Louge, Thierry and Karray, Mohamed Hedi and Archimede, Bernard Investigating a method for automatic construction and population of ontologies for services: performances and limitations. (2018) In: 15th ACS/IEEE International Conference on Computer Systems and Applications (AICCSA), 28 October 2018 - 1 November 2018 (Aqaba, Jordan).

Investigating a method for automatic construction and population of ontologies for services: performances and limitations.

Thierry Louge
Université de Toulouse
Institut de Recherche en Astrophysique et Planétologie
Tarbes, France
tlouge@irap.omp.eu

Mohamed Hedi Karray
Université de Toulouse
Ecole Nationale d'Ingénieurs de Tarbes
ENIT
Tarbes, France
mkarray@enit.fr

Bernard Archimède
Université de Toulouse
Ecole Nationale d'Ingénieurs de Tarbes
ENIT
Tarbes, France
Bernard.Archimede@enit.fr

Abstract*— Ontological engineering is a complex process, involving multidisciplinary skills. The Semantic Web, and more specifically Semantic Web Services spreading suffer from the difficulty of producing an ontology sufficiently detailed to be able to correctly describe the data flows exchanged between services. These data are often described using sector-specific vocabulary. Linking these descriptions to external knowledge sources capable of unifying them is often a complex process, requiring adequate sources to be found and properly used. In this paper, we investigate a method combining existing string distance measurement, NLP-analysis and clustering algorithms for automatic construction and population of an ontology. This method takes services capacities descriptions as only input, without external sources of knowledge. It is tested on a set of more than 10,000 services for 106,000 different measures to classify in an ontology, performances and limitations are exposed.

***Keywords*— Ontology, ontology population, unsupervised learning, syntactic matching, semantic Web services**

I. INTRODUCTION

Ontologies, that can be defined through the widely-used Gruber definition like "explicit specification of a conceptualization"[1], play a key role in the Linked Data that consists in "a set of best practices for publishing and connecting structured data on the Web"[2]. The structured data published on the Web form the "Semantic Web"[3], designed for automatic crawling of Web content by algorithms for information retrieval. Service-oriented computing (SOC) may be defined as "the computing paradigm that utilizes services as fundamental elements for developing applications"[4]. A huge number of WS may be accessed and used over the World Wide Web nowadays. As a consequence, several composition approaches and techniques aims a successful integration of WS in services-oriented applications (SOAs)[5], [6]. Those approaches vary in many ways, such as the interaction protocols (Service Oriented Access Protocol SOAP, REST…), data formats and description languages, etc.

Following the principles of the Semantic Web, WS may be semantically annotated through the use of ontologies to improve their interoperability. Such Web Services are known as Semantic Web Services (SWS). The semantic description of SWS may be contained in specific ontologies called "services ontologies". OWL-S[7] and WSMO[8] are the more widely-used ontologies for describing SWS. The semantic description of a SWS provides many information about the service (the description of its inner details such as its methods and its URL called "grounding", pre-conditions and post-conditions, etc.), among which the services inputs and outputs can be found. Unfortunately, providing adequate semantic annotation for a service remains an issue[9]. Description of the capacities of a services may contain domain-specific vocabulary and habits, and the grammatical structure of these descriptions is also far from guaranteed. These specificities are frequently encountered in eScience, e.g.:

Mechanism of action information for FDA-approved drugs[1]

Tanimoto similarity[2]

Are capacities of biological WS, and:

Right Ascension J2000[3]

? 2MASS J total magnitude uncertainty[3]

Are capacities of astrophysical WS. Nevertheless, an ontological representation of those capacities is needed to express the relevant SWS.

Many approaches for automatic ontology construction are discussed in the literature. Automatic and semi-automatic ontology construction and population are based on Information Extraction (IE) techniques, using Natural language Processing (NLP) and comparison with knowledge references. While the majority of approaches successfully use the structure of text and external ontologies to learn a conceptualization from text, they remain poorly efficient in generating a taxonomy from unstructured text or without external source of knowledge.

