Interactive Ontology Acquisition from Texts

Rafael Valencia García, Jesualdo-Tomás Fernández Breis, Rodrigo Martínez Béjar

Departamento de Ingeniería de la Información y las Comunicaciones,
Universidad de Murcia, CP30071, Murcia, Spain
rvg1@alu.um.es, {jfernand,rodrigo}@dif.um.es

Abstract. The approach presented in this work could simplify Knowledge Acquisition Processes by means of extracting knowledge directly from natural language texts, so that knowledge could be acquired straight from experts. This approach uses a morphological analyser to improve the processes with the purpose of achieving language independency. The knowledge acquired from text is represented by means of ontological categories.

1 Introduction

Extracting knowledge directly from natural language text is a challenging task, as it would allow obtaining knowledge easily and, what is more, without the intervention of knowledge engineers. Our ultimate goal is the development of tools capable of extracting knowledge from text and able to interact directly with experts of any application domain. To do this, we agree with [2] in that a person who knows a language should in part know the rules of the language. In particular, we accounted for this assumption in designing and implementing a morphological analyser. This paper presents a technique for generating knowledge from text through the combination of knowledge modelling and natural language processing techniques. The main idea behind this approach is straightforward: the system stores knowledge found by the expert in order to be able to automatically identify this knowledge whenever it reappears.

Knowledge has been represented in this work by means of ontological categories. In literature, ontologies are commonly defined as specifications of domain knowledge conceptualisations [13]. An advantage of ontologies is the possibility of making a mathematical study on their properties (see [3]). A series of functions to capture knowledge have been implemented in order to represent the knowledge acquired through ontological components.

The structure of the paper is described as follows. Section 2 presents an overview of the approach presented. Section 3 accounts for the language level used in this work. Section 4 explains the knowledge level. Section 5 describes the system implementation. Finally, in Section 6 some final conclusions are remarked.

2. An Overview of the Approach

The aim of this work was to implement a system able to extract knowledge from natural language texts. More precisely, we have focused on building an ontology from *text*. So, an implicit assumption (Assumption 1) is that ontologies can be used to represent knowledge. As the whole text can be very long, our approach divides it into minor *fragments* in order to facilitate its processing. Furthermore, texts come from some domain or *task*, that is, its content is about some specific application domain. An expert is supposed to build the ontology. This gave rise to another assumption (Assumption 2): experts can build ontologies from text. This expert must have expertise on the specific task described along the text, and the expert is somehow associated with the system and to the text by the task itself. It can be said that knowledge resides inside the text. So, there is another implicit assumption (Assumption 3): text can contain knowledge. Ontologies permit to divide knowledge into categories such as concepts, attributes, relationships, rules, axioms, etc. These knowledge entities can appear explicitly in the text, although sometimes knowledge is only implicitly referred to. Thus, the process attempts to find only explicit knowledge from texts.

By constraining ourselves to the above assumptions, the starting point was an empty *knowledge base*, so that the system is unable to find any knowledge in the text and the expert has to introduce knowledge manually. However, in the approach experts do not just find knowledge in a single fragment, but they also identify expressions from which that knowledge can be derived. The expert identifies all the knowledge entities and (s)he also tells the system the *expressions* (fragments of a sentence) in which they appear. These *expressions-knowledge* associations are stored by the system in order to be used for new knowledge findings thereafter. For instance, if the linguistic expression “car” is considered a concept, then the association (car, concept) would be created. The expert only has to identify these associations once, and from that moment on the system will automatically proceed, and the expert’s task will just be to confirm the results output by the system. Thus, the system has to identify knowledge in fragments as well as knowledge referenced by it. The process revealed some problems: (1) searching for meaningful expressions in a text; (2) deciding what to do when an expression has more than one knowledge association in the knowledge base; and (3) identifying knowledge referred to by “non-concepts”. The first problem must be solved at language (i.e., grammar) level, whereas the other two must be solved at knowledge level. In the following sections, both levels are introduced.

