Towards Automatic Domain Classification of LOV Vocabularies

Alexis Pister, Ghislain Atemezing

MONDECA, 35 boulevard de Strasbourg 75010 Paris, France.
<firstname.lastname@mondeca.com>

Abstract. Assigning a topic or a domain to a vocabulary in a catalog is not always a trivial task. Fortunately, ontology experts can use their previous experience to easily achieve this task. In the case of Linked Open Vocabularies (LOV), a few number of curators (only 4 people) and the high number of submissions lead to find automatic solutions to suggest to curators a domain in which to attach a newly submitted vocabulary. This paper proposes a machine learning approach to automatically classify new submitted vocabularies into LOV using statistical models which take any texts description found in a vocabulary. The results show that the Support Vector Machine (SVM) model gives the best micro F1-score of 0.36. An evaluation with twelve vocabularies used for testing the classifier shades light for a possible integration of the results to assist curators in assigning domains to vocabularies in the future.

Keywords: Ontologies, Classification, Machine Learning, Linked Open Vocabularies

1 Introduction

Linked Open Data (LOD) refers to the ecosystem of all the open source structured data which follows the standard web technologies such as RDF, URIs and HTTP. As the number of available data grows with time, new datasets following these principles appear. Linked Open Vocabulary (LOV) [1] is an initiative which aims to reference all the available vocabularies published on the Web following best practices guided by the FAIR (Findable - Accessible - Interoperable - Reproducible) principles. Each vocabulary can be seen as a knowledge graph, describing the properties and the purpose of the vocabulary, and which can be connected to other vocabularies by different types of links. Therefore, LOV can be seen as a knowledge graph of interlinked vocabularies [16] accessible on the Web of data.

When a new ontology is submitted for integration into LOV, a curator needs to assign at least one tag representing a domain or a category to the vocabulary

Copyright © 2019 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

[1] https://lov.linkeddata.es/dataset/lov/

CEUR-WS.org/Vol-2465/profiles_paper3.pdf

among existing 43 categories, such as “Environment”, “Music” or “W3C REC”. A category aims at grouping ontologies according to a domain. For example, the tag “W3C REC” represents ontologies recommended by the W3 Consortium, such as `rdf` or `owl`. As the number of domains increases and some vocabularies[2] can be relatively small, the tagging process can be biased. Figure. 1 depicts the list of the tags available in LOV as the time of writing this paper, while Figure 2 depicts their distribution. One of the benefit of assigning a tag to a vocabulary is to index it according to a domain and make it easy to access from the interface. For example, to access to vocabularies in the IOT domain, the direct URL in LOV is `https://lov.linkeddata.es/dataset/lov/vocabs?tag=IoT`. Additionally, any newly added vocabulary should belong to at least one domain.

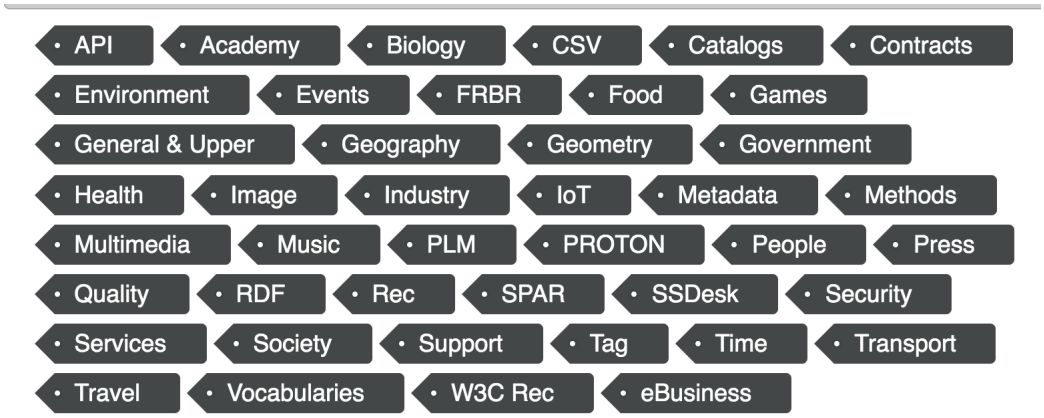

Fig. 1: A view of the list of the tags available in LOV backend used for classifying ontologies