This is critical for the description of inputs and outputs of services, which we call "service capacities". Web services are often developed for specific applications, within an ecosystem sharing usage and vocabulary habits that are not specified in the description of the services themselves. Information describing inputs and outputs are very often composed of short, unstructured text with domain-specific vocabulary. That makes difficult to find usable matches with general description of the application domain, as provided in

[1] https://www.ebi.ac.uk/chembl/api/data/mechanism

[2] https://www.ebi.ac.uk/chembl/api/data/similarity/CN1C(=O)C=C(c2cccc(Cl)c2)c3cc(ccc13)[C@@](N)(c4ccc(Cl)cc4)c5cncn5C/80

[3] http://vizier.u-strasbg.fr/viz-bin/votable/-A?-out.all&-source=B%2Fpastel%2Fpastel&

most ontologies Nevertheless, defining a SWS from a WS require an ontological description of many elements, including its capacities. Moreover, finding ontologies dedicated to a particular application domain is already a challenge in itself. Lastly, following the linked data latest statistics of 2014, 72.75% of proprietary vocabularies are not "dereferencable" at all[4], meaning that no definition can be found for any term in the requested vocabulary.

In this paper, we propose a method that allows automatic generation of an ontology from short, unstructured text with domain-specific vocabulary. This generation shall not require pre-existing background knowledge, and ensure the automatic population of the ontology from services capacities.This method is based on similarity matrixes clustering and NLP processing for automatic ontology generation and population.

Section II of this paper discusses existing works for ontology generation, and the motivation for proposing a different method specifically for unstructured text with specific vocabulary, without external sources of knowledge. Section III formalises the method itself, which is evaluated in section IV with astrophysical services. Section V concludes this paper, with the future directions and applications of the method that is proposed. The entire code, tests examples and tests results presented in this paper are available for download, reuse and testing at: https://github.com/tlouge/OntologyGeneration.

II. Related work and motivations

A. Ontology construction techniques

Semi-automatic and automatic construction of ontologies is an ongoing topic, combining IE techniques, NLP analysis and ontologies alignment (consisting in finding matching between different ontologies, sometimes up to the merging of different ontologies into a new ontology). This topic is of importance in the SW development and diffusion, because of the central role played by ontologies in the SW. Since it is not a new topic, a survey conducted in 2011[10] presents methods and techniques, that still underlies this research field today: Ontologies alignment and merging, NLP pattern recognition and the use of external sources (frequently WordNet) for words disambiguation. The process of ontology construction consists in extracting an ontological representation of the knowledge contained in a corpus of documents. This corpus may be composed with different sources of data that need to be reified in a single ontology.

In the approach by Touma, Romero and Jovanovic[11], SQL schema and XML documents are transformed into single ontologies (one ontology per document). Single ontologies are then mapped, matched and merge to obtain a final ontology. This approach has the advantage to eliminate the need for external model to act as a target schema. The target schema is constructed on-the-fly, by single ontologies comparison and refinement. This approach has yet to be adapted for text documents.

The ReVerb relationships extractor[12] extracts verb-based relations, assuming that there is a verb in the sentence, and that the overall structure of the sentence is grammatically constructed around the verb. OLLIE[13] extends the extraction to the presence of other elements in the phrases rather than only verbs (nouns, adjectives,…).

BONIE[14] is built on the same principles than OLLIE, and enhances IE for sentences including numerical quantities. While this approach seems promising for IE extraction from eScience-related WS descriptions, it is based on patterns that are not applicable to short, unstructured sentences.

Several initiatives for IE (including OLLIE and BONIE) have been gathered in Open IE5.0[5], and a comprehensive survey of those works has been exposed in[15].

DEFIE[16] is another work towards IE from text. Each sentence of the text corpus is analyzed individually, represented as a dependency graph from which every concept and relation is disambiguated using Babelfy[6] word sense disambiguation. The autors of DEFIE state that the choice of Babelfy is not mandatory, however the approach needs a reliable source of knowledge for disambiguation. DEFIE is presented as a significant improvement over its competitors like ReVerb (it extracts more relations, with better precision) whereas it remains to be compared with OLLIE.

In order to extract features from heterogeneous raw text resources, Vicient et al[17] use Web documents together with the texts and an input ontology. Web documents are retrieved through Web search engines to ensure the relevancy of the concepts extracted from the text and their mapping to the concepts in the ontology. This method, while still relying on external ontologies and relevant Web documents, is automatic, unsupervised and domain-independent. It aims to build a taxonomy, and does not extract relations from the texts.

