ISSN 2007-9737

Gate-based Rules for Extracting Attribute Values

Juan Carlos Blandón Andrade[1], Carlos Mario Zapata Jaramillo[2]

[1] Universidad Católica de Pereira,
Programa de Ingeniería de Sistemas y Telecomunicaciones,
Colombia

[2] Universidad Nacional de Colombia Sede Medellín,
Facultad de Minas,
Colombia

juanc.blandon@ucp.edu.co, cmzapata@unal.edu.co

Abstract. Automated ontology population is intended to enrich ontologies. The main entities of an ontological model are classes, subclasses, attributes (datatype properties), relationships (object properties), and instances. Class instances are important to the scientific community, since some work is devoted to automatically populating ontologies by using statistical methods, information extraction, and natural language processing, among others. The problem is focused on identifying and extracting attribute values of instances. Commonly, such values have a predefined type like numeric, string, boolean, etc. The difficulty arises when you want to know which instance belongs to such values. In this paper we propose an approach based on Natural Language Processing (NLP) and Information Extraction (IE) technologies for extracting attribute values. We use syntactic patterns implemented on the GATE (General Architecture for Text Engineering) tool. The results are independent of the application domain and they exhibit promising values of recall, precision, and F-measure.

Keywords. Automated ontology population, GATE-JAPE patterns, information extraction, natural language processing, ontologies.

1 Introduction

Ontologies have some kinds of entities such as classes, subclasses, attributes, relations, and instances. Classes represent concepts such as person, location, means of transportation, etc. Relations represent a type of association among concepts: the first argument is the domain and the second one is the range. Subclasses are types of subclass-of relation for building taxonomies. Instances are used to represent either elements or individuals belonging to a class in the ontology [18].

Datatype properties are attributes with a different value for each instance of the concept, e.g., color, measurement, etc. A class definition includes either properties or instance attributes inherited from super-classes. For example, if we define a class called *person*, we also define an attribute *has_age*. The subclass *employee* includes this attribute. Hence, when we create an instance *employee_1*, we need to specify a value of age [18].

Ontology population process has several tasks [3]. A task is related to the extraction of relation instances, *i.e.*, founding relations among the different concepts belonging to the ontology [13, 6].

Another task, instance extraction, is related to identify values from any information sources and assign such values to instances. Several authors work on different approaches [4] to ontology population with statistical methods [30, 31], information extraction [11], natural language processing techniques [24, 7], machine learning [22], and others [36, 16].

Automated ontology population is intended to identify concept and relation instances by using a computational tool. An empty ontology is used as input and the ontology with the corresponding instances is the output.

This process is commonly made by hand, but it is very expensive in terms of time and manpower. Automated ontology population is crucial to the semantic web for structuring web information, so machines can understand it, and refining the searches made by users [14].

In the aforementioned proposals, attribute values can be used to describe an object [5], *e.g.*, number, color, measure, id, price, and others. Attribute values have a predefined type: numeric, string, Boolean, date, double, etc.

The problem arises when gathering the major amount of attributes values and linking them to instances of the classes they belong. The state of the art exhibits a need for automated ontology population from different sources [8, 23, 40, 26, 15, 41, 32, 38].

In this paper we propose an approach based on Natural Language Processing (NLP) and Information Extraction (IE) technologies for extracting attribute values. We use syntactic patterns implemented on the GATE (General Architecture for Text Engineering) tool.

The natural language processing refers to the analysis and representation of texts with some computational tools intended to use linguistic processing at the morphological, syntactic, and semantic level [24].

We select the GATE architecture, since it allows for using information extraction tools and defining rules with generic patterns in the Java Annotation Patterns Engine (JAPE) based on the Common Pattern Specification Language (CPSL) [12, 19].

We extract attribute values belonging to instances of classes into text. In addition, we use documents belonging to twelve different domains for testing the process. Promising values of recall, precision, and F-measure are shown.

This paper is organized as follows: in Section 2 we present the related work; in Section 3 we justify the problem; in Section 4 we propose the method; in Section 5 we show the results and discussion. Finally, in Section 6 we discuss conclusions.

2 Related Work

2.1 Theoretical Concepts

2.1.1 Natural Language Processing

Natural Language Processing (NLP) refers to a range of computational techniques for analyzing and representing texts at one or more levels of linguistic analysis in order to process some applications and tasks.

Machine translation, information recovery, summary extraction, intelligent tutors, and voice recognition, among others, are some of the most popular NLP tasks [33].

Computational tools are used for linguistic processing at the morphological, syntactic, and semantic level. In morphological analysis we determine the grammatical category of words.

Syntactic analysis comprises the structure of the sentence and work products *e.g.*, parsing trees and dependencies.

Semantic analysis refers to the text meaning by using structures generated from syntactic analyzers [24, 7].

2.1.2 Information Extraction

IE is concerned with collecting texts in order to transform them into information to be easily understood and analyzed. With IE, the fragments of relevant texts are identified, the relevant information of the fragments is extracted, and, by using such information, a coherent structure can be created. Relevant information contained in the documents is recognized and structured for treating it and recovering it [10, 2].

