An approach for acquiring structured knowledge from text

Rafael Valencia-García, J. T. Fernández-Breis,
Pascual Cantos-Gómez and Rodrigo Martínez-Béjar

Departamento de Ingeniería de la Información y de las Comunicaciones, Universidad de Murcia. Campus Universitario de Espinardo; Departamento de Filología Inglesa, Universidad de Murcia, Campus de La Merced (Murcia)

E-mail: rafavalencia@ono.com; jfernand@dif.ums.es; rodrigo@dif.ums.es; pcantos@um.es

Abstract

Knowledge Acquisition Processes (KAPs) could be simplified by means of extracting knowledge directly from natural language texts. This would simplify the KAP as knowledge engineers would become redundant in this process and knowledge could be acquired straight from experts. The approach presented here uses techniques from both knowledge acquisition and natural language recognition research areas. The KAP described is represented by means of ontologies and has been applied to the leukaemia domain.

1. Introduction

Extracting knowledge directly from natural language text is a challenging task, as it would allow extracting knowledge easily and, what is more, without the intervention of knowledge engineers. The ultimate goal is the development of tools capable of extracting knowledge from text and able to interact directly with experts of any application domain.

This paper presents a technique for generating knowledge from text. This work combines knowledge acquisition and natural language recognition, two disciplines that have been following opposite roads. Knowledge is introduced by experts into the system; these have to specify first where knowledge resides in the text. The system's charge is to associate text and knowledge. As knowledge is expressed by means of natural language, both language and knowledge can reappear throughout the text. Thus, the system has to reach a decision regarding the nature of the linguistic utterance as it might refer to knowledge that has already been included into the system. This work also addresses the usage of the associations between text and knowledge in order to derive knowledge from text fragments.

The whole process can be divided into two main processes: (1) the search phase. This previous stage is completely human guided: an expert reviews a text and knowledge is generated from scratch; and (2) the setting-in-a-context phase. The main aim here is to store knowledge found by experts so that the system can automatically identify this knowledge whenever it reappears. The language chosen for context setting is English.

Amongst the different properties of language those that make it particularly difficult to understand are polysemy and ambiguity. In a multi-domain environment, the same word can refer to different entities or knowledge. The technique presented here attempts to arrange knowledge that is possibly associated to a linguistic expression. The techniques that rely on natural language cannot be perfect due to the intrinsic problems that natural language presents. We shall illustrate some of these problems and come up with some possible solutions. The system might make wrong knowledge associations but the tool has been designed and implemented in such a way that the user can modify the system's decisions whenever these are considered to be inappropriate or wrong. 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. Let us suppose that the user recognises the expression "patient" as a concept. If this expression occurs again in the text, the system will realize that the expression has associated knowledge already. So it will be able to present this knowledge to the user, who will then decide whether this knowledge is correct or not. In particular, the system has been applied to a medical domain, namely, leukaemia, and an ontology has been built by applying the KAP described in this work to natural language texts on this disease.

Knowledge has been represented in this work by means of ontologies. In literature, ontologies are commonly defined as specifications of domain knowledge conceptualisations (Van Heijst, Schreiber and Wielinga 1997). Due to the nature of an ontology, there is not a unique (valid) manner for defining ontologies (Musen 1998a Moreover, several definitions have historically been given to the term ontology, although an ontology is commonly considered to be an enumeration of the relevant concepts in an application area, as well as a definition of classes of concepts and relationships among these classes (Martínez-Béjar and Martín-Rubio 1997). An advantage of ontologies is the possibility of performing a mathematical study on their properties (Martínez-Béjar and Martín-Rubio 1997). The operators formalised in Martínez-Béjar and Martín-Rubio (1997) are implemented in this work in order to build a domain ontology. Two knowledge-based systems have been implemented by means of this approach (Martínez-Béjar, Cadenas-Figueredo y Martín-Rubio 1997).

The structure of the paper can be described as follows: (a) section 2 presents an overview of the approach presented in this work; (b) section 3 accounts for the algorithms used for parsing natural language texts; (c) section 4 addresses how to set words in a context; (d) section 5 explains the system implementation and its application to a real domain; and finally (e) section 6 puts forward some final conclusions.

2. Overview of the approach

This section is devoted to the description of the process of building an ontology. All the steps of the process are covered in this section.