Another approach for automatic learning of a taxonomy from text is presented in M. Rani et al.[18]. The analyzed corpus of texts should be domain-specific, which corresponds to the analysis of a set of WS capacities sharing the same application domain. The authors present a comparison between two algorithms, Latent Semantic Indexing (LSI) and Latent Dirichlet Association (LDA) through its implementation in MapReduce, MrLDA. The authors state that ontology learning using MrLDA is effective.

Some approaches for ontology construction from text use pattern recognition inside texts to identify concepts, sub-concepts and relations. Those patterns frequently derive from Hearst patterns[19], or may be learnt by the use of machine learning algorithms[20]. In both cases, the assumption is that the sentences grammatical structure helps in finding useful patterns, and that some words in the phrase can give information about the importance of the patterns.

Besides ontology construction, ontology population is another important aspect for SWS. Ontology population encompasses several phases, among which the candidate instances for population are identified, then classified inside the ontology[21]. Techniques used to accomplish those phases are similar to those used for ontology construction (NLP, IE, and some machine learning when a learning set is available).

A recent approach for Web services ontology population from text has been exposed by Reyes-Ortiz et al.[22], oriented towards Web services classification rather than capacities description. The goal of [22] is to obtain an ontology that classifies Web services following their topic (application domain). Features (meaningful terms in the WS descriptions)

[4] http://lod-cloud.net/state/state_2014/#toc7

[5] https://github.com/dair-iitd/OpenIE-standalone

[6] http://babelfy.org/

are extracted using term frequency (TF) and inverse document frequency (IDF) measures. WS are then classified following those features by the means of a machine learning classifier. The description of input/output descriptions is part of the future work planned in the paper.

B. *Motivations*

Constructing an ontology describing the capacities (inputs and outputs) of an ensemble of Web Services is a specific case of constructing an ontology from text. Each service capacity is an information in itself, composed by a sentence that is not necessarily grammatically well-formed and may contain domain-specific vocabulary and habits. Providing an ontology structure for a high number of services is a task that needs automation. In the example we used for our tests concerning astrophysical services, more than 10,000 services were involved, for more than 100,000 information to describe. Obtaining an ontology capable of describing this amount of information without automation is not reasonably possible. On the other hand, finding external sources of knowledge to help build an ontology for very specific topics is difficult. Moreover, the automation of the use of external sources is also an important brake to the automatic construction of ontologies. This is why the method explored in this paper aims at the automatic construction of an ontology without recourse to an external source, for descriptions unstructured grammatically and in large numbers. This method includes ontology population, leading to the ontological description of services capacities inside the ontology built from the analysis of their content.

III. Proposed Method

The method investigated in this paper is based on the grouping of services capabilities inside matrices, whose elements are the value of the similarity measured between each capability present in the matrix. We therefore call these matrices "similarity matrices", they are by construction square, sparse and symmetrical. The content of these matrices is processed by a clustering algorithm, which groups together the most similar capabilities. The analysis of the content of these clusters by NLP procedures reveals patterns that are repeated. Using the syntactical comparison between the terms contained in these patterns, we can deduce the classes and subclasses present in the clusters. The detection of patterns in our method does not rely on any pre-established scheme. In order to handle large amounts of service capabilities, the size of the similarity matrices is part of the parameters of our method, since it can influence the appearance of clusters. The other parameters are the clustering algorithm used, as well as the method used to measure the similarity between the capabilities.

Section A exposes the notations used for describing taxonomy generation. Sections B to E present the details of the method.

A. *Notations*

- Dx, Dy are descriptions of information given by services (capacities).
- $Wn(Dx)$ is the n-th word contained in $Dx, Wp(Dy)$ is the p-th word contained in Dy.
- $|Wn(Dx)|$ is the number of characters composing $Wn(Dx)$, $|Wp(Dy)|$ is the number of characters composing $Wp(Dy)$.
- $D1 \dots Dj$ are bag of words W, $|Di|$ is the number of words in the description Di.
- $S(Dx, Dy)$ is the similarity value between Dx and Dy.
- C is the number of elements in a cluster.
- V is the similarity threshold value.
- $PM$is the pattern multiplier.
- $CM$is the cluster multiplier.
- PO is the number of occurrences of the same pattern inside the same cluster.
- k is the number of clusters detected in a similarity matrix.
- p is the numbers of patterns detected in a cluster.