IE researchers aim to find items interesting for the human analysis from documents. Also, relevant information should be obtained from IE systems, while irrelevant information is ignored. IE systems only deal with specific types of texts with partial results [10].

2.1.3 Ontologies

Ontologies are special types of information collections and they are used to formally model a system structure, *i.e.,* the relevant entities and relationships arising from observation with a particular intention in mind [34]. *"An ontology is a formal, explicit specification of a shared conceptualization"* [35]. An ontology can be thought of as a set of concepts, relations among the concepts, and their instances [27].

Concepts are the main component of the major number of ontology-based formalisms. Attributes describe properties belonging to two types: (i) instance attributes describing concept instances and their values, and (ii) class attributes describing concepts and their values [18]. Instance attributes are defined in the context of concepts and they are inherited to their sub-concepts and instances, *e.g.,* the concept of natural person has an attribute instance called *identification number* [18]. Class attributes can be designed by using sub-concepts and instances, *e.g.,* the concept of person has a class attribute called person type and some of the values person type can take are *natural person* and *legal entity* [18].

2.1.4 Ontology Population

Ontology population is a process for inserting concept and relation instances into an existing ontology [9]. Ontology population process has two inputs: an ontology and an instance extraction engine. Such an engine is responsible for extracting instances of concepts and relations from a corpus. Next, the extracted list of concept and relation instances is used for populating the ontology [27].

Initially, an ontology is required to be populated by the end of this process. Also, a corpus-text set and an instance extraction engine is used to find the instances of both classes and relations in the corpus [1]. Then, the corpus should be processed by using the engine described in the **Figure** 1, so concepts can be located into the text, and a list with instances candidates of concepts and relations is created [27].

These instances are used for ontology population. Ontology population process is shown in Figure 1.

Then, JAPE (Java Annotation Pattern Engine) is used which is intended to support recognition of regular expressions by using annotations in documents. The grammar comprises a set of phases sequentially running as a cascade of finite state transducers over annotations [12].

The left-hand-side (LHS) includes an annotation pattern description while the right-hand-side (RHS) includes annotation manipulation statements. Annotations matched on the LHS of a rule should be referred to the RHS by using labels [19].

2.1.5 Instance Extraction Evaluation

Usually, some methods are used for evaluating an information retrieval algorithm in the testing phase. Precision, recall, and F-measure are criteria needed to evaluate the ontology population process [14, 28].

Precision is the ratio between the number of instance correctly extracted (NICE) and the number of instance extracted (NIE). See equation 1:

$$P = \frac{NICE}{NIE}, \tag{1}$$

Recall is the ratio between the number of instance correctly extracted (NICE) and the number of instances in the corpus (NIC). See equation 2:

$$R = \frac{NICE}{NIC}, \tag{2}$$

F-measure is the harmonic mean between precision and recall. F-measure value is near to zero is the worst case and one is the best. See equation 3:

$$F_1 = 2.\frac{Precision.Recall}{Precision + Recall}. \tag{3}$$

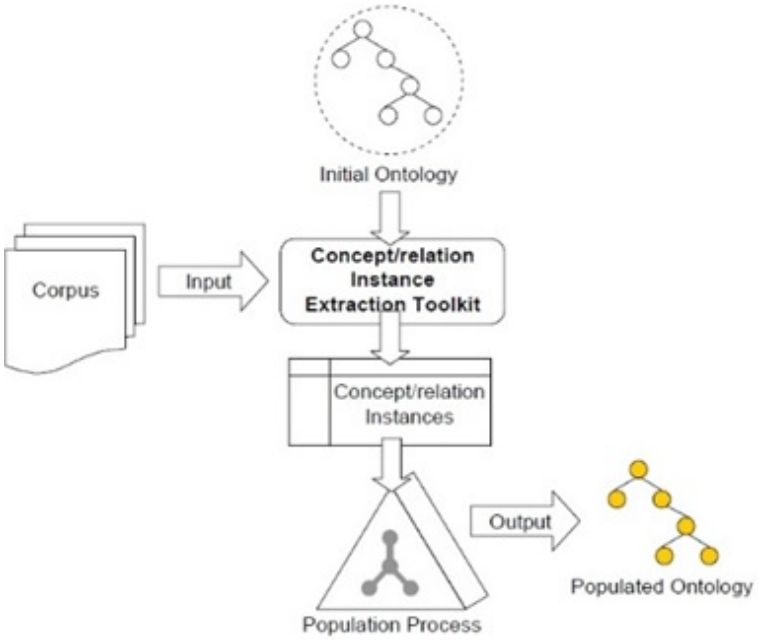

Fig. 1. Ontology population process. Source: [27]

2.2 Background

A perceptron algorithm [23] is proposed for simultaneously learning parameters of local and global features. Attribute values are extracted from unstructured texts in the Chinese language. The approach is based on global features for enhancing the extraction performance. Two types of global feature are defined.