2.1 Process structure

The aim of this work is to implement a system able to extract knowledge from natural language texts; more precisely, to build an ontology from a text. As the whole text can be very long, it seems convenient to divide it into minor fragments in order to facilitate its processing. Furthermore, texts might be on some domain or task, that is, its content might be about some specific application domain. An expert builds the ontology. This expert must have expertise on the specific task described throughout the text, and the expert is somehow associated with the system and to the text by the task itself. The text contains knowledge. In a way, we can say that knowledge resides within the text. Ontologies divide knowledge in knowledge entities such as concepts, attributes, relationships, rules, etc. These knowledge entities can appear explicitly in the text, although sometimes knowledge is only referred to implicitly. Thus, the process attempts to find only explicit knowledge from the text.

The starting point is an empty knowledge base. In this phase, the system is unable to find any knowledge in the text and the expert has to introduce knowledge manually. However, experts do not just find knowledge in a single fragment, but they also identify expressions from which that knowledge can be derived from. The expert identifies all the knowledge entities of the fragment and (s)he also tells the system the expressions in which they appear. These expressions-knowledge associations are stored by the system in order to be used for new knowledge findings thereafter. The expert only has to identify these associations once, and from that moment on the system will perform automatically, and the expert's task will just be to confirm the results produced by the system. In principle, we might think that in a system with a huge knowledge base, the expert just has to take a backseat, divide the text into minor fragments and confirm the system's proposals. Unfortunately, the process is not that simple. The system checks the fragments for expressions with already associated knowledge. A word with associated knowledge can appear in plural or singular, replaced by a pronoun, and verbs can appear in different inflected forms (number, tenses, etc.).

In a huge knowledge base, we are likely to find expressions with multiple knowledge associations, as words can have various meanings and these meanings are reflected in the knowledge base by storing multiple knowledge for the same expression: knowledge pieces that refer to other ones. For instance, an attribute does not exist on its own, it belongs to a concept. A relationship implies the existence of at least two concepts. Thus, the system has to identify knowledge in fragments as well as knowledge referenced by it. The process reveals some problems: (1) searching for expressions in a fragment; (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 two problems are faced in the search phase whereas the third one is dealt with in the setting-in-a-context phase.

These two phases can be approached from different points of view. In this work, those problems are solved according to the solution proposed in (Musen 1998b). This approach does not deal with implicit knowledge: the system cannot find something that has not appeared, and it is the expert's job to identify the implicit knowledge in each fragment.

3. Parsing a text and looking for knowledge

The first goal of this phase is to find expressions with associated knowledge in the knowledge base. Next, it has to decide what to do whenever an expression has more than one knowledge chunk associated to it. Given that the text can be too long, this search performs fragment by fragment. The search process is quite simple and the result of this process is a list containing all the expressions of the fragment already contained in the knowledge base. New expressions are not associated to any knowledge, but are "potentially associated" to a knowledge list. This list contains all the possible knowledge for the expression found in the knowledge base. Immediately after the setting-in-a-context phase, this list is visualised for the user, so that (s)he is given the possibility of choosing potential knowledge from the list. The iterative algorithm used for performing this task can be described as follows.

The initial set of fragment expressions with associated knowledge is empty, and the iterations of the algorithm finish once all words of the current fragment have been analysed. Next, it takes the remaining non-analysed words of the text fragment (current words) and looks for similar words in the already existing expressions in the knowledge base. Then, for each expression of the knowledge base similar to the current word, if it is considered to be an acceptable expression, the following actions are performed: (1) obtain and sort the associated knowledge to the expression present in the knowledge base; (2) create a new expression that matches the knowledge-base expression and associates previously sorted associated knowledge to it as possible knowledge; and (3) add the new expression to the list of fragment expressions with its associated knowledge. The algorithm is expressed in pseudo-code as follows:

```
FUNCTION SEARCH_PROCESS (input TF:TextFragment,
 input KB:KnowledgeBase, output FE:FragmentExpressionsAs
sociatedKnowledge)
VAR current_word: Word;
VAR non_analyzed_words, similar, expression_knowledge:
        List;
VAR current_expression, fragment_expression: Expression;
BEGIN
FE = Ø;
non_analyzed_words=Obtain_words(TF);
WHILE  !empty(non_analyzed_words) DO  BEGIN
  current_word = first(non_analyzed_word);
  similar= Similar(KB, current_word);
  WHILE !empty(similar) DO   BEGIN
    current_expression = first(similar);
```

```
      IF acceptable(current_word, current_expression, TF)
      THEN BEGIN
       expression_knowledge=Obtain_knowledge(KB, current_
       expression);
         expression_knowledge=Sort(expression_knowledge);
       fragment_expression= new Expression;
          fragment_expression.associated_knowledge=  expression_
       knowledge;
       FE=add(FE, fragment_expression);
     ENDIF
    similar=remove(similar, current_expression);
    ENDWHILE
   Non_analyzed_words=remove(non_analyzed_words, current
   word)
   ENDWHILE
   RETURN FE;
   END
```

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 the expression. Alternatively, these expressions might also be straightforwardly ignored. This implies that the system needs to provide that user possibility.