We propose a machine learning approach to automatically classify newly submitted vocabularies with statistical models which take texts describing the subjects of the vocabularies as input. Indeed, the majority of the graphs contains a lot of text describing the subjects and the properties of the vocabularies, in the form of string literals. For example, the URI in a given ontology (Class or Property) is often described by the predicate `rdfs:comment` with a text mentioning the comment of a given resource. Other predicates are often linked to texts containing information, such as `rdfs:label` or `dct:description`. We used all this text information to train several machine learning models in the purpose of classifying the vocabularies into different categories. This paper is structured as follows: Section 2 describes related work in graph classification, followed by the

[2] In this paper, the terms ontology and vocabulary are interchangeable

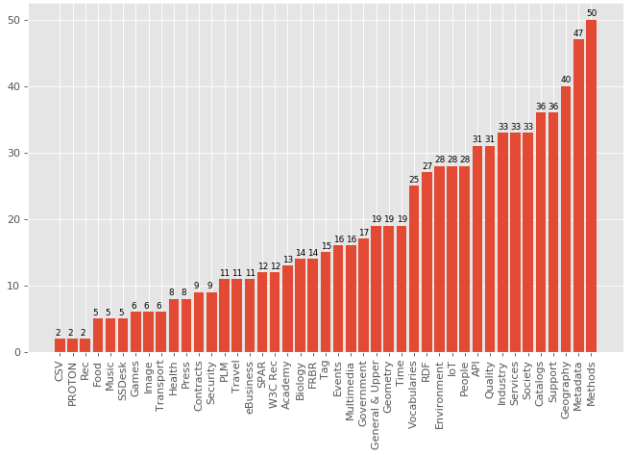

Fig. 2: Distribution of the tags among the vocabularies in LOV

machine learning approach to build the classifier in Section 3. Section 5 provides an evaluation of our approach and a brief conclusion in Section 6

2 Related Work

Graph classification is a problem well studied in the literature. Several strategies have been developed to tackle this problem such as kernel methods or more recently graph neural networks [13]. However, there is way less work made in knowledge graph classification. What comes closest are the entity or triples classification problems which consist in the categorization of a really small subset of a knowledge graph [17]. It is because these types of graphs are mainly described by their entities and relations, so it would be very difficult to find similarities or dissimilarities between knowledge graphs which share very little or not a common entity or relation, like it is often the case. This is why we used a different approach of traditional graph classification methods for our problem, using a text mining strategy. Indeed, a lot of work has been made in document classification [1]. Various processing methods have been elaborated such as Bag of Words or Latent Semantic Analysis (LSA), whose input can be easily exploited by machine learning algorithms.

Classifying datasets created using semantic technologies has been applied in the literature. The most closest work in the literature is described in [7] and [15]. Meusel et al. present a methodology to automatically classify LOD datasets based on the different categories presented in the LOD cloud diagram. The paper uses eight feature sets from the LOD datasets, among others are text from `rdfs:label`. One of the main conclusions of the paper is that vocabulary-level features are good indicator for the topical domain.

While the mentioned approach uses supervised learning, we apply two more steps in preparing the corpus for input of the classifier, using Bag-of-Word and a Truncated SVD transformation. Additionally, we have a very small amount of corpus inherent to the size of vocabularies compared to the entire LOD datasets, and a higher number of available tags (43 in LOV compared to 8 for the LOD cloud).

