | 10 | 20% | 37% |
| Politicians ODS | 1083 | 11 | 0 | 40 | 100% | 22% |
| Politicians Wikidata | 7778 | 17 | 0 | 34 | 100% | 33% |
| Journalists ODS | 369 | 4 | 0 | 4 | 100% | 50% |
| Journalists Wikidata | 3727 | 8 | 0 | 0 | 100% | 100% |
| Political Parties ODS | 131 | 23 | 0 | 2 | 100% | 92% |
| Political Parties Wikidata | 447 | 4 | 0 | 21 | 100% | 16% |
| Political Parties Wikidata Mod. | 1293 | 26 | 7 | 6 | 73% | 76% |

through standard SPARQL queries. By virtue of this approach it is possible to avoid tedious and time-consuming development of ad hoc solutions for each wanted Gazetteer list. Indeed, once the setup of the system is completed, a user just needs to define and execute a set of SPARQL queries to obtain the lists of interest. Such queries are specified taking into account the lists to populate, the Wikidata ontology structure, and its data model. The result of each SPARQL query is a CSV file, which has to be simply renamed into a `.lst` file to be processed by the Gazetteer PR.

To compare the two approaches, we considered five lists containing Italian Politicians, Journalists, First Names, Criminal Organizations, and Political Parties, respectively. Each list has been produced and used in two versions, i.e., ODS/HK based and Wikidata based. The comparison has been done on 15 articles and followed the same criteria of the first test described in Section 4: a hand-based annotation aimed at identifying entities in the five chosen lists has been done by a domain expert; then, the articles have been annotated through the Gazetteer PR by using the two different sets of lists generated by the two approaches. The results are shown in Table 3.

From an analysis of the lists created by Wikidata we noticed that they contain various entries with typos and flaws, which was expected to some extent. Our experiments however show that the Precision value for annotations with the Wikidata lists is the same or better than Precision with ODS/HK lists in all cases but the Criminal Organizations one. For several Wikidata lists, also the Recall is close to the one measured for the corresponding ODS/HK list. For the cases of Criminal Organizations and Political Parties we instead initially got very low values for the Recall. This was mainly due to problems with uppercase/lowercase letters (since GATE is case sensitive). We thus refined the two lists by adding for each term the three versions: all uppercase letters, all lowercase letters, and capitalized words. In Table 3 these new lists are denoted as *Wikidata Mod.* We can observe that with this fix the value of Recall become greater than that of the original Wikidata lists, and has reached acceptable levels. However, for the *Political Parties Wikidata Mod.* list this approach has generated a decrease of Precision. This has been caused by the introduction of new entries, e.g., the word *si* (which is the lowercase version of the acronym SI, an Italian Political Party). This word is indeed used in Italian as reflexive pronoun, thus leading to some wrong annotations. Although the

decrease of Precision, we notice that the modification of the Political Parties list increase the Recall of 60%.

From the data present in Table 3, we can conclude that our experiments have been quite successful, since the Wikidata based approach has adequate values in terms of number of annotations, Precision and Recall. Realistically, these values keep unchanged when the number of analysed articles increases. We believe that this confirms that resorting to Wikidata for producing lists to be used by the Gazetteer PR allows for both time savings and good results.

## 6 Conclusions

Future improvements of our approach include the insertion of the PC component discussed in Section 4 in our pipeline. Also, we are working on further refinement of our solution in order to simplify some of the phases that now require a manual intervention, in the spirit of the work done on Gazetteer lists described in Section 5. In particular, we are currently investigating a way to streamline the definition of JAPE rules.

**Acknowledgments.** This work has been partly supported by Leonardo Company in the context of the *XASMOS* initiative, and by the Italian project *RoMA* (SCN_00064). Giulio Ganino has been supported by the FILAS grant *Laboratori teorico-sperimentali a supporto delle applicazioni spaziali delle industrie laziali* (FILAS-RU-2014-1058).

## References

1. N. Antonioli, F. Castanò, S. Coletta, S. Grossi, D. Lembo, M. Lenzerini, A. Poggi, E. Virardi, and P. Castracane. Ontology-based data management for the Italian public debt. In *Proc. of FOIS '14*, pages 372–385, 2014.
2. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2nd edition, 2007.
3. R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537, Nov. 2011.
4. H. Cunningham. *Developing Language Processing Components with GATE Version 8*. University of Sheffield Department of Computer Science, 2014.
5. H. Cunningham, D. Maynard, K. Bontcheva, and V. Tablan. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In *Proc. of ACL'02*, 2002.
6. G. Ganino, D. Lembo, M. Mecella, and F. Scafoglieri. Ontology population for open-source intelligence: a GATE-based solution, 2018. Submitted to an Int. Journal.
7. N. Guarino. Formal ontology in information systems. In *Proc. of FOIS '98*, Frontiers in Artificial Intelligence, pages 3–15. IOS Press, 1998.
8. M. Johnson, S. Khudanpur, M. Ostendorf, and R. Rosenfeld. *Mathematical Foundations of Speech and Language Processing*. Springer New York, 2004.
9. R. Navigli. Word sense disambiguation: a survey. *ACM COMPUTING SURVEYS*, 41(2):1–69, 2009.
10. R. Witte, N. Khamis, and J. Rilling. Flexible ontology population from text: The OwlExporter. In *Proc. of LREC'10*, may 2010.
11. H. Zhao, X. Zhang, and C. Kit. Integrative semantic dependency parsing via efficient large-scale feature selection. *J. of Artificial Intelligence Research*, 46:203–233, 2013.