B. *Clustering similarity matrices from services capacities*

Services capacities are randomly dispatched into groups containing a number of capacities E. A similarity matrix is constructed, which rows and columns refer the capacities in the group. The values of the matrix are the similarity measurements between the capacities, $S(Dx, Dy)$.

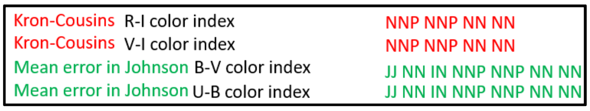
The matrix used to find relevant clusters of information among the descriptions is symmetrical ($S(Dx, Dy) = S(Dy, Dx)$) and composed as follow:

$$\begin{array}{ccccccc}
D_0 & \dots & D_x & \dots & D_y & \dots & D_n \\
\dots & 1 & & & & & \\
D_x & & 1 & & S(Dy, Dx) & & \\
\dots & & & 1 & & & \\
D_y & & S(Dx, Dy) & & 1 & & \\
\dots & & & & & 1 & \\
D_n & & & & & & 1
\end{array}$$

Processing the similarity matrices with a clustering algorithm groups most similar capacities together. Each resulting cluster of capacities is then analyzed, existing patterns detected and a taxonomy emerges. The maximum size of the matrices E may have an influence on the resulting taxonomy, which will be quantified in the experiments in section 4.

C. *Making patterns emerge from the clusters and deriving a taxonomy*

A cluster is composed of capacities. In Fig.1, four capacities are presented. Those capacities are part of the same cluster, and NLP analysis shows up two different patterns for those four capacities. The capacities of a cluster sharing the same pattern are compared, and the same words in the same position indicate a concept candidate.



Kron-Cousins R-I color index	NNP NNP NN NN
Kron-Cousins V-I color index	NNP NNP NN NN
Mean error in Johnson B-V color index	JJ NN IN NNP NNP NN NN
Mean error in Johnson U-B color index	JJ NN IN NNP NNP NN NN

Figure 1: Detection of patterns inside the content of clusters

In Figure , the concept candidates specific for a pattern are “Kron-Cousins” and “Mean error in Johnson” with PO = 2. “Color index” is another concept candidate, detected in both patterns.

A confidence, or belief value is asserted to each concept candidate. For a concept C_1:

$$B_{C_1} = \sum_{0}^{k} \sum_{0}^{p} \frac{PM \;\; \times PO + CM}{C}$$

The sum is done on every cluster, from 0 to k, for every pattern from 0 to p containing the concept C1. If $B_{C_1} \geq V$, then C_1 is elected as a class in the ontology.

The values of PM, CM and V are parameters of the method.

D. *Characterizing the method*

1) *Similarity measurement*

The first step of the method consists in forming the similarity matrices. Several distance measurement exist, among which the Levensthein[23], the Jaro[24] and the Jaro-Winkler[24] distances. A very different way of measuring distances between terms consists in using vector representations of words like proposed in Word2Vec[25] software. Word2Vec trains a model on a corpus of text and returns distance between words based on the model. In the next section, we expose a custom similarity measurement called "StringDist" that combines existing distance measures. StringDist has been used in the method for automatic construction and population of ontologies presented on this paper.

2) *StringDist similarity measurement*

To obtain a similarity measurement between capacities, the first step is to add to the capacity containing the lesser number of words enough "dumb words" to reach the number of words contained in the larger one. The dumb word must not be recognized by the algorithm as a potential matching word between the capacities, for that would false similarity measurement. We used "###" as the dumb word added to the lesser capacity.

The values for the similarity matrix are based on similarity measurements between each pair of words of each pair of capacities, produced as follows:

$Lev\big(Wn(Dx), Wp(Dy)\big)$ is the Levenshtein distance[17] between Wn(Dx) and Wp(Dy).