The first is called *boundary distribution* and the second one is called *value-name dependency*. Experiments are carried out on different kinds of entities, *e.g.,* a mobile phone, for extracting values of attributes such as CPU, *operating system, screen, camera and memory*. Experimental results exhibit high levels of precision and recall and, according to the authors, they can be generalized to open-domain.

A rule extraction approach for word sense disambiguation in English propositions by using attribute features [40] is based on the theory of formal concept analysis. The authors obtained a precision of 93.2%. A method for extracting attribute values based on snippet analysis form a search engine [41] is applied to locate the candidate attribute values in snippets. Sufficient training data can be automatically generated by matching triples back to snippets and titles in order to avoid redundancy of correct value in many fragments, all the individual predictions are assembled together by voting. Experiments on the celebrity domain reach 85% of precision.

An automatic non-supervised and domain-independent methodology to extend ontological classes in terms of learning concept attributes, datatypes, value ranges, and measurement units [32] is proposed. Data sparseness of pattern-based approaches is minimized by using the web as a massive learning corpus for retrieving data and inferring information distribution with highly contextualized queries. The corpus is automatically updated according to the knowledge already acquired. Results are manually checked, showing reliable results and a reasonable learning performance.

A general methodology for extracting attribute-value pairs from web pages [38] has two phases: i) candidate generation, in which attribute-value candidates are annotated syntactically, and ii) candidate filtering, where semantically improbable

annotations are removed. The system is tested by processing 10 billion web pages in 6 hours and extracting 10 billion attribute-value pairs. The system achieves 70% F-measure compared to a hand-annotated corpus.

An automated method for extracting ontology attribute-values based on web [42] is proposed. First, a method based on a seed set is described and related to the interaction between relevant sentence selection by including attribute values and attribute-value extraction. Accordingly, we can extract and expand the target attribute value set by the redundancy of web. Then, the authors construct the seed set by using an automatic method. Finally, they build hierarchical clusters of the candidate attribute values. The average of the experiments they performed obtained precision of 72% and recall of 89%.

A method for extracting sets of attribute names by using the bootstrapping algorithm [25] on semi-structured documents, *i.e.,* web documents is proposed. A method for extracting attributes and their values from web pages [39] includes word distributions estimated from plain Web pages. The word distribution is estimated by consulting ontologies built from HTML tables for estimating the probability of each word occurring on Web pages. They used the Good Turing Estimation in calculating probabilities and they introduced a rare role and a sentence role to filter out useless blocks in Web pages.

A framework for generating attribute value extractors [29] can be adapted for dealing with specific types of data sources and incorporating distinct types of heuristics for achieving good extraction performance. The experimental results are at least as good as results in the state of the art.

3 Problem

Ontologies can constitute alternatives for storing knowledge at the concept and instance level. Domain experts should find concept names and their relations and instances for populating ontologies, but this process is expensive: the process is time-consuming and hand-made. Ontology population is needed for obtaining useful information from texts and comprises enrichment by means of class and relationship instances by using an existing ontology as input [2].

Attribute-value extraction linked to instances is still needed. Even though the state-of-the-art review exhibits the extraction of attribute values, some problems still remain: (i) all of the methods are domain-dependent; and (ii) some methods ignore the attribute values implied by the adjectives.

An ontology can be formally defined as we show in equation 4 [15, 17]:

$$O = (C, H, I, R, P, A), \tag{4}$$

where, $C = C^C \cup C^I$ is the set of entities of the ontology. C^C set comprises classes, *i.e.*, concepts representing entities (for example, *Person* $\in C^C$) describing a set of objects, class instances in the C^I set (for example *Erik* $\in C^I$).

$H = \{kind\_of\,(c_1, c_2) \mid c_1 \in C^C, c_2 \in C^C\}$ is the set of taxonomic relations between concepts, which define a concept hierarchy. Taxonomic relationships and are denoted by kind_of(c_1,c_2), meaning c_1 is a subclass of c_2, for instance, kind_of (Lawyer,Person).

$R = \{rel_k\,(c_1, c_2, \cdots, c_n) \mid \forall i,\, c_i \in C^C\}$ is the set of nontaxonomic ontology relations, *e.g.*, represents(Lawyer,Client).

$P = \{prop^C(c_k, datatype) \mid \in C^C\}$ is the set of properties of ontology entities. The relation $prop^C$ defines the basic datatype of a class property. For instance, subject (Case, String) is an example of a $prop^C$ property.

$I = \{is\_a\,(c_1, c_2) \mid c_1 \in C^I, c_2 \in C^C\} \bigcup \{prop^I\,(c_k, value) \mid c_k \in C^I\} \bigcup \{rel_k\,(c_1, c_2, \cdots, c_n) \mid \forall i,\, c_i \in C^I\}$ is the set of instance relations related to the C^C $(e.g.\text{“}is\_a\,(Anne, Client)'')$, P $(e.g.\text{“}subject\,(Case12, \text{“}adoption'')'')$ and R $(e.g.\text{“}represents\,(Erik, Anne)'')$ sets.

