A Methodology for Extracting Knowledge about Controlled Vocabularies from Textual Data using FCA-Based Ontology Engineering

1st Simin Jabbari
Information Management Institute
University of Neuchâtel
Neuchâtel 2000, Switzerland

Diagnostics Data Science Lab.
F. Hoffmann La Roche Ltd.
Basel 4070, Switzerland
simin.jabbari@unine.ch

2nd Kilian Stoffel
Information Management Institute
University of Neuchâtel
Neuchâtel 2000, Switzerland
kilian.stoffel@unine.ch

Abstract—We introduce an end-to-end methodology (from text processing to querying a knowledge graph) for the sake of knowledge extraction from text corpora with a focus on a list of vocabularies of interest. We propose a pipeline that incorporates Natural Language Processing (NLP), Formal Concept Analysis (FCA), and Ontology Engineering techniques to build an ontology from textual data. We then extract the knowledge about controlled vocabularies by querying that knowledge graph, i.e., the engineered ontology. We demonstrate the significance of the proposed methodology by using it for knowledge extraction from a text corpus that consists of 800 news articles and reports about companies and products in the IT and pharmaceutical domain, where the focus is on a given list of 250 controlled vocabularies.

Index Terms—Semantic knowledge extraction, Ontology learning, Controlled vocabulary, Formal Concept Analysis, Natural Language Processing.

I. INTRODUCTION

Knowledge extraction from textual data especially about a set of controlled vocabularies has been recognized as one of the important elements in semantic data analytics. It has variety of applications which includes building chatbots and smart search engines. For such a purpose, building a knowledge representation model is a must. Although ontologies have been introduced as promising tools for representing knowledge in a formal way [1]–[3], building ontologies from text corpora seems to be a difficult task. This is mainly due to difficulty of giving structure to unstructured data (such as texts).

In this paper, we propose a methodology to help us extract knowledge (about some vocabularies of interest) from a text corpus by querying an engineered ontology. In the proposed method, we use Natural Language Processing (NLP) [4], [5] and Formal Concept Analysis (FCA) [6], [7] for engineering an ontology as a backbone for reasoning and semantic knowledge extraction. NLP and FCA have been both recognized as remarkable tools for ontology building [8]–[11].

The significance of FCA in knowledge representation and data mining has been shown in earlier research works [12]–[15]. In the following, we describe our proposed methodology in more detail and apply it on an interesting use case, where the aim is to extract knowledge about vocabularies such as companies and products using a list of documents containing news articles and reports in IT and pharmaceutical domain. We further discuss about the opportunities on how to improve the proposed methodology as a topic for follow up research.

II. METHODS

In this section we guide you, step by step, through our methodology for knowledge extraction about a given set of controlled vocabularies from a text corpus. Fig. 1 demonstrates our proposed pipeline for that methodology and summarizes all the steps from raw text pre-processing to ontology building and querying it, for the purpose of knowledge extraction about a given list of controlled vocabularies.

A. From Text Corpus to the Triplets Containing Controlled Vocabularies

In our proposed methodology, the first step is to extract triplets of the form \((subject, predicate, object)\) from texts which provides information about the vocabularies we are interested in. For that purpose, we convert .pdf files to .txt files (for building a text corpus) and then we clean the corpus by removing all web links, email addresses, non-English sentences, and all other irrelevant piece of data that is captured in the corpus. We then only keep those sentences that contain one of the controlled vocabularies. Otherwise, we face with huge amount of sentences which may not provide any valuable information about the vocabularies of interest. Once we extracted sentences which contain controlled vocabularies, we apply a sequence of natural language processing (NLP) operators such as tokenization (i.e., splitting sentences into...
words), part-of-speech (POS) tagging (i.e., labelling each word with POS tags such as noun, verb, adjective etc.), stop word removal (i.e., ignoring non-informative words such as the, a, etc.), and chunking (i.e., defining a grammar to extract the piece of sentence we are interested in). The latter is done by looking for grammars of form \( NP \rightarrow VP \), where \( NP \) denotes a noun phrase and \( VP \) denotes a verb phrase, as a predefined sequence of POS tags.

The final aim of the above NLP operators is just to extract meaningful triplets of the form \( \langle subject, predicate, object \rangle \), where either subject or object of those triplets is one of the controlled vocabularies. From now on, we may refer to the above triplet as \( SVO \) which indicates the relation between a subject \( S \) and object \( O \) via a verb (i.e., predicate) \( V \). The list of all extracted \( SVOs \) are stored in an XML format which is easier to be used for next steps during ontology engineering. Note that at this stage, we convert all the verbs to their corresponding lemmatized forms. For instance, the verbs goes, went, going will all be represented by go in the XML file.