3. The Language Level

This level is in charge of the following tasks: (1) performing morphological analysis; (2) searching for linguistic expressions similar to the text expressions; and (3) providing grammatical rules for inferring knowledge associations from grammatical ones.

A morphological analyser was designed and implemented using the learning algorithm C4.5 [10] in order to categorise each word of each sentence. The results of such analysis (i.e., grammar categories) were used to decide which words have no semantic meaning. In particular, words categorised as preposition, particle, conjunction, interjection, pronoun, and determiner were considered to be semantically meaningless. Also, grammar categories were used to define grammar patterns (see below) in order to support the knowledge inference process.

One word is processed at each step of the algorithm. The system looks for words which are *similar* to the currently processed word in a database. Then, for each similar expression found, it must be checked whether it is *acceptable*. The expressions that are similar and acceptable will be treated by the knowledge level. The detailed description of both functions is done next.

The similar function is in charge of identifying which expressions of the database are similar to the current word of the fragment. In its simplest case, it would be an "equal" function. Nevertheless, this function cannot deal with compound expressions by itself; therefore a function of the type "isPrefix" is needed, which checks whether the current word is a substring of another word or not. In here, a word in the current fragment is "similar" to an expression in the knowledge base if the expression starts with the current word.

The acceptable function was introduced in order to determine whether the current word and a similar expression are not just "similar by chance". The "isPrefix" function has an important drawback: if the current word is the article "a", any expression starting with "a", as "assurance", "added value", "a hundred" or "advert" will be (candidates to be) considered as similar. This function, which limits the number of acceptable options amongst the similar ones, accepts an existing expression in the database if it actually appears in the current fragment. Current words in a text fragment are always single constituents. However, database expressions can contain more than one word. If a word is acceptable, then the current fragment will contain all the words of the database expression. That is, the current word needs to be enlarged to cover all the words of the database expression, creating a new object that contains all the words.

Grammar patterns indicate a relation between words by knowing only their grammar category, and each language has its own grammar patterns. Thus, by using these patterns we can approach this process to be language-independent. The grammar patterns used in this work (see Table 1), are based on the ones presented in [12] for English. In such a table, we say "property" of one word, because *a priori*, the existing relation between two words is unknown. The knowledge level will be in charge of making it explicit the specific relation.

Table 1. Grammar Patterns.

Previous word(s)	Current word	Relation	Example
Adjective	Adjective	The previous word is a property of the current one	Sweetie lovely
Adverb	Adjective	The previous word is a property of the current one	Very popular
	Adverb	The previous word is a property of the current one	Very strongly
Adjective	Noun	The previous word is a property of the current one	Tall boy
Noun		The previous word is a property of the current one	Telephone directory
Noun+prep+(det)		The current Noun is a property of the first one	The table of wood

4. The Knowledge Level

This level is in charge of extracting knowledge from texts. Associations between linguistic expressions and knowledge categories will be made in this layer. For this purpose, the grammar (language) level will also be needed. In this work, four knowledge categories can be assigned to a linguistic expression, namely, concept, attribute, value, and relation. The knowledge extraction process can be split into three phases: (1) knowledge hypotheses formulation; (2) setting hypothesis in a context; and (3) decision making. A hypothesis is an association between a knowledge category and a linguistic expression.

At the beginning, the initial set of hypotheses is set to empty, and the algorithm, which is text fragment-oriented, will finish once all words in the current fragment have been analysed from both grammar and knowledge levels.

Then, for each word, the knowledge level is supplied by the grammar level with a set of acceptable linguistic expressions. For such expressions, these actions are performed: (1) obtaining and sorting the associated knowledge to them in the knowledge base; (2) creating a new hypothesis that matches the knowledge base expression and associates previously sorted associated knowledge to it; and (3) adding the new hypothesis to the list of fragment expressions with its associated knowledge. Obviously, there might be cases where no good options are found. In that case, the user has to be provided with the possibility of defining new knowledge associated to that particular expression. Alternatively, these hypotheses might also be straightforwardly ignored. This implies that the system needs to provide that user such possibility. In case different hypotheses exist, the system will have to make a decision.