$A = \{condition_x \Rightarrow condition_y\,(c_1, c_2, \cdots, c_n) \mid \forall j,\, c_j \in C^C$ is a set of axioms and rules allowing for checking the consistency of

an ontology and inferring new knowledge by using some inference mechanism. The term $condition_x$ is given by $condition_x = \{(cond_1, cond_2, \cdots, cond_n) \mid \forall z,\ cond_z \in H \cup I \cup R\}$. For instance,"$\forall Defense\_Argument$, Oldcase, NewCase, applied_to (Defense_Argument, OldCase), similar_to $(OldCase, NewCase) \Rightarrow applied\_to(Defense\_Argument, NewCase)$" is a rule that indicates that if two legal cases are similar, then the defense argument used in one case could be applied to the other one.

We can deduce the problem is focused on finding P, *i.e.,* attribute values. For example, in the natural language sentence, *the red car* we should identify and create *car* as a class with an attribute called *color*. Next, we should create an instance called *car1* and assign the attribute value *red* to such instance, *i.e., color(car1,"red")*.

4 Method

Our proposal is based on Natural Language Processing (NLP) and Information Extraction (IE) techniques. We use statistical and computational methods coming from NLP, but we enrich such methods by using linguistic features of the natural language texts.

Information Extraction refers to discover and structure information contained in texts, *i.e.,* the extraction of certain entities, relations, and events.

In this work we use Information Extraction techniques as named-entity recognition and co-reference resolution.

We develop our system by using the GATE tool and a pipeline-shaped architecture, *i.e.,* a process should finish for starting the next one. Our system requires a text written in either txt or PDF format as input. The process flow is shown in Figure 2.

Document reset PR (Processing Resource) refers to removing all the annotation sets and their contents from the original document, *i.e.,* the document is returned to its original state. Tokenizer is used to divide the text into words, symbols, numbers, space tokens, and punctuation marks.

Gazetteer refers to identifying entity names based on predefined lists: plain text files with one entry per line.

An index file (*.def) is required for accessing lists and a major type is specified for each list, where a minor type is optionally specified. Lists are compiled by using finite state machines. Sentence splitter is used to segment the text into sentences by using finite-state transducers.

PosTagger includes a lexicon and a rule set in order to add category features: tags based on Penn treebank annotations [37]. NE (Named Entity) transducer (also referred as semantic tagger) is employed to find terms for suggesting entities with new token types. Orthomatcher is based on the entities found with the NE transducer and adds co-reference information to entities by using the orthographic information. HashGazetteer refers to identifying entity names with predefined lists based on hash tables of words. OntoGazetteer incorporates mapping between pairs of term lists and ontology classes. Also, Ontogazetteer is used to assigns the proper class in case of term matching [21]. We use both HashGazeteer and OntoGazetteer in order to find other entity names and confirm the existing ones.

Then, JAPE-Plus transducer allows for defining the rules and recognizing regular expressions in annotations of documents. It comprises a set of pattern(LHS)/action(RHS) rules running sequentially. In our system, we take the rules defined for finding all the coincidences into the text. The rules are designed for tagging classes, instances, and attribute values, but we focus on the extraction of attribute values, the target of our work.

We designed a system with a total of 139 generic syntactic rules created by hand, and we proved it in several domains. The rules can be used for extracting ontology elements (specifically for attribute values). We create 40 rules in total. Those rules allow for extracting attribute values from any document and assigning them to a property belonging class.

Adjectives before nouns contain information about attribute values. For this reason, we create a series of categories as: *Appearance, Authenticity, Brightness, Color, Condition, Cooking, Difficulty, Dimension, Distance, Distributive, Domain, Feelings, Justification, Materials, Numeric, Opinion, Origin, Personality, Position, Primacy, Purpose, Qualification, Quantity, Religion, Shape,*