3 Data Preparation and Machine Learning Models

3.1 Data Preparation

Our approach has been to use the texts contained in the vocabularies to classify them into categories. Indeed, usually the subject of a RDF graph and the purpose of its entities are described in string literals following some specific predicates. We first extract this relevant textual information (string or literal) inside each graph (a dump representing the latest version of the vocabulary in N3), and concatenate it into one paragraph describing their subjects. To this end, we first download each recent version of the vocabulary from LOV SPARQL endpoint (taking the most recent version tracked by LOV) and import them into graph objects with RDFLib[3]. Listing 1.1 depicts the SPARQL query used to retrieve the latest version of each vocabulary, alongside with their domains and unique prefix.

```
SELECT DISTINCT  ?vocabPrefix ?domain ?versionUri  {
        GRAPH <https://lov.linkeddata.es/dataset/lov>{
        ?vocab a voaf:Vocabulary.
        ?vocab vann:preferredNamespacePrefix ?vocabPrefix.
        ?vocab dcterms:modified ?modified.
    ?vocab dcat:keyword ?domain.
    ?vocab dcat:distribution ?versionUri.
    BIND ( STRAFTER(STR(?versionUri), "/versions/") as ?v)
    BIND(STRBEFORE(STR(?v), ".") as ?v1)
    BIND (STR(?modified) as ?date )
    FILTER ( ?date = ?v1)
  }} GROUP BY ?vocabPrefix ?domain ?versionUri
ORDER BY ?vocabPrefix ?domain ?versionUri
```

Listing 1.1: SPARQL query to retrieve the latest versions of vocabularies stored in LOV

We then concatenate all the strings followed by the predicates having one of these suffixes : `comment`, `description`, `label` and `definition`. The predicate `rdfs:label` is often used to give a name of an URI in natural language, while the suffixes `comment`, `description` and `definition` are used to give insight on the meaning and purpose of a given ontology or entity. The result of this step has been the generation of a paragraph for each vocabulary. As the texts describe the

[3] `https://github.com/RDFLib/rdflib`

RDF properties of the graphs, they often contain the suffixes of these properties formed of several words not separated by spaces, in camel case format. For example, if an extracted text mentions the property "UnitPriceSpecification", this expression will remains as a single unit in the final text. However, it can imply a bias on the statistical model to be applied on this data. Consequently, we separate all these types of expression with spaces, when a uppercase occurs in the middle of a word. Therefore, by using this method, the expression "UnitPriceSpecification" will be transformed to "Unit Price Specification" in the final text. After this transformation, the whole corpus' vocabulary is formed of 21,435 different words. The mean word count for the paragraphs is 1168.5, the maximum is 86208 and the minimum 0. Two paragraphs were empty and 25 of them have less than 20 words. The text describing the rooms vocabulary [4] obtained with the pre-processing step described in this section is presented in Listing 1.2. This ontology describes the rooms one can find in a building and has the following assigned tags in LOV: Geography and Environment.

Floor Section. Contains. Desk. Building. Floor. A space inside a structure, typically separated from the outside by exterior walls and from other rooms in the same structure by internal walls. A human-made structure used for sheltering or continuous occupancy. Site. A simple vocabulary for describing the rooms in a building. An agent that generally occupies the physical area of the subject resource. Having this property implies being a spatial object. Being the object of this property implies being an agent. Intended for use with buildings, rooms, desks, etc. Room. The object resource is physically and spatially contained in the subject resource. Being the subject or object of this property implies being a spatial object. Intended for use in the context of buildings, rooms, etc. A table used in a work or office setting, typically for reading, writing, or computer use. A named part of a floor of a building. Typically used to denote several rooms that are grouped together based on spatial arrangement or use. A level part of a building that has a permanent roof. A storey of a building. Occupant. An area of land with a designated purpose, such as a university Campus, a housing estate, or a building site.

Listing 1.2: Paragraph describing the rooms vocabulary, obtained with the preprocessing pipeline described in Section 3.