Next, two functions of the algorithm above are described: the similar function and the acceptable function.

3.1 The *similar* function

This function is in charge of identifying which expressions in the knowledge base are similar to the current word found in 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", which is also used in Sánchez-Carreño et al. (2000), is needed. The "isPrefix" function checks whether the current word is a substring of another word or not. For instance, "a" is a substring of the word "and", so the function "isPrefix" will identify "a" as a prefix of "and". It would also be desirable that the function could deal with words families (types associated to a single lemma/lexeme) and other language peculiarities. For instance, if the expression "causes" already exists in the knowledge base and the current fragment contains the word "caused", it would be positive that the system realised that both words actually allude to the same verb (lemma). This task might be partially implemented using parts-of-speech taggers and lemmatisers. 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 function similar is expressed in pseudo-code as follows:

```
FUNCTION Similar (input KB:KnowledgeBase, input word:Word,
output SIMILAR:SetOfSimilarWordsInKB)
VAR wKB: Word;
BEGIN
similar = Ø;
FOR each word wBK in the KB DO BEGIN
  IF isPrefix(word,wBK)
  THEN BEGIN
    SIMILAR= add(SIMILAR, wBK);
  ENDIF
ENDFOR
RETURN SIMILAR;

                    END
```

3.2 The *acceptable* function

This function is an extension of the "*similar*" function. As the "*similar*" function can be very permissive, the "acceptable" function is 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. Therefore, this function limits the number of acceptable options amongst the similar ones. This function has been designed with strong requirements: an existing expression in the database is acceptable if it actually appears in the current fragment.

The function *acceptable* is expressed in pseudo-code as follows:

```
      FUNCTION Acceptable (input curr_w: Word, input curr_exp:
  Expression, input TF: TextFragment, output Acceptable: Boolean)
       IF (AppearsInTextFragment(curr_exp,curr_w.position, TF)
      THEN RETURN True;
       ELSE RETURN False;

                  END
```

Let us now illustrate this with an example that combines both the “similar” and “acceptable” functions. Suppose that the current word is “mortality”. When looking up the database, two expressions are identified as similar: “mortality rate” and “mortality risk”. The following step is to look at the current fragment and check whether any of the previous expressions can be accepted. If the word following “mortality” in the current fragment is “rate”, then the first expression will be considered acceptable whereas if the following word is “risk”, the second one will be accepted. Otherwise, none of them will be considered as acceptable. Constraining this way reduces the applicability of one of the benefits of the similar function: identifying words with the same root but a different suffix as similar.

3.3 New expressions

Current words in a text fragment are always single constituents. However, database expressions can contain more than one word (multiple-word expressions). 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.

3.4 Obtaining associated knowledge

The correctly recognised relationships between expressions (found by the expert) and their associated knowledge are stored in the database. Once an expression is obtained that meets certain requirements (similar and acceptable), knowledge associated with that expression is searched for in the database (additionally, also in other texts and in texts from other experts). Whenever different association possibilities in the database exist, the system sorts them out and displays them.

3.5 Sorting knowledge

In the above description of the search phase, we stated that there might be instances where we might get a set of possible associated knowledge chunks for a single expression. The existence of more than one possibility for associating knowledge is likely due to the following reasons:

- Domain dependency: the different meaning given to a term can vary according to the domain in which it is used. In a domain such as physics, the expression or word "velocity" can be associated to a concept, whereas in other domains that word could be an attribute.
- Person dependency: it is likely that various experts assign different meanings to the same expression. For instance, in Philosophy expressions such as "idea" are restricted to certain authors, assigning very specific meanings to it.
- Spatial location: if an expression has recently been used with a specific meaning and the same expression appears again, then it is very likely that both expressions mean the same.

Whenever different possibilities are considered as inferred knowledge from an expression, the system rearranges them according to the previous three factors. Amongst those factors, the 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 fragment (this case is given the highest priority).