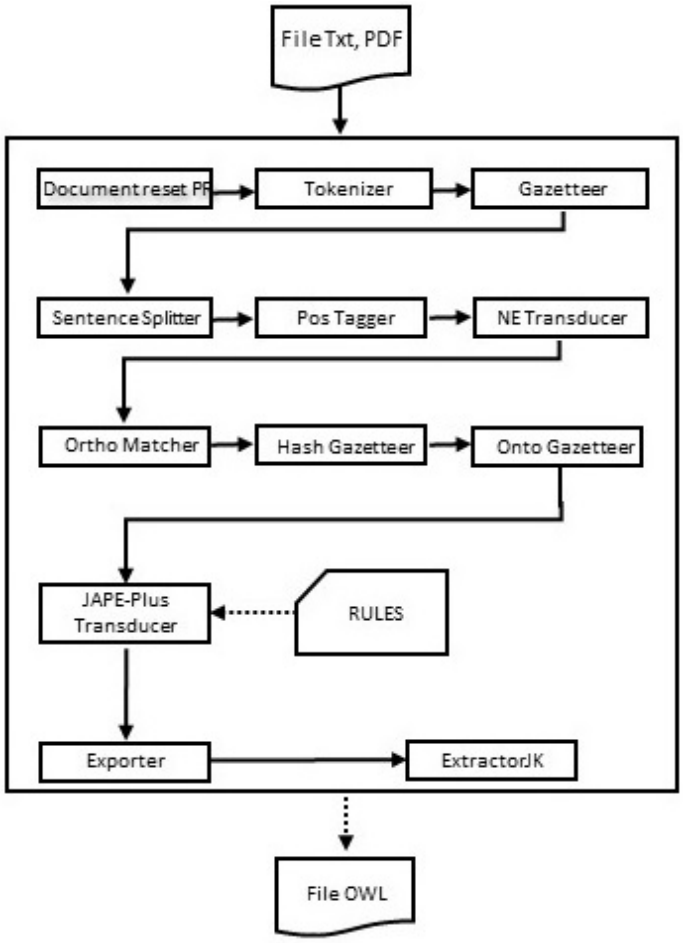

Fig. 2. Extraction process of attribute values Source: The authors

Similarity, Size, Smell, Sound, Speed, State, Taste, Temperature, Texture, Time, Touch, Type, Use, Value, and Weather.

Each category includes several adjectives. For example, difficult, hard, easy, and simple are adjectives belonging to *difficulty*.

If an adjective is followed by a noun, this adjective or attribute value should belong to an instance of the class represented by the noun found. Such categories have been implemented into the JAPE language.

For example, the sentence *you are an authorized person* has the following morphological information: *You* (personal pronoun), *are* (Verb, non 3rd person singular present), *an* (determiner), *authorized* (adjective), and *person* (noun). Accordingly, the system should identify *person* as a class. Then, the system searches into the rules whether the adjective matches the LHS part (pattern) of the JAPE rule depicted in Figure 4.

The first part of the rule (phase, input, and options) comprises the configuration parameters. The second part (macro: Atrib_authenticity) is invoked several times by the main rule (third part), including the words belonging to this category (real, actual, and authorized). We should ascertain the words are tagged as adjectives.

The third part (Rule:Authenticity) is the pattern (main rule).

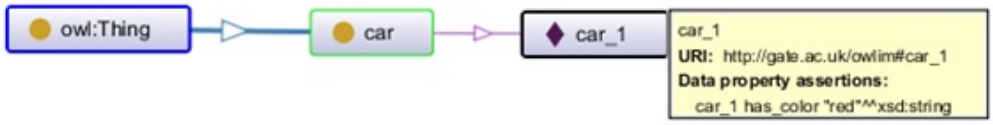

Fig. 3. Ontology example. Source: The authors

Table 1. Results of precision, recall, and f-measure in twelve different domains. Source: The authors

Criteria / Domain	NICE	NIE	NIC	*precision*	*recall*	*F-measure*
Freeze Drying Domain	39	41	42	95,12%	92,86%	93,98%
Showiz Domain	78	80	83	97,50%	93,98%	95,71%
Agriculture Domain	560	630	650	88,89%	86,15%	87,50%
Mental imagery Domain	49	51	54	96,08%	90,74%	93,33%
Air Security Domain	67	72	75	93,06%	89,33%	91,16%
Tuorism Domain	38	40	43	95,00%	88,37%	91,57%
Finance Domain	91	95	105	95,79%	86,67%	91,00%
Politics Domain	50	54	56	92,59%	89,29%	90,91%
Music Domain	68	72	75	94,44%	90,67%	92,52%
Legal Domain	165	180	187	91,67%	88,24%	89,92%
SMS Domain	490	529	560	92,63%	87,50%	89,99%
Twitter Domain	80	84	88	95,24%	90,91%	93,02%
Overall				**94,00%**	**89,56%**	**91,72%**

In the aforementioned example, the macro is searching for the word “authorized” in the macro “atrib_authenticity.” Also, the next word should be a noun, *i.e.,* “*person.*”

When the LHS part is true, the RHS part should be run. Then, the word *person* is tagged as a class. After, we should tag an attribute called *has_Authenticity* and link it to the class. Also, *authorized* is recognized as an attribute value belonging to the *person* class.

We save all labels and information extracted in a file with XML extension by using the exporter. ExtractorJK needs as input the XML file. Then, an ontology is automatically created with all the information of classes, attributes, and instances. In our example, the *person* class is created with an attribute called *has_Authenticity*.

After, we create a generic instance called *person_1* and assign *authorized* as attribute value belonging to the instance in the field *has_Authenticity*. Finally, we create a file of extension OWL with the ontology populated with attribute values found in the text by using ExtractorJK. This process is completely automated.

5 Results and Discussion

Our approach is a contribution on the field of automatic ontology population. The main contribution is related to find adjectives preceding nouns and containing attributes values. For example, we use a document as input with the sentence *the red car*, which has the following morphological information: *The* (article), *red* (adjective), and *car* (noun). Accordingly, the system identified *car* as a class.

Then, the system searched into the rules whether the adjective matched the LHS part (pattern) of the JAPE rule, and how it was found in the *color* category, the system assigned *car1* as instance and *red* color as an attribute value, and it was represented as *“has_color red”*, and its type is string. The system processed the input text and generated OWL file with the ontology; then, the file was loaded in Protégé Software, which it generated the ontology depicted in Figure 3.