[4] `https://lov.linkeddata.es/dataset/lov/vocabs/rooms`

3.2 Machine Learning Models

As we cannot feed directly text paragraphs to the machine learning models, we applied a processing pipeline for transforming the texts into fixed-size vectors of attributes. For this purpose, we used several techniques described in [14] : we first apply a Bag-of-Words (BoW) transformation, mapping the texts to vectors containing the frequencies of each word and ngram made of 2 and 3 words in the documents which have a frequency value between 0.025 and 0.25. Then, a Term Frequency-Inverse Document Frequency (TF-IDF) is applied to normalize the frequencies of the words and ngrams by the length of each document. Finally, we apply a Latent Semantic Analysis (LSA) [3] which is a dimensionality reduction technique using a linear algebra method called truncated SVD, to map the space of word frequencies to a smaller space of concepts. Indeed, the dimension of the TF-IDF vectors is big, as it corresponds to the number of words used in the whole corpus plus the frequent ngrams (21,435). It is well-known in the literature that a high number of attributes often impact negatively a machine learning approach [2]. We tried different values of n representing the dimension of the vector space : 50, 150 and 300. These vectors of attributes are then used as input for the machine learning classifiers. The entire processing pipeline is summarized in Figure 3.

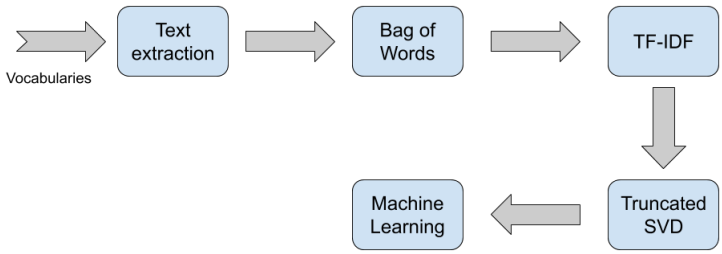

Fig. 3: Schematic view of the processing pipeline. From left to right, the diagram depicts the different steps: 1-Text extraction from Vocabulary dump; 2-BoW Transformation; 3-Normalization with TF-IDF; 4-Vector dimension reduction and finally the classifiers.

We then separated the data in two subsets composing of a training set (80% of the vocabularies) and a test set (the remaining 20%). In this paper, the dataset version of LOV used for the experiment is the snapshot as of May 7th, 2019 [5], containing 666 vocabularies. We claim that the approach described in this paper can be replicated to any type of machine learning multi-label task with a knowledge graph as input.

[5] `https://tinyurl.com/lovdataset`

As each vocabulary can have one to many tags, we tackle the problem as a multi-label classification task. A machine learning model is trained on the training set, trying to find relation between the attributes describing the graphs and their labels. The trained model is then applied to the test set. The predicted labels are finally compared to the ones tagged by human curators, and the micro precision, recall and f1-measure are computed, which are current supervised learning metrics [11]. We have tested several machine learning models with the python library scikit-learn [10], with an emphasis on the Support Vector Machine (SVM) and the Multi Layer Perceptron (MLP) which are ranked among the best classifiers for text classification task, mainly because they can handle large feature spaces [4, 12]. The K-Nearest-Neighbors (KNN) and the Random Forest (RF) classifiers have been tested as well, because they natively support multi-label classification, as well as the MLP.

However, we had to apply a One-vs-Rest strategy for the SVM [9], which consists in training a separate binary classifier for each label. The MLP had one hidden layer of size 100 with a Rectified Linear Unit (ReLU)[6] activation function. Similarly, we set the parameters $C = 10$, $gamma = 1$ for the SVM, with a radial basis function kernel (RBF kernel)[7] and weighting the classes uniformly. We chose $k = 7$ for the KNN model.

4 Results

The results of the classification for the 4 machine learning models, using $k = 50, 150, 300$ for the truncated SVD are presented in Table 1. The MLP and the SVM give the best micro F1-score respectively of 0.34 and 0.36, with $n = 150$.

	n = 50			n = 150			n = 300		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
SVM	0.22	**0.50**	0.31	0.39	0.32	**0.36**	0.47	0.23	0.31
RF	**0.74**	0.07	0.12	0.7	0.03	0.07	0.68	0.02	0.04
MLP	0.33	0.32	0.33	0.34	0.33	0.34	0.33	0.25	0.29
KNN	0.62	0.10	0.17	0.65	0.06	0.11	0.58	0.06	0.11

Table 1: Results of the classification on the test set for the 4 machine learning algorithms, with 3 values of the dimension of the feature space.