We define $L\big(Wn(Dx), Wp(Dy)\big)$ as the normalized Levensthein distance between Wn(Dx) and Wp(Dy).

$$L\big(Wn(Dx), Wp(Dy)\big) = 1 - \left(\frac{Lev\big(Wn(Dx), Wp(Dy)\big)}{\max(|Wn(Dx)|, |Wp(Dy)|)}\right)$$

$J(Wn(Dx), Wp(Dy))$ and $JW(Wn(Dx), Wp(Dy))$ are the Jaro and Jaro-Winkler distance between Wn(Dx) and Wp(Dy), respectively. The similarity value $S(Wn(Dx), Wp(Dy))$ is then:

$$S\big(Wn(Dx), Wp(Dy)\big) = \max\big(\, L\big(Wn(Dx), Wp(Dy)\big)\,, J\big(Wn(Dx), Wp(Dy)\big)\,, JW\big(Wn(Dx), Wp(Dy)\big)\big)$$

We define then the trust value $T(Wn(Dx))$ as the maximum matching score for a single word Wn in Dx with every word from Dy, considering the normalized Levensthein distance, Jaro and Jaro-Winkler:

$$T(Wn(Dx)) = \max(S\big(Wn(Dx), Wp(Dy)\big);\ p = 0\,..\,|Dy|$$

And finally, the similarity value S(Dx,Dy) (called StringDist in the rest of this paper) is the sum of every trust values from words in Dx divided by the number of words |Dx| (and |Dx| = |Dy|, as we filled the lesser description with dumb word instances).

$$S(Dx, Dy) = \frac{\sum_{n=0}^{n=|Dx|} T\big(Wn(Dx)\big)}{|Dx|}$$

The method we propose is based on the analysis of similarity matrices, obtained through the measurement of similarities between descriptions of services capacities. The algorithm used for determining those similarities is a hyper-parameter for the method. We tested the method with Levensthein distance, Jaro-Winkler distance, Word2Vec and StringDist. The experiments are described in section IV.

A clustering algorithm is applied on the matrixes to obtain the clusters of capacities inside which the patterns will be detected, and the taxonomy extracted. The algorithm used for this clustering is another hyper-parameter. This clustering algorithm must not need the number of clusters as an input a-priori ; as the goal of the method is to discover a structure from scratch, only using the capacities of services.

E. *Populating the taxonomy with services capacities*

Populating the ontology consists in assigning individuals inside the ontology to capacities of the services. The same similarity measure used for ontology construction (Jaro-Winkler, StringDist…) is also used for ontology population. Each service capacity is compared to the annotation of each of the classes extracted from the capacities corpus. An individual is created inside the ontology for the concerned capacity, under the best matching class providing that the best matching score be superior to the parameter V.

IV. EXPERIMENTS AND RESULTS

A great number of tests have been conducted, in order to measure the influence of the PM, CM, V parameters on the results of the method. The experiments exposed hereafter use PM = 0.7, CM = 0.3 and V = 0.7. We conducted our tests on a total of 106000 capacities automatically extracted from real-world astrophysical services (comporting 55513 different capacities, out of the total of 106000). The clustering algorithms used for the tests are "Affinity Propagation"[26], "DBSCAN"[27] and "Mean Shift"[28]. The similarity measure come from Jaro-Winkler, Levensthein, Word2Vec or StringDist. A first set of tests quantified the influence of matrix size, clustering algorithm and similarity measurement method on the resulting taxonomies. Those tests have been run on a random subset of 4000 capacities from the total of 106000 capacities available. DBSCAN and Mean Shift produced no relevant extraction of concepts, while the affinity propagation produced clusters allowing the extraction of relevant concepts from the corpus. Fig. 2 to Fig. 4 present the results of the affinity propagation clustering algorithm with different similarity measures, following different matrices size.

Using StringDist and Levensthein similarity measurements performs best on those tests, as they lead to the extraction of more classes and sub-classes than using Jaro-Winkler and Word2Vec. Results for Word2Vec lead to no extraction of any class. Fig. 4 exposes the number of individuals in the ontology after its population, following matrices size and similarity measurement method.