Likewise, we obtained a test document presented in a the finances domain [20]. The document has a total of 75 instances and they obtained 80% of precision, 70% of recall, and

```
Phase:Authenticity
Input:  Token Lookup
Options: control = appelt

Macro:Atrib_authenticity
(
{Token.string==~"[Rr]eal",Token.category==JJ}|
{Token.string==~"[Aa]ctual",Token.category==JJ}|
{Token.string==~"[Aa]uthorized",Token.category==JJ}
)

Rule: Authenticity
Priority: 35
(
(Atrib_authenticity):sust2  /*Attribute value */
({Token.category==NN}):sust1  /*Class*/
):inst
-->
{
:sust1.Attribute={rule= "Authenticity"},
:sust2.Clases={rule= "Authenticity"}
}
```

Fig. 4. JAPE rule for extracting attributes values. Source: The authors

75% of F-measure in the instances extraction, but excluding the attributes values hidden in the adjectives. If we consider attribute values, recall falls to 50% and F-measure falls to 61.9%. We used the same document and we manually count 75 possible instances (and 30 more instances by considering the values hidden in the adjectives). Then, we processed the document with our system and we obtained 95.79% of precision, 86.67% of recall, and 91% of F-measure. In this way, we demonstrate the precision and recall of our method is bigger. Something similar could happen to systems found in the state of the art, since recall and F-measure considerably fall when attribute instances are included.

We test the system in twelve distinct domains: freeze drying, legal, agriculture, showbiz, mental imagery, air security, tourism, finance, politics, music, SMS messages, and Twitter conversations, in order to prove domain independence. Results are shown in Table 1. In average we obtained 94% of precision, 89.56% of recall, and 91.72% of F-measure. Such results include the adjectives as attributes values.

Results showed the system received a written document at natural language, extracted classes, created an instance and attributes values was assigned to the instances. The extraction was an automated process, so, users did not intervene.

The system used Reset, Tokenizer, Gazetteer, Sentence Splitter, PosTagger, NE transducer, Orthomatcher, HashGazetteer, OntoGazetteer and JAPE-Plus processes. Then, the OWL file was obtained, which could be shown in an ontologies editor.

Experiments in the Table 1 showed the three criteria: precision, recall, and F-measure, which exceeded the 85%. Thus, we demonstrated the combination of natural language processing, information extraction, adjective categories, and GATE tools, allowed for a high performance in order to extract both classes and attributes values. In addition, we highlight most adjectives before nouns contain information about attribute values, and such information is domain-independent.

Our system only works with documents lower than 100 pages, but our system is automated, what is better compared to other semi-automatic methods [36, 14, 26]. Also, our system can find attribute values hidden in the adjectives, increasing the number of instances identified.

6 Conclusion and Future Work

In this paper we proposed an approach for automatically extracting attribute values from natural language. Our resulting system receives a document in either PDF or TXT format as input. Such a document is then processed by using natural language processing and information extraction techniques.

Specifically, the system includes syntactic GATE-JAPE patterns in order to label entities like classes, relations, instances, and attribute values. An important contribution of this paper is related to find adjectives preceding nouns and containing attributes values, so we create forty categories for classifying adjectives from natural language text. The categories of the classification are used to assign the name the attribute to the class and

then save the adjective as attribute value into the instance. Most of the methods for ontology population miss the adjective values, thus the recall of these methods should approximately decrease between 15% and 20% and f-measure falls to 10% to 15%.

We used the file with extension XML (saved in the Exporter process), which contains all the labels and information extracted from text. Also, we programmed in the Java Language the process called ExtractorJK in order to generate the ontology into file extension OWL as an output.

We used twelve different domains in order to test the domain independence of our system. We evaluated the system by using three criteria: (i) precision; (ii) recall, and (iii) F-measure. Our system exhibited good performance in the proposed domains compared to others methods.

As future work, we plan to test more domains where the ontologies can be seen as valid systems for building knowledge bases. Also, we want to extract relationships among classes and instances. Finally, we intend to implement a system in the Python NLTK tool for comparing our results.