At this point, the system is fitted with a set of knowledge hypotheses for the linguistic expression. However, the system's task has not finished yet, unless the hypothesis infers a concept; else, that is, if the inferred knowledge is a different knowledge entity (i.e., attribute, value, relation) some operations still need to be performed. In what follows, we shall explain the operations that need to be performed for different knowledge entities. When the system proposes an attribute as the hypothetical knowledge category, the system searches for the most left-nearby concept in the current fragment. However, that is not always correct. For example in the following fragment: "... due to the **weight** of the **table**", the concept **table** is on the right of its attribute **weight.** In this system, the knowledge level receives support from the grammar level for accomplishing such task. In particular, the grammar patterns are very helpful for this purpose. Once all the hypotheses have been formulated, the system will obtain attributes, concepts, and values, so that when the system has to find relations between knowledge entities, the system makes use of such patterns. For example, in the following fragment: "… the red car …", if the system has tagged **red** as a value, and **car** as a concept in the *search* phase, by using the pattern 'Adj + Noun' the system will find a relation between the concept **car** and the value **red**.

All relations are assumed to be binary between concepts. That is, two concepts need to be found. Let us consider this fragment now: "...synchronized points are stored in the ship's log ... ". In this sentence one of the candidates is on the left hand-side of the linguistic expression "are stored", and the other one on the right hand-side. The system searches for linguistic expressions with associated hypotheses on the left and right hand-side, and candidates are selected according to various criteria:

- If the current expression is associated to a relation of the type "is-a" or "part-of", only concepts can be chosen as candidates as these relations can only exist between concepts. Therefore, the system searches for two concepts, one on the left and one on the right hand-side of the current expression.
- It is very rare that any of the candidates of a relation is a value. Thus, the system is designed to ignore values for such task.
- If an attribute is found, the process of searching for a related concept is the same as the one described above to provide a context for attributes.
- The search process is similar to the one described in previous sections. Candidates are searched (1) in a pre-determined number of linguistic expressions for which the user has inferred knowledge, (2) in the hypotheses obtained in the previous phase and, finally (3) in the user expressions.

In the previous example, the linguistic expression "are stored" is associated to a "part-of" relation, so the system will search a concept on the left hand-side ("synchronized points") and another concept on the rigth hand-side ("ship's log"). In case one knowledge hypothesis has been formulated for one linguistic expression, such knowledge will be associated to the linguistic expression. However, there could be more than one hypothesis for one linguistic expression. This can happen due to (1) (domain dependency) the term meaning can vary according to the domain in which it is used; or (2) (person dependency) it is likely that various experts assign different meanings to the same expression; or (3) (spatial location) if an expression has been used recently with a specific meaning and the same expression appears again, then it is very likely that both expressions mean the same.

Whenever different hypotheses are obtained, the system rearranges them according to the previous three factors. Amongst those factors, spatial location interacts in two different ways. The system considers whether an expression has already been used in the same text file and/or in the current textual fragment (this case is given the highest priority). The various sorting criteria are characterized by three parameters, namely, who recognised the knowledge, the type of domain and whether the expression belongs to the same fragment and/or text.

5 The Software Tool

A tool based on the approach described above has been designed and implemented for acquiring knowledge from texts (text needs to be specified in a text file; i.e., in ASCII format). Text length is irrelevant as it can be split into minor fragments. So the system composes each fragment by one sentence and then the expert (i.e., the end user) can accept this fragment selection. If the expert rejects that selection, (s)he will have to select the next fragment to be analysed. Text samples might belong to one or more specific domains or tasks. The distinction of domains is important as word meanings depend heavily on the domain they appear in. Each expert can be acquainted with knowledge of one or more domains. The system also accounts for the associations between experts and tasks.

An expert on a specific task specifies the file to work with and a new work session is created that is associated to this expert, the task and the file. While processing fragments, the expert finds or recognises knowledge. This knowledge can explicitly or implicitly appear in the textual fragment. If knowledge appears explicitly in the fragment, then the expert has to identify the expression in which such knowledge appears, associating expressions to knowledge or inferring knowledge from them. Recall that expressions and knowledge do not necessarily coincide.