5 Evaluation and Discussion

In this section, we describe the evaluation of the classifier on newly submitted ontologies in LOV, and we discuss the results obtained comparing with manual assignment by two curators.

[6] The ReLU is the most used activation function in neural network. f(z) is zero when z is less than zero and f(z) is equal to z when z is above or equal to zero.

[7] `https://en.wikipedia.org/wiki/Radial_basis_function_kernel`

5.1 Evaluation

For evaluating our model, we took a list of 12 vocabularies in the back-end of LOV and asked two curators to assign domains to each of the vocabulary. Then, we passed the same vocabularies to the SVM classifier. The classifier's results is then compared with the human assignment tags as presented in Table 2.

Table 2: Comparison of tags suggested by the classifier and the curator. The underlined tags are the perfect match by both the human and the SVM classifier.

Vocabulary URI	Curator tag(s)	Classifier's tag(s)
`https://w3id.org/vir`	Multimedia	Support
`https://w3id.org/usability`	Support, Events	API
`https://www.w3.org/ns/solid/terms#`	Services, General & Upper	Services, General & Upper, RDF
`http://ns.inria.fr/munc/v2#`	Metadata	RDF
`https://w3id.org/arco/ontology/core`	Services, Society	Catalogs, Events, Government, Multimedia
`https://w3id.org/arco/ontology/catalogue`	Catalogs, society	Catalogs, Events, Government, Multimedia
`https://w3id.org/arco/ontology/context-description`	Support, General & Upper	Catalogs, Environment, Events, Government, Multimedia
`https://w3id.org/arco/ontology/denotative-description`	Support, General & Upper	Catalogs, Environement, Events, Government, Multimedia
`https://w3id.org/arco/ontology/cultural-event`	Events, society	Events, Catalogs, Goverement, Multimedia
`https://w3id.org/arco/ontology/location`	Geography, Geometry	Catalogs, Events, Government, Multimedia
`https://w3id.org/arco/ontology/arco`	General & Upper	Catalogs, Environment, Events, Government, Multimedia
`https://w3id.org/cocoon/v1.0`	Services, Contracts	Industry, Services

As the main goal of the system is to suggest recommendation to a curator, we compute a soft accuracy metric, corresponding to the number of graph with at least one match between one of the curator tags and the classifier suggestions, divided by the total number of tested vocabularies.

For a vocabulary i, its associated tags $y_i = \{y_{i1}, y_{i2}, ..., y_{il}\}$ and the prediction of the classifier $y_i^{pred} = \left\{y_{i1}^{pred}, y_{i2}^{pred}, ..., y_{im}^{pred}\right\}$, we say that the classifier is *softly accurate* for the vocabulary i if $\exists y_{ik}^{pred} \in y_i^{pred}$ such that $y_{ik}^{pred} \in y_i$. The soft accuracy is then computed by the ratio of the number p of outputs softly accurate on the total number n of inputs. We get a result of 0.33 for this evaluation.

5.2 Discussion

The results seem average regarding the precision in the detection from the classifier, compared to the curator. Their could be several explanations, like the disparity between the tags in the dataset (13 labels are used in less than 10 vocabularies), or the difference of subjects in vocabularies tagged by the same label. For example, the "geography" tag is used for the `rooms` and the `Postcode` [8] ontologies, whereas they both describe completely different things, thus we can expect different words usage and very different feature vectors.