### B. Extracting Ontological Concepts from Triplets Containing Controlled Vocabularies

Once the triplets containing controlled vocabularies are extracted from text corpus, we create a formal context by mapping subjects and objects of the triplets to the verbs with which they form a triplet. A formal context is represented in the form of a binary table \( < X, Y, I > \), where \( X \) denotes the set of individuals (rows) that are characterized by a set \( Y \) of attributes (columns), and \( I(\cdot, \cdot) \) denotes the binary relation between members of sets \( X \) and \( Y \).

In our example, the set of all subjects and objects in the list of \( SVO \) triplets will be considered as rows, and the set of all verbs in those triplets will be considered as columns, but in two different forms \( V_{\text{subj}} \) and \( V_{\text{obj}} \). In other words, the number of columns in the formal context is twice as the number of unique verbs in triplets \( SVO \), while the number of rows in the formal context is the number of unique subjects and objects in those triplets. For each triplet \( SVO \), the subject \( S \) is linked to \( V_{\text{subj}} \) and the object \( O \) is linked to \( V_{\text{obj}} \), i.e., \( I(S, V_{\text{subj}}) = 1 \) and \( I(O, V_{\text{obj}}) = 1 \). The argument behind defining two forms for each verb in the formal context is explained in following example. Imagine we have the triplet (Google, Hire, John Smith). Although Google and John Smith are both related to verb Hire, they are different in the sense that the former can be characterized by its feature that is being able to hire someone, while the latter is characterized by its feature that can be hired by some organization. Fig. 2 demonstrates how different extracted triplets will contribute in creating a formal context.

After creating the formal context \( < X, Y, I > \), the formal concept analysis (FCA) will be used to automatically generates a set of formal concepts and a concept lattice that determines the hierarchy between the formal concepts [6], [7]. Each formal concept is characterized by a pair \( (A, B) \) where \( A \subseteq X \) denotes a subset of individuals that have a subset \( B \subseteq Y \) of attributes in common. Each formal concept is uniquely characterized by its extent \( A \) or intent \( B \) and mimics the human interpretation of real concepts in the world. In this paper we use a python module called concepts for building formal concepts, although formal concepts can be extracted using different methods [6], [7], [16].

Once the set of all formal concepts are identified, we apply K-means clustering algorithm to group similar concepts. In order to do so, we represent each formal concept \( (A, B) \) by a binary vector \( w \) of size \( |Y| \), where \( w(i) = 1 \) if \( y_i \in B \), and \( w(i) = 0 \) otherwise. Here \( |Y| \) denotes the cardinality of the attribute set \( Y \) in the formal context and \( y_i \) denotes the \( i \)-th attribute (i.e., \( i \)-th column in the formal context). Thanks to the clustering algorithm, those formal concepts whose intents \( B \) have similar attributes will share the same cluster. The idea behind clustering is to define a set of ontological concepts (also known as ontological classes). The concepts within each cluster then denote the instances or individuals to be defined for each ontological class.

Note that the instances we defined above are in fact formal concepts that can be uniquely determined. The number of formal concepts for big formal contexts can be extremely
large. Moreover, what matters for ontology is to find a way to associate each individual (i.e., rows of the formal context) to a class or cluster. To make this happen, we use a reduced labelling method. As the extent and intent of each formal concept overlaps with those of its super-concept and sub-concept, a reduced labelling method can be used to represent each extent and intent of a formal concept by its reduced form [17]. Reduced labelling simplifies visualization of the complex concept lattices that are derived from big formal contexts with so many formal concepts.

According to this method, for each individual \( x \in X \) in the formal context, there is one and only one formal concept \((A, B)\) such that \( \gamma(x) := (\{\{x\}\}^u, \{x\}^u) = (A, B) \). Here, \( (.)^u \) and \( (.)^d \) respectively denote \emph{up} and \emph{down} operators in FCA. In fact for a formal concept \((C, D)\), \( C^u \) denotes a set of those attributes that all individuals in \( C \subset X \) have in common. Similarly, \( D^d \) denotes a set of those individuals that all attributes in \( D \subset Y \) have in common. Note that for a formal concept \((C, D)\), we always have \( C^u = D \) and \( D^d = C \).

Thanks to the reduced labelling, each of the individuals \( x \in X \) in the formal context can be associated to one of the formal concepts \( \gamma(x) = (A, B) \) that are already clustered by K-means. This means that if for instance \( \gamma(x) = (A, B) \) belongs to cluster number \( k \in \{1, ..., K\} \), the individual \( x \) will be identified as an \emph{instance} of the \( k \)-th class in the ontology. Using this method we can associate each individual \( x \in X \) to one of existing ontological \emph{class}.

So far, the ontological classes as well as their corresponding instances have been identified. In the next section, we describe how to complete that ontology by adding object properties between classes and relation between instances.