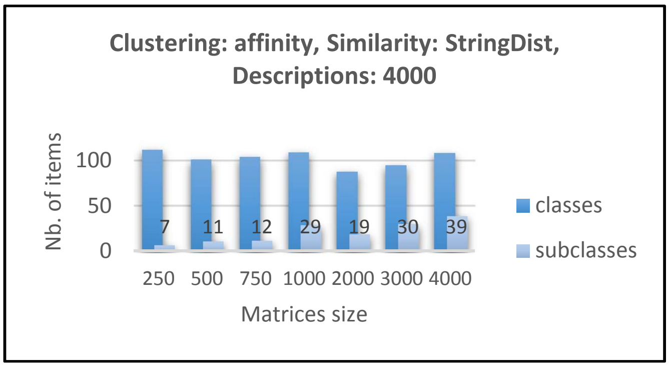


Figure 2: Extracted classes and subclasses for 4000 descriptions

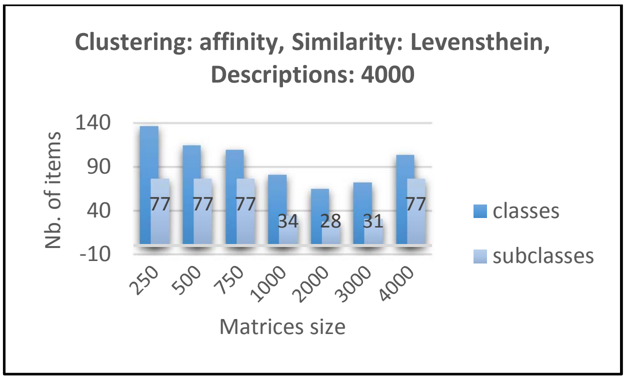


Figure 3: Extracted classes and subclasses for 4000 descriptions

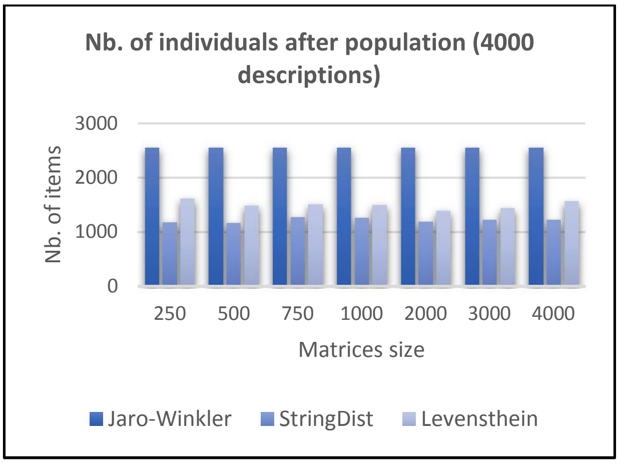


Figure 4: Number of individuals following the matrices size

While Jaro-Winkler similarity measurement method ended in no usable taxonomy generation, both Levensthein and StringDist method produced usable taxonomies. The behavior of the method using Levensthein is less stable than using StringDist. The latter provides more classes and sub-classes as matrices size increase, whereas the first shows a drop off in extraction for matrices sizes of 1000, 2000 and 3000 and finally goes to a comparable performance for 4000 and 750. Populating the ontology with relevant individuals for services capacities is the last step of the method. Figure 4 shows that Jaro-Winkler method populates the ontology with every available individual (2552 different individuals, out of the 4000 descriptions used). That means that Jaro-Winkler is not well suited for the method we are investigating, as it does not filter the individuals according to the most reliable class during population. Using Levensthein and StringDist, on the other side, provide stable population regarless of the matrices size. This leads to a more detailed taxonomy as the matrices size increase (more classes and sub-classes for roughly the same number of individuals). When we look at the results in Figures 4 and 5, we see that this is truer when the StringDist method is used.

The second phase of testing consisted in applying the entire method on 106,000 descriptions with affinity propagation clustering algorithm, StringDist and Levensthein similarity measurement. The results are presented in Fig. 5 and Fig. 6.