Furthermore, multi-label classification for tagging recommendation is a hard task, especially when the number of possible tag is high (43) and the number of examples is low (666) [5] like in this particular setting . It has been demonstrated that SVM classifiers work well for text classification problem, however their performance decrease strongly as the number of labels increases [6]. The list of domains grows depending on the need and some have a more organizational function. For example, LOV curators introduced the IOT tag to group all the vocabularies related to the IoT domain. Historically, some of the tags are related to W3C vocabularies recommendations (W3C Rec).

6 Conclusion and Future Work

This paper addresses one main issue: build and evaluate a classifier based on the content of LOV catalog using machine learning technique. The final goal of this work is to help the human curator of vocabularies to have a list of recommendations for a new ontology submitted in the back-end. The classifier implemented gives a micro F1-score of 36%. Although this score seems low, the system will not be used without a human that validates or not the suggested tag. We do not intend to compare the system with the human curator. Instead, we want to have a system that reduce possible risk of bias when assigning domains to vocabularies and suggest tags to the curator. Future work includes ingesting the feedback from the curators into the classifier to learn from newly added vocabularies for a continuous learning workflow, and test deep learning models with a transfer learning strategy to overcome the low-frequency of training examples.

[8] `https://lov.linkeddata.es/dataset/lov/vocabs/postcode`

Indeed, deep learning approach can perform well on multi-label classification, but it needs a lot of training examples [8].

References

1. M. Allahyari, S. Pouriyeh, M. Assefi, S. Safaei, E. D. Trippe, J. B. Gutierrez, and K. Kochut. A brief survey of text mining: Classification, clustering and extraction techniques. *arXiv preprint arXiv:1707.02919*, 2017.
2. P. F. Evangelista, M. J. Embrechts, and B. K. Szymanski. Taming the curse of dimensionality in kernels and novelty detection. In *Applied soft computing technologies: The challenge of complexity*, pages 425–438. Springer, 2006.
3. N. Halko, P.-G. Martinsson, and J. A. Tropp. Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, 53(2):217–288, 2011.
4. T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *European conference on machine learning*, pages 137–142. Springer, 1998.
5. I. Katakis, G. Tsoumakas, and I. Vlahavas. Multilabel text classification for automated tag suggestion. In *Proceedings of the ECML/PKDD*, volume 18, 2008.
6. T.-Y. Liu, Y. Yang, H. Wan, H.-J. Zeng, Z. Chen, and W.-Y. Ma. Support vector machines classification with a very large-scale taxonomy. *Acm Sigkdd Explorations Newsletter*, 7(1):36–43, 2005.
7. R. Meusel, B. Spahiu, C. Bizer, and H. Paulheim. Towards automatic topical classification of lod datasets. In *Workshop on Linked Data on the Web, LDOW-co-located with the 24th International World Wide Web Conference, WWW 19 may*, volume 1409, 2015.
8. J. Nam, J. Kim, E. L. Mencía, I. Gurevych, and J. Fürnkranz. Large-scale multi-label text classification—revisiting neural networks. In *Joint european conference on machine learning and knowledge discovery in databases*, pages 437–452. Springer, 2014.
9. M.-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729. IEEE, 2008.
10. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.
11. D. M. Powers. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. 2011.
12. M. E. Ruiz and P. Srinivasan. Automatic text categorization using neural networks. In *Proceedings of the 8th ASIS SIG/CR Workshop on Classification Research*, pages 59–72, 1998.
13. F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2008.
14. F. Sebastiani. Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47, 2002.
15. B. Spahiu, A. Maurino, and R. Meusel. Topic profiling benchmarks in the linked open data cloud: Issues and lessons learned. *Semantic Web*, (Preprint):1–20, 2019.

16. P.-Y. Vandenbussche, G. A. Atemezing, M. Poveda-Villalón, and B. Vatant. Linked open vocabularies (lov): a gateway to reusable semantic vocabularies on the web. *Semantic Web*, 8(3):437–452, 2017.
17. Z. Wang, J. Zhang, J. Feng, and Z. Chen. Knowledge graph embedding by translating on hyperplanes. In *Twenty-Eighth AAAI conference on artificial intelligence*, 2014.