The tool is fitted with two distinct working modes: *query mode* and *maintenance mode*. In the maintenance mode, users are provided with the full functionality of the tool (adding new experts and tasks, associating experts to tasks; saving the work/session(s) in the database, loading previously saved work, etc). The query mode has a reduced functionality. Users can neither perform management activities nor save work/sessions in the database in query mode. In maintenance mode, the user inserts knowledge with the help of the tool; the system proposes knowledge to the user by making use of natural language recognition techniques. On the other hand, in query mode, the user cannot insert new knowledge as ontologies are built automatically.

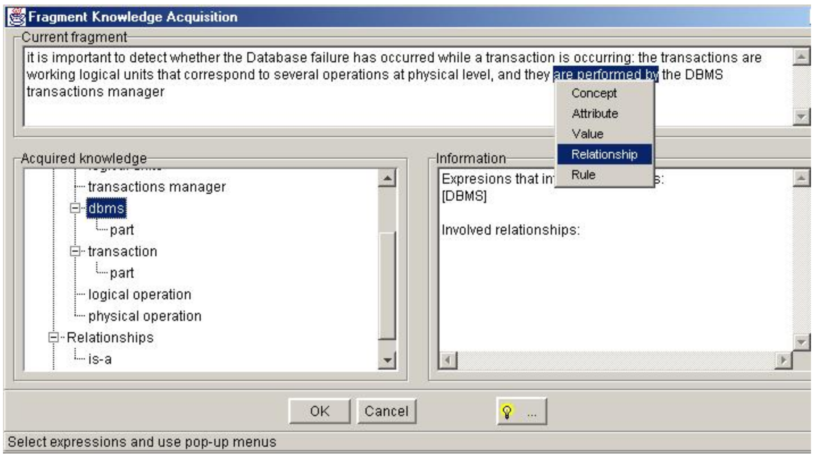
The system is able to infer concepts, attributes, values and relations. However, users can define those axioms they consider necessary or relevant for the application domain.

In our tool, five ontological knowledge categories are used, namely:

- Concepts, representing a class of domain entities.
- Attributes, representing the properties of a given concept.
- Values. The tool is oriented to cover both quantitative and qualitative values.
- Relations.
- Axioms, which are domain rules that include relational operators. For instance, Force = mass * acceleration.

Relations play the same role as in a relation/entity model, although some constraints have been imposed. In this tool, relations are binary, and there is a pre-defined set of relation types: IS-A (this relation allows establishing taxonomies; example: A man is a human being); PART-OF (this mereological relation indicates that a concept is comprised of other ones; example: The engine is part of the car); ASSOCIATION (this accounts for any relation between two concepts that is neither taxonomic nor mereological; example: Hair colour is related to skin colour); and INFLUENCE (this is an association relation in which a concept can influence the existence of another concept).Taxonomic and mereological relations do only exist between two concepts. The remaining relations can exist between whatever two ontological categories, although a relation cannot be part of another relation.

The structure of the ontologies resulting from using our approach can be seen in Figure 1. The tree on the left hand-side of Figure 1 is the ontology, this having three main branches: concepts, relations, and rules (i.e., axioms). Axioms appear as branches of the "rules" node. Each concept has branches for its attributes and each attribute has branches for its values. Relations are branches of the "relationships" node, and relations instances can be viewed on the right side of the screen (i.e., the IS-A relation).