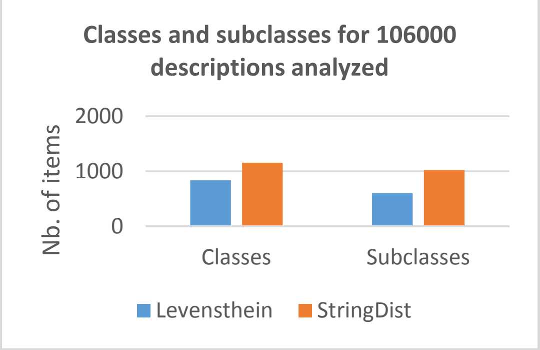


Figure 5: Classes and subclasses for 106000 descriptions

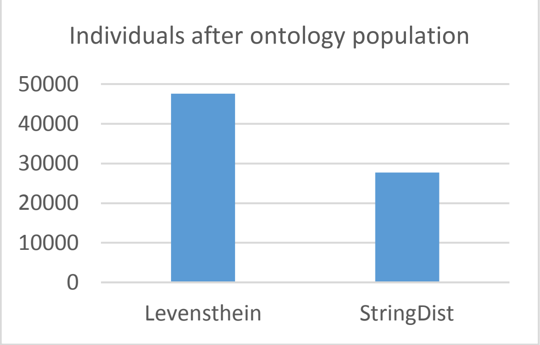


Figure 6: Individuals after ontology population for 106000 descriptions

StringDist similarity measurement produces more classes and sub-classes than Levensthein, but provides a lower number of individuals.

Combining affinity propagation algorithm with StringDist similarity measurement gives the best results when experiments are conducted on the 106,000 descriptions of services. Both the number of classes and sub-classes are the highest, meaning that the knowledge extracted from the corpus is not expressed in a flat, but on the contrary in a taxonomy with several levels of details. Nevertheless, a step of manual validation following the automatic extraction and population of the ontology remains necessary. Regardless of the similarity measure, the clustering algorithm and the matrices size, there is still a need to eliminate irrelevant concepts and individuals attached to unsuitable concepts. Limitations of this method are discussed in the conclusion of this paper.

V. Discussion and conclusion

The method that we are investigating in this paper does create an ontology from services descriptions without external source of knowledge, for unstructured text. Unfortunately, there is still a lot of noise in the resulting taxonomy and room for improvement. Constructing an ontology for describing more than 10,000 services would induce a hardy manageable amount of work. Such an amount of services is found in astrophysics, and probably in other scientific or industrial fields. Many companies or organizations may have developed services during years before considering using semantic description for interoperability and standardization.

We presented in this paper a method for automatic ontology extraction and population from unstructured text, with domain-specific vocabulary without using external resources of knowledge. We argue that such a method may greatly help the development of Semantic Web Services, by giving a relevant structure for the description of services outputs and outputs. The method knows several parameters such as the clustering algorithm, the similarity measurement method and the parameters PM, CM and V. Automating the choice of those parameters PM, CM, V and determining the best combination of clustering algorithm and similarity measurement would certainly improve the performance of the method, under various conditions of use. Such automation would require systematic testing, measuring the number of concepts and sub-concepts derived for various combinations of parameters.

REFERENCES

[1] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowl. Acquis.*, vol. 5, no. 2, pp. 199–220, 1993.

[2] C. Bizer and F. U. Berlin, "Linked Data - The Story So Far," 2009.

[3] T. Berneers-Lee, J. Hendler, and O. Lassila, "The Semantic Web," *Sci. Am.*, vol. 284, no. 5, pp. 34–43, 2001.

[4] M. P. Papazoglou, "Service -oriented computing: Concepts, characteristics and directions," *Proc. - 4th Int. Conf. Web Inf. Syst. Eng. WISE 2003*, pp. 3–12, 2003.

[5] A. L. Lemos, F. Daniel, and B. Benatallah, "Web service composition: a survey of techniques and tools," *ACM Comput. Surv.*, vol. 48, no. 3, p. 33, 2016.

[6] Q. Z. Sheng, X. Qiao, A. V. Vasilakos, C. Szabo, S. Bourne, and X. Xu, "Web services composition: A decade's overview," *Inf. Sci. (Ny).*, vol. 280, pp. 218–238, 2014.

[7] D. Martin *et al.*, "Bringing semantics to web services with OWL-S," *World Wide Web*, vol. 10, no. 3Martin, D. et 2007. Bringing semantics to web services with OWL-S. World Wide Web, 10(3), p.243–277., pp. 243–277, 2007.

[8] R. Dumitru *et al.*, "Web Service Modeling Ontology," *Appl. Ontol.*, vol. 1, no. 1, pp. 77–106, 2005.