The various sorting criteria are characterized by three parameters, namely, (1) who recognises the knowledge, (2) the type of domain and (3) whether the expression belongs to the same fragment and/or text. Below are some of these sorting criteria:

- By the same expert (person dependency), for the type of domain (domain dependency) and in the same fragment and text (spatial location).
- By the same expert, for the same type of domain and text.
- By a different expert, for the same type of domain, text and fragment.
- By the same expert, for a different type of domain with the same text and fragment.
- By a different expert, for a different type of domain, in the same text and fragment.
- By a different expert, for the same type of domain

and text.

- By the same expert, for a different type of domain and in the same text.
- By a different expert, for a different type of domain and in the same text.
- By the same expert, for the same type of domain but in a different text.
- By a different expert, for the same type of domain in a different text.
- By the same expert, for a different type of domain and a different text.
- By a different expert, for a different type of domain and in a different text.

Once knowledge sorting has concluded, the search phase ends. At this point, the system is likely to have processed the current fragment and expressions present in its database. Additionally, inferred knowledge would have been sorted out according to the above criteria in an attempt to overcome ambiguity.

4. Context assessment

Once the search phase has been performed, the system is fitted with a list of associated knowledge expressions. However, the system's task has not finished yet, unless the inferred knowledge is 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.

In English, attributes usually follow a concept. This property is used by the system to look for concepts attributes belong to. Therefore, current fragments are processed backwards from the current expression on, until an expression that is labelled as a concept is found. For example: ".... the two groups of patients were comparable for patient age and history of antecedent....". If the expression "age" appears in the database linked to an attribute, then it will be recognised in the search phase. Next, the program will search for the most left-nearby concept of "age". The system looks for expressions for which knowledge has already been inferred from in the current working session. In this way, if the user had inferred the concept, say, “person” from the expression “patient”, the system would find it and would associate the concept “person” to the attribute “age”. It is also likely to find no expression with associated knowledge on the left of the attribute. In this case, the system would search the database for the corresponding concept.

In practice, if an attribute is far from its corresponding concept, then it is unlikely to be associated with it. Moreover, it is also possible that the system finds during the search phase an expression near to an attribute for which a concept in the database already exists. Therefore, after checking a predetermined number of expressions for which the user has inferred knowledge, if no concept has been inferred from them, then the system searches the expressions that were found during the search process, looking for one expression with an inferred concept in it. Another heuristic is applied every time the system is looking for a concept and a different attribute is found. In these cases, the system can use the concept associated to the second attribute. This heuristic is not used with those expressions obtained in the search phase due to their lack of stability and reliability.

Let us now consider a further example: “... resulted in high intracellular vit E level...”. It is difficult to know where the attribute is going to appear in the text, although it often appears on the right hand-side of its value. This means that, normally, there would not be many expressions on the right hand-side of “high” with associated knowledge. This implies that the system must be guided by means of the expressions found during the search phase. Let us suppose that the user has just started with a fragment and knowledge has only been associated to expressions on the left of “high”. If the system searched on the right of “high” for an expression with an attribute inferred by the user, the system would not be able to find it unless the tool had been previously used in a medical domain and both the concept “intracellular vit E” and the attribute “level” already existed in the database. In this instance, the system would find the expression “level” in the search phase, and the value “high” would be associated to the attribute “level”. The system would attempt to set the attribute “level” in a context and, at least in the case that the user had just started with the fragment, the system would associate the concept “intracellular vit E”. In this particular case, the context for the expression “high” proposed by the system would be: “intracellular vit E.level.high”. This type of result is frequently obtained in practice. As with concepts, attributes are not the only elements considered whenever contexts for values are provided. If the system finds any expression with any inferred value while searching on the right hand-side of the value, the attribute associated to the latter value is assigned to the value under process. As with concepts, if after analysing a pre-determined number of expressions for which the user has associated knowledge nothing is found, the system searches amongst the expressions found in the search phase.

All relations are assumed to be binary. That is, two elements need to be found. Consider this case now: "...antioxidants inhibit the activation of the NF-kB transcription factor... ". This type of structure is quite frequent between relations: one of the candidates is on the left of the expression, inferring the relation, and the other one on the right side. The system searches for expressions with inferred knowledge 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", any ontological category can be chosen as a candidate 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 (the system is designed to ignore values).
− 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 expressions for which the user has associated knowledge, (2) in the expressions obtained in the search phase and, finally (3) in the user expressions.