References

1. **Astrakhantsev, N. A., Turdakov, D. Y. (2013).** Automatic construction and enrichment of informal ontologies: A survey. Programming and Computer Software, Vol. 39, No. 1, pp. 34–42. DOI: 10.1134/S0361768813010039.
2. **Benammar, R., Trémeau, A., Maret, P. (2015).** An Approach for Ontology Population Based on Information Extraction Techniques. In On the Move to Meaningful Internet Systems: OTM 2015 Conferences, Lecture Notes in Computer Science. Springer Cham, Rhodes, Greece, pp. 397–404.
3. **Blandón A., J. C. (2017).** Extracción de instancias de una clase desde textos en lenguaje natural independientes del dominio de aplicación. PH.D. Thesis, Universidad Nacional de Colombia, Medellín, Colombia.
4. **Blandón A., J. C., Zapata J., C. M. (2018).** A State-of-the-art Review About Ontology Population. Revista Ingeniería y Desarrollo. Universidad del Norte, Vol. 36, No. 1, pp. 259–284.
5. **Botero T., R. (2010).** Patrones Grasp y Anti-Patrones: un Enfoque Orientado a Objetos desde Lógica de Programación. Entre Ciencia e Ingeniería, Vol. 4, No. 8, pp. 161–173.
6. **Carlson, A., Betteridge, J., Wang, R. C., Hruschka, E. R., Jr., Mitchell, T. M. (2010).** Coupled Semi-supervised Learning for Information Extraction. Proceedings of the Third ACM International Conference on Web Search and Data Mining, WSDM '10, ACM, New York, NY, USA, pp. 101–110. DOI: 10.1145/1718487.1718501.
7. **Clark, A., Fox, C., Lappin, S.**, editors **(2010).** The Handbook of Computational Linguistics and Natural Language Processing. Wiley-Blackwell, London, UK, 1 edition.
8. **Contreras, J. (2004).** Incremento crítico del conocimiento de la Web semántica mediante poblado automático de ontologías. Tesis Doctoral, Facultad de Informática Universidad Politécnica de Madrid, Madrid, Spain.
9. **Corcoglioniti, F., Rospocher, M., Aprosio, A. P. (2016).** Frame-Based Ontology Population with PIKES. IEEE Transactions on Knowledge and Data Engineering, Vol. 28, No. 12, pp. 3261–3275. DOI: 10.1109/TKDE.2016.2602206.
10. **Cowie, J., Lehnert, W. (1996).** Information extraction. Communications of the ACM, Vol. 39, No. 1, pp. 80–91.
11. **Cunningham, H. (2006).** Information Extraction, Automatic. In **Brown, K.**, editor, Encyclopedia of Language & Linguistics. Elsevier, Oxford, UK, second edition, pp. 665–677.
12. **Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V., Ursu, C., Dimitrov, M., Dowman, M., Aswani, N., Roberts, I., Li, Y. (2015).** Developing Language Processing Components with GATE Version 8.1:(a User Guide). University of Sheffield Department of Computer Science, Sheffield, UK.
13. **De Boer, V., van Someren, M., Wielinga, B. J. (2007).** A redundancy-based method for the extraction of relation instances from the Web. International Journal of Human Computer Studies, Vol. 65, No. 9, pp. 816–831. DOI: 10.1016/j.ijhcs.2007.05.002.
14. **Faria, C., Girardi, R. (2011).** An Information Extraction Process for Semi-automatic Ontology Population. **Corchado, E., Snášel, V., Sedano, J., Hassanien, A. E., Calvo, J. L., ?lezak, D.**, editors, Soft Computing Models in Industrial and Environmental Applications, 6th International Conference

SOCO 2011, Springer Berlin Heidelberg, Berlin, Heidelberg, Germany, pp. 319–328.