\section{C. Completing Ontology by Adding Object Properties, Relations, and Data Type Properties}

After identification of ontological classes and the individuals within each class, the next step towards completing the ontology is to add \emph{object properties} between different classes and the \emph{relation} between instances of each. This is done by screening all triplets in the XML file, and adding edges between instances (for relations) and also between classes that contain those instances. For example, for triplet \((\text{Google, Hire, John Smith})\), the directional relation \textit{Hire} from \textit{Google} to \textit{John Smith} will be created. Moreover, an object property with the same name \textit{Hire} will be added from class \textit{company} (to which \textit{Google} belongs) to class \textit{Person} (to which \textit{John Smith} belongs) in the ontology. Note that right after adding each object property or relation we also add its corresponding inverse relation (i.e., \textit{inverse-Hire} in example above). Such directional relation is necessary to capture both types of relations that can be derived from one verb, i.e., hiring someone or being hired by some organization.

The next step is then to add \emph{data type properties} for each instance of a class. In order to do so, we use named entity detection technique in NLP. If there is any entity (such as Time, Money, Date, etc.) detected by NLP for a given instance of a class, we insert that entity as one of the data type properties for that instance. Moreover, if there is a given list of synonyms for our controlled vocabularies, we will add them as data type properties of those individuals. This will increase our capability in semantic knowledge extraction when we query the ontology. In addition to data type properties, we can further add taxonomies, OWL restrictions on classes and individuals (such as data formats), as well as intra-class constraints such as disjointness or sub-class and super-class relations. There is always a way to improve an engineered ontology given additional source of domain expert knowledge.

Once the ontology is built, we can start querying it to extract knowledge that was initially hidden in the text corpus. However, thanks to the engineered ontology we can now deduce explicit and implicit knowledge that is somehow embedded in the ontology in the form of a knowledge graph. In the following section, we describe the results of applying our proposed methods on a real use case for extracting knowledge about a list of controlled vocabularies from textual data.

\section{III. Results}

We applied our proposed methodology to a real problem, where the aim was to extract knowledge about \( \sim 250 \) given controlled vocabularies from \( \sim 800 \) pdf files (2-10 pages each) consisting of news articles and reports collected from web in IT and pharmaceutical domains. We first started by cleaning and pre-processing the corpus using NLP techniques that were mentioned earlier. As a result of data cleaning, we ended up with 233 sentences that contained at least one of the controlled vocabularies. There is no surprise that some of those controlled vocabularies never showed up in the text corpus.

We created triplets of the form \((\text{subject, predicate, object})\) from extracted sentences above. We then generated a formal context in the same way as described before as a binary table that encodes the relation between subjects / objects and the verbs of triplets. Note that for each verb, there are two columns available in the formal context to identify the link between the verb \( V \) and both subject \( S \) and object \( O \) of corresponding triplet \( SVO \). The formal context that was created by the extracted triplets in our dataset consisted of 285 individual (rows) and 118 attributes (columns). Note that not all individuals were among the controlled vocabularies because it is sufficient for a sentence to be extracted as one of the triplets of interest if either subject or object of that triplet is a member of controlled vocabularies.

After applying FCA to the formal context above, 26712 formal concepts were identified. We then applied K-means clustering algorithm with \( K = 3 \) to group formal concepts into three clusters based on the similarity between their intents, i.e., the attributes that are characteristics of those formal concepts. We then kept only those formal concepts that could be directly associated to the 285 individuals in the formal context via reduced labelling function. In other words, we only kept formal concepts \((A, B)\) for which there is an individual \( x \in X \) so that \( \gamma(x) = (A, B) \). Fig. 3 depicts the output of K-means clustering for \( K = 3 \).
Fig. 3. The formal concepts clustered into three categories based on the similarity of their intents. Each circle represents a formal concept and each cluster is identified by a separate colour. For the sake of visualization, the formal concepts are projected into a two dimensional space, where each dimension represents one of the two principle components of the data. Thanks to the reduced labelling function $\gamma(x)$, we can identify association between each of the individuals in the formal context (including controlled vocabularies) to one of the formal concepts. After filtering out the rest of concepts for which there is no $\gamma(x) = \{A,B\}$, we reduce number of formal concepts from 26712 to 285. We then looked at the individuals $x$ that are grouped in the same cluster. Fig. 4 depicts examples of individuals that are categorized into three separate clusters, indicating that most of the individuals that belong to the same class are really part of the same group (mainly Companies, Products or People). These three groups then form the basis of the ontology (as the first three classes to be defined in the ontology).

Fig. 4. Examples of individuals that are categorized into three clusters we identified in Fig. 3. The colour of each column is consistent with the colour of its corresponding cluster in Fig. 3. This table shows that each cluster identifies a separate group of individuals (here Persons in cluster-1, Products in cluster-2, and Companies in cluster-3). These three clusters from the first three classes in the ontology. Individuals within each cluster will then be considered as instances of its corresponding ontological class.