[9] J.-Y. K. Yong-Yi Fanjiang, Yang Syu, Shang-Pin Ma, "An overview and classification of services description approaches in automated services composition," *IEEE Trans. Serv. Comput.*, vol. 10, no. 2, pp. 176–189, 2017.

[10] G. Petasis, V. Karkaletsis, G. Paliouras, A. Krithara, and E. Zavitsanos, "Ontology population and enrichment: State of the art," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6050, pp. 134–166, 2011.

[11] R. Touma, O. Romero, and P. Jovanovic, "Supporting Data Integration Tasks with Semi-Automatic Ontology Construction," *Proc. ACM Eighteenth Int. Work. Data Warehous. Ol. - Dol. '15*, pp. 89–98, 2015.

[12] A. Fader, S. Soderland, and O. Etzioni, "Identifying relations for open information extraction," *Proc. Conf. …*, pp. 1535–1545, 2011.

[13] Mausam, M. Schmitz, S. Soderland, R. Bart, and O. Etzioni, "Open Language Learning for Information Extraction," *EMNLP-CoNLL*, no. July, pp. 523–534, 2012.

[14] S. Saha, H. Pal, and Mausam, "Bootstrapping for Numerical Open IE," *Proc. 55th Annu. Meet. Assoc. Comput. Linguist. (Volume 2 Short Pap.*, pp. 317–323, 2017.

[15] Mausam, "Open Information Extraction Systems and Downstream Applications," pp. 4074–4077, 2016.

[16] C. Delli Bovi, L. Telesca, and R. Navigli, "Large-Scale Information Extraction from Textual Definitions through Deep Syntactic and Semantic Analysis," *Trans. Assoc. Comput. Linguist.*, vol. 3, pp. 529–543, 2015.

[17] C. Vicient, D. Sánchez, and A. Moreno, "An automatic approach for ontology-based feature extraction from heterogeneous textualresources," *Eng. Appl. Artif. Intell.*, vol. 26, no. 3, pp. 1092–1106, 2013.

[18] M. Rani, A. K. Dhar, and O. P. Vyas, "Semi-automatic terminology ontology learning based on topic modeling," *Eng. Appl. Artif. Intell.*, vol. 63, pp. 108–125, 2017.

[19] M. A. Hearst, "Automatic acquisition of hyponyms from large text corpora," *Proc. 14th Conf. Comput. Linguist.* -, vol. 2, p. 539, 1992.

[20] R. Snow, D. Jurafsky, and A. Y. Ng, "Learning syntactic patterns for automatic hypernym discovery," *Adv. Neural Inf. Process. Syst. 17*, vol. 17, pp. 1297–1304, 2004.

[21] C. Faria, R. Girardi, and P. Novais, "Analysing the problem and main approaches for ontology population," *Proc. 2013 10th Int. Conf. Inf. Technol. New Gener. ITNG 2013*, no. 1, pp. 613–618, 2013.

[22] J. A. Reyes-ortiz, M. Bravo, and H. Pablo, "Web Services Ontology Population through Text Classification," *2016 Fed. Conf. Comput. Sci. Inf. Syst.*, vol. 8, pp. 491–495, 2016.

[23] Levenshtein, Vladimir I. "Binary codes capable of correcting deletions, insertions, and reversals". Soviet Physics Doklady. 10 (8): 707–710, 1966.

[24] W. E. Winkler, "The State of Record Linkage and Current Research Problems," p. 15, 1999.

[25] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space," pp. 1–12, 2013.

[26] B. J. Frey and D. Dueck, "Clustering by passing messages between data points.," *Science*, vol. 315, no. 5814, pp. 972–976, 2007.

[27] M. Ester, X. Xu, H. Kriegel, and J. Sander, "Density-based algorithm for discovering clusters in large spatial databases with noise," *Proc. Acm Sigkdd Int. Conf. Knowl. Discov. Data Min.*, vol. pages, pp. 226–231, 1996.

[28] D. Comaniciu and et al., "Mean shift: a robust approach toward feature space analysis," *Pattern Anal. Mach. Intell. IEEE Trans.*, vol. 24, no. 5, pp. 603–619, 2002.