Fragment Knowledge Acquisition

Current fragment

it is important to detect whether the Database failure has occurred while a transaction is occurring: the transactions are working logical units that correspond to several operations at physical level, and they are performed by the DBMS transactions manager

Information

Expresions that in[…]s:
[DBMS]

Involved relationships:

Select expressions and use pop-up menus

Fig. 1. Analysis of a text fragment

In order to evaluate the usefulness of the approach in real settings, a case study (experiment) was performed. It consisted of applying it to several sub-domains of Computer Science with 'simulated experts', namely, 5th year students instructed for the experiment (one expert per sub-domain). The instruction was done through the provision of abundant information concerning the sub-domain they had to work thereafter. Concerning motivation, a list containing descriptions of each sub-domain (already well-known by them through the corresponding subjects studied in the career) utilised in the case study was first shown to them so that they selected those most 'attractive' to them. With this we tried to ensure the 'expert' was motivated enough to do his/her job in the experiment. With all, each expert was given a text from the domain which they had been instructed on. Then, we checked whether the assumptions aforementioned were too strong or not. The results of the knowledge acquisition process from text in this experiment show that the simulated experts study overcame the technical, implicit restrictions of our approach and extracted and represented explicit knowledge from text, as it was our goal. The data of the experiment can be seen in Table 2. The experiment results can be accessed at our web page http://www.klt.dif.um.es.

Table 2. Results of the experiment.

Domain	Concepts	Attributes	Values	IS-A	PART-OF	ASS	INF	OTHERS	Time
Computer control	277	117	114	58	40	16	15	1	3:05
DBMS Design	125	63	99	90	45	0	1	6	4:50
Netware Quality Service (QoS)	41	11	26	36	2	0	0	0	1:06
Process execution	124	23	68	136	7	1	0	2	3:50
Operating Systems	113	71	134	84	39	12	0	6	1:53
Shared memory	72	37	64	46	14	36	12	1	5:56
Compilers	116	44	110	57	47	0	2	1	2:12
Netware protocols	133	56	59	67	35	43	47	0	6:14

6. Discussion and Conclusions

In this paper, an approach that combines knowledge acquisition and natural language recognition techniques has been used for implementing a system capable of extracting knowledge from natural language texts in a supervised mode. The methodology, which is based upon a set of explicit assumptions, presented in this work offers a new and promising method for knowledge acquisition from text. The system has been evaluated in one Computer Science domain and several sets of ontological categories corresponding each to a different sub-domain have been discovered by applying the framework described in this paper. We are confident that this approach for acquiring knowledge from text offers some advantages with respect to pure linguistic methods such as: (1) ambiguity is taken into account (i.e., person dependency, spatial location, domain dependency); (2) rhetoric is not considered; (3) implicit knowledge can be identified and added by the user; (4) the system is incremental and automatic; (5) the system's performance and transparency are acceptable. Two processing levels have been used in this work, namely, knowledge level and language level. The way in which the acquisition process has been divided into allows, in principle, the system to be used for any language, by only adapting the language level to the particular situation.

The way we approach knowledge structuring differs from the one presented in [11]: our knowledge entities are concepts, attributes, values, relations, and rules whereas in the same work, the discussion is about concepts, roles, individuals and axioms. Another difference with the referred work is that the concept acquisition process is performed differently in that work, too: the system's suggestions are hypotheses the user accepts or rejects. In [7], the process is structured in three phases: (1) generating quality labels for hypotheses; (2) estimating the hypotheses credibility; and (3) computing the hypotheses preference order. The expression-oriented analysis to capture knowledge from text in the system presented here is somewhat more general than the classic word-based approach described in [2], for whom words can be derived from other words by means of transformation rules.

Semantics associated to terms has been dealt with also elsewhere. In particular, in [4], the author recognises that semantic variations permit to recognise, for example, verbal and adjectival phrases as conceptually equivalent to nominal terms. Concerning tools for terms acquisition from text, there are others well-known in literature, for instance, LEXTER [1], which was built for term acquisition from French corpora. In our work, we go beyond term extraction to distinguish several kinds of semantic terms through several ontological knowledge categories. The use of ontologies for knowledge acquisition from text is discouraged in [5; 8] for domains in which changes in expert knowledge is rapid and substantial. However, we believe to have shown that our approach can easily be adapted to new requirements.