15. **Faria, C., Serra, I., Girardi, R. (2013).** A domain-independent process for automatic ontology population from text. Science of Computer Programming, Vol. 95, pp. 37–50. DOI: 10.1016/j.scico.2013.12.005.
16. **Garanina, N. O., Sidorova, E. A. (2015).** Ontology population as algebraic information system processing based on multi-agent natural language text analysis algorithms. Programming and Computer Software, Vol. 41, No. 3, pp. 140–148. DOI: 10.1134/S0361768815030044.
17. **Girardi, R. (2010).** Guiding Ontology Learning and Population by Knowledge System Goals. Proceedings of the International Conference on Knowledge Engineering and Ontology Development, KEOD 2010, Valencia, Spain, pp. 480–484.
18. **Gómez Pérez, A., Corcho, O., Fernández López, M. (2004).** Ontological Engineering. Springer-Verlag, London, UK, 1 edition.
19. **Halioui, A., Valtchev, P., Diallo, A. B. (2018).** Bioinformatic workflow extraction from scientific texts based on word sense disambiguation and relation extraction. IEEE/ACM Transactions on Computational Biology and Bioinformatics, pp. 1–1. DOI: 10.1109/TCBB.2018.2847336.
20. **IJntema, W., Sangers, J., Hogenboom, F., Frasincar, F. (2012).** A lexico-semantic pattern language for learning ontology instances from text. Web Semantics: Science, Services and Agents on the World Wide Web, Vol. 15, pp. 37–50. DOI: 10.1016/j.websem.2012.01.002.
21. **Kedad, M. B. Z., Métais, E. (2007).** Natural Language Processing and Information Systems. Springer, Paris, France.
22. **Kordjamshidi, P., Moens, M.-F. (2015).** Global machine learning for spatial ontology population. Web Semantics: Science, Services and Agents on the World Wide Web, Vol. 30, pp. 3–21. DOI: 10.1016/j.websem.2014.06.001.
23. **Liu, Q., Wu, D., Liu, Y., Cheng, X., Pang, L. (2016).** Extracting attribute values for named entities based on global feature. Jisuanji Yanjiu yu Fazhan/Computer Research and Development, Vol. 53, No. 4, pp. 941–948. DOI: 10.7544/issn1000-1239.2016.20140806.
24. **Mitkov, R. (2003).** The Oxford Handbook of Computational Linguistics. Oxford University Press, Oxford, UK, 1 edition.
25. **Nakane, F., Otsubo, M., Hijikata, Y., Nishida, S. (2008).** A basic study on attribute name extraction from the web. 2008 IEEE International Conference on Systems, Man and Cybernetics, Singapore, Singapore, pp. 2161–2166. DOI: 10.1109/ICSMC.2008.4811612.
26. **Nederstigt, L. J., Aanen, S. S., Vandic, D., Frasincar, F. (2014).** FLOPPIES: A Framework for Large-Scale Ontology Population of Product Information from Tabular Data in E-commerce Stores. Decision Support Systems, Vol. 59, pp. 296–311. DOI: 10.1016/j.dss.2014.01.001.
27. **Petasis, G., Karkaletsis, V., Paliouras, G., Krithara, A., Zavitsanos, E. (2011).** Ontology Population and Enrichment: State of the Art. In **Paliouras, G., Spyropoulos, C. D., Tsatsaronis, G.**, editors, Knowledge-Driven Multimedia Information Extraction and Ontology Evolution: Bridging the Semantic Gap. Springer Berlin Heidelberg, Berlin, Heidelberg, Germany, pp. 134–166.
28. **Powers, D. M. (2011).** Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation. Journal of Machine Learning Technologies, Vol. 2, No. 1, pp. 37–63.
29. **Reis, D. d. C., Araújo, R. B., Silva, A. S. d., Ribeiro-Neto, B. A. (2002).** A Framework for Generating Attribute Extractors for Web Data Sources. String Processing and Information Retrieval, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, Lisbon, Portugal, pp. 210–226. DOI: 10.1007/3-540-45735-6-19.
30. **Salton, G., Buckley, C. (1988).** Term-weighting approaches in automatic text retrieval. Information Processing & Management, Vol. 24, No. 5, pp. 513–523. DOI: 10.1016/0306-4573(88)90021-0.
31. **Sammut, C., Webb, G. I. (2010).** Statistical Natural Language Processing. In Encyclopedia of Machine Learning. Springer US, Boston, MA, USA, pp. 916–924.
32. **Sánchez, D. (2010).** A methodology to learn ontological attributes from the Web. Data & Knowledge Engineering, Vol. 69, No. 6, pp. 573–597. DOI: 10.1016/j.datak.2010.01.006.
33. **Shafi, J., Ali, A. (2012).** Defining Relations in Precisiation of Natural Language Processing for Semantic Web. International Journal on Computer Science & Engineering, Vol. 4, No. 5, pp. 723–728.
34. **Song, Q., Liu, J., Wang, X., Wang, J. (2014).** A Novel Automatic Ontology Construction Method

Based on Web Data. 2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), IEEE, Kitakyushu, Japan, pp. 762–765. DOI: 10.1109/IIH-MSP.2014.194.

35. **Staab, S., Studer, R. (2004).** Handbook on ontologies. Springer, London, UK, 2 edition.
36. **Talukdar, P. P., Reisinger, J., Pasca, M., Ravichandran, D., Bhagat, R., Pereira, F. (2008).** Weakly-supervised acquisition of labeled class instances using graph random walks. Proceedings of the Conference on Empirical Methods in Natural Language Processing, Seattle, USA, pp. 582–590.
37. **Taylor, A., Marcus, M., Santorini, B. (2003).** The Penn Treebank: An Overview. In **Abeillé, A.**, editor, Treebanks: Building and Using Parsed Corpora. Springer Netherlands, Dordrecht, Netherlands, pp. 5–22.
38. **Wong, Y. W., Widdows, D., Lokovic, T., Nigam, K. (2009).** Scalable Attribute-Value Extraction from Semi-structured Text. 2009 IEEE International Conference on Data Mining Workshops, Miami, FL, USA, pp. 302–307. DOI: 10.1109/ICDMW.2009.81.
39. **Yoshida, M., Torisawa, K., Tsujii, J. (2003).** Extracting attributes and their values from web pages. Series in machine perception and artificial inteligence, Vol. 55, pp. 179–202.
40. **Yu, J., Li, C., Hong, W., Li, S., Mei, D. (2015).** A new approach of rules extraction for word sense disambiguation by features of attributes. Applied Soft Computing, Vol. 27, pp. 411–419. DOI: 10.1016/j.asoc.2014.10.037.
41. **Zhang, X., Ge, T., Sui, Z. (2013).** Learning to extract attribute values from a search engine with few examples. Proceedings 12th China National Conference, CCL 2013 and First International Symposium, Suzhou, China, pp. 154–165.
42. **Zhao, Q., Sui, Z. (2008).** To extract Ontology attribute value automatically based on WWW. 2008 International Conference on Natural Language Processing and Knowledge Engineering, Beijing, China, pp. 1–7. DOI: 10.1109/NLPKE.2008.4906749.

Article received on 24/09/2020; accepted on 07/06/2021.
Corresponding author is Carlos Mario Zapata Jaramillo.