We also suggest to have the possibility of adding / removing or even merging some of those classes, depending on a domain expert knowledge if it is available. To show such a possibility, we created a fourth class in our ontology named as Location, where its instances were initially grouped in one of those initial three clusters. If you increase the number $K$ of clusters in the beginning, the chance of appearing a new cluster that automatically identifies instances of the class of location will increase. However, it may also cause further separation of a bigger class (such as Products) into smaller groups (which can be later merged according to a domain knowledge expert).

To complete the ontology, we further added object properties between classes and also relations between instances of those classes. We used OWLready python library [18] for that purpose, to connect subjects and objects of triplets SVO via verbs. Moreover, we added named entities (identified by NLP) as data type properties to some instances of each class. Once the ontology was complete, we started querying it by using SPARQL query language. As an example, we queried the ontology to find answers to following questions: (1) What is the list of products produced by each company?, and (2) What is the list of persons hired by Apple? Fig. 5 depicts SPARQL queries for above questions and show their output results.

IV. DISCUSSIONS

Building ontologies from unstructured data allows us to semantically extract knowledge about a given domain of interest. In this paper, we proposed a methodology to guide us at each step from data cleaning to querying an engineered ontology for the purpose of knowledge extraction about a list of controlled vocabularies.

Although building an ontology from a text corpus is a full automatic process, it is really appealing to apply it in an automated manner. We can use a set of tools and techniques that are available in the field of text mining, such as using NLP to extract entities from the text and using clustering algorithms to group similar entities together. The resulting clusters can be used as the basis for constructing an ontology, where each cluster represents a distinct concept or theme.

In the next step, we can use a process called concept discovery to identify new concepts that were not included in the initial list of controlled vocabularies. This can be done using techniques such as association rule mining or gene expression analysis.

Once we have a list of candidate concepts, we can then use a process called concept evaluation to determine which of these concepts should be included in the final ontology. This can be done using techniques such as heuristic evaluation or user feedback.

Finally, we can use a process called concept description to provide a more detailed description of each concept. This can be done using techniques such as natural language generation or domain-specific generation.

In conclusion, building an ontology from unstructured data is a complex and challenging task, but it is also extremely valuable for extracting knowledge and understanding the structure of a domain.
Although in our proposed methodology we used FCA for clustering individuals (including controlled vocabularies) in the formal context (to be considered as instances of the ontological classes), different alternative methods are also available. For instance, we can also cluster those individuals \( x \in X \) by using only their attributes \( \{y|I(x,y) = 1\} \subset Y \) and not the subset of attributes \( B \subset Y \) that are associated to their equivalent formal concept in the reduced label form \( \gamma(x) = (A, B) \). This approach may reduce the computational complexity of clustering individuals, but in the expense of losing information on the hidden hierarchies among the concepts which can be revealed by FCA.

The engineered ontology is the cornerstone of a knowledge model, which can be further improved by adding more data type properties. For instance, completing data type properties by adding synonyms of each individual and also identification of equivalent verbs (that indicate relations between different individuals) will improve deriving implication rules and semantic knowledge from that ontology. Moreover, advancing text pre-processing methods by NLP and named entity semantic knowledge from that ontology. Moreover, advancing text pre-processing methods by NLP and named entity detection in order to extract meaningful triplets from texts can significantly improve the trustworthiness of the extracted knowledge and insights from ontology.

Besides building an ontology from text corpus (as described in this paper), if we can merge it with other existing ontologies in that domain, we can end up with more powerful and more complete ontologies. Another topic of interest for future work is to find a way to effectively update an engineered ontology using new additional data sources which never stop growing (especially in the web). Bringing more and more automation in handling big data for engineering domain-specific ontologies is also a topic of interest for future research.

V. CONCLUSION

We proposed a methodology on how to extract knowledge from a text corpus about a given list of controlled vocabularies. In the proposed method we take into account NLP techniques for cleaning and pre-processing of texts to extract meaningful triplets about those controlled vocabularies. We then use FCA to extract the list of formal concepts and the hierarchies between them, which will be used to build the ontological classes and instances within each. We finally complete the ontology by adding object properties, relations as well as data type properties extracted using triplets and named entity detection in NLP. After the ontology is engineered, we then query it to find answer for some questions of interest about the controlled vocabularies.

We demonstrated the significance of the proposed methodology by applying it on a real use case, where the aim was to find answers to some questions of interest about companies and products in IT and pharmaceutical domain using a repository of news articles and reports in .pdf format. The underlying method was able to engineer an ontology from unstructured textual data, as the cornerstone of a knowledge model, to be queried for extracting knowledge. We further discussed different possibilities and challenges for improving the trustworthiness of the engineered ontology, and proposed some topics of interest for potential follow up research.

REFERENCES