The methodology presented in this work can be considered an approach for learning domain ontologies, although our purpose is not building a whole ontology but acquire ontological components. In [9], a comparison of ontology learning approaches is made. There, three different types of approaches, according to the type of ontology to be learnt, are distinguished: natural language ontology, domain ontology, and instance ontology. Most of the techniques for learning domain ontologies only generates hierarchies of concepts or use a very reduced set of relations whereas our approach is richer in that sense. Moreover, such techniques are guided by a human knowledge engineer while our purpose is to generate the ontological knowledge automatically (at least in the user mode).

In [6], an architecture for learning ontologies for the Semantic web is presented. Such architecture includes a system for acquiring knowledge from texts. Statistical methods are used for proposing new lexical entries, the ontology engineer being in charge of making the final decision about the creation of new concepts. Moreover, such a system is not only capable of dealing with taxonomic relations but it is also capable of discovering different association rules that describe relations between concepts. However, not much about the consistency of the ontologies built is also done there as the ontologist is free to modify at any stage of the ontology learning process. In our approach, by using the properties stated for the semantic relationships we ensure to some extent consistency in the expert's decision while inferences can be made through the transitive property, if it is the case for the (semantic) relationship under question.

Concerning further work, some improvements should be made with regard to different knowledge entities: (a) relations (i.e., solving situations such as those in which no participants appear on either side of the relation; a possible solution might be checking whether the concept is directly followed by an attribute; here, we might think that it is more likely that the attribute is the second participant and not the concept); (b) pronouns (these are not dealt with in this work and would be another interesting feature to include).

Acknowledgements

This paper has been possible thanks to the financial support of the Séneca Foundation, Coordination Centre for Research, through the Séneca Program (FPI). The research has been performed under projects FIT-150200-2001-320, FIT-070000-2001-785, and Seneca PL/3/FS/00.

References

1. Bourigault, D.: LEXTER, a Natural Language tool for terminology extraction, In Proceedings, 7th EURALEX International Congress, (1996), 771-779, Goteborg, Sweden,.
2. Chomsky, N.: Knowledge of Language: Its Nature, Origin, and Use, Praeger, (1986).
3. Fernández-Breis, J.T., Castellanos-Nieves, D., Valencia-Garcia, R., Vivancos-Vicente, P.J., Martínez-Béjar, R., and De las Heras-González, M.: Towards Scott domains-based topological ontology models. An application to a cancer domain, in Proceedings of International Conference on Formal Ontology in Information Systems. Maine, EEUU, (2001).
4. Jacquemin, C.: Spotting and Discovering Terms through Natural Language Processing, MIT Press, (2001).
5. Jones, D.M., Paton, R.C.: Acquisition of Conceptual Structure in Scientific Theory. In E.Plaza & R. Benjamins (Eds), Proceedings of the European Knowledge Acquisition Workshop, (1997),145-158, Sant Feliu de Guixols, Spain.
6. Maedche, A., Staab, S.: "Ontology Learning for the Semantic Web", IEEE Intelligent Systems, vol. 16, no. 2 (2001) 72 – 79.
7. Musen, M.A.: Domain Ontologies in Software Engineering: Use of Protegé with the EON Architecture. Methods of Information in Medicine, 37 (1998) 540-550.
8. O'Leary, D.E.: Impediments in the use of explicit ontologies for KBs development. International Journal of Human-Computer Studies,46 (1997) 327-338.
9. Omelayenko, B.: Learning of Ontologies for the Web: the Analysis of Existent Approaches. Proceedings of the International Workshop on Web Dynamics, London, UK, (2001).
10. Quinlan, J.R.: C4.5: programs for Machine Learning, San Mateo, Morgan Kaufmann, (1993).
11. Romacker, M., Hahn, U.: Context-based Ambiguity Management for Natural Language Processing, Lecture Notes in Artificial Intelligence 2116 (2001) 184-197.
12. Thomas, L.: Beginning Syntax, Oxford Blackwell, (1993).
13. Van Heijst, G., Schreiber, A.T., Wielinga, B.J.: Using explicit ontologies in KBS development. International Journal of Human-Computer Studies, 45 (1997) 183-292.