4.1 Comparing the two phases

The search phase is a semantic process: words are selected from a text and their meanings are looked up in a database. The database and the knowledge contained are essential for correctly associating knowledge to expressions, whereas context setting is a syntactic-like process (this phase starts once all knowledge has been found in the database). Furthermore, this knowledge will only be used once a context has been found, that is, derived from the former. Attributes are associated to concepts and participants to relations through a simple linear search method.

The search phase is language independent. That is, knowledge is searched in the database and the meaning of words is looked up in a dictionary: both knowledge and dictionary share the same structure independently of the language used. In contrast, providing a context for a knowledge entity is language dependent. Concepts are searched for on the left hand-side of the attributes, as this is where they usually appear in English, whereas attributes are searched for on the right hand-side of the values. If the language chosen had been Spanish, then concepts would have been searched for on the right hand-side of the attributes and attributes on the left hand-side of the values.

5. Implementing 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 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 occur in. The final user of the tool is an expert. Each expert is acquainted with knowledge of one or more domains. The system also accounts for the associations between experts and tasks.

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

The tool is fitted with two distinct working modes: (1) the query mode and (2) the maintenance mode. In 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. The user can neither perform management activities nor save work/sessions in the database. Other differences between both modes include: (a) 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; and (b) in query mode, the user cannot insert new knowledge as ontologies are built automatically.

It needs to be pointed out that the user cannot: (a) input knowledge into the system, (b) select the expert, (c) select the task, (d) select a text for recognition. If this were so, the system would "understand" the user to be an expert and "believe" that the task corresponds to the domain from which knowledge has previously been acquired.

The system is able to infer concepts, attributes, values and relations. However, axioms cannot be automatically inferred. The main problem with axioms is that the number of participating elements is unlimited. Which and how many participants take part in an axiom is as yet unexplored. The quantity of axioms present in a text is not huge compared to the quantity of ontological categories. Under these circumstances, the system has been designed not to recognise axioms in texts, but just concepts, attributes, values and relations. However, users can define those axioms they consider necessary or relevant for the application domain.

5.1 Ontologies in the tool

CommonKADS (Schreiber et al. 1999) uses ontologies as a way of structuring, sharing and reusing the domain knowledge. In CommonKADS, six ontological categories are distinguished: concepts, attributes, values, instances, relations and expressions (axioms). In our tool, only five of these categories are used, namely:

- Concepts: these represent a class of objects in the domain.
- Attributes: these represent the properties of a given concept.
- Values: attributes belong to a domain; attributes such as length are numeric whereas attributes like colour are enumerated. The elements of those domains are the possible values attributes can take. The tool is oriented to cover qualitative values.
- Relations: relations in a domain ontology play the same role as in a relation/entity model, although some

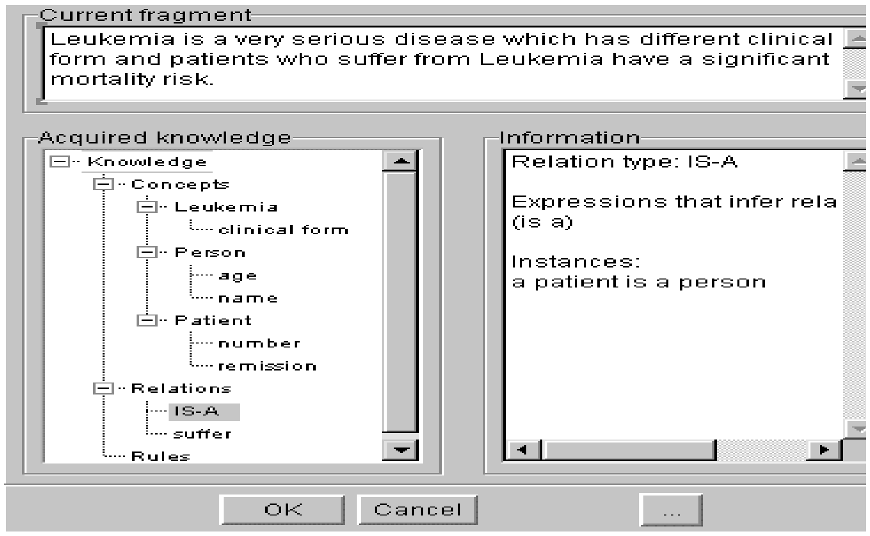
constraints have been imposed. In this tool, relations are binaries and are pre-defined:

1. IS-A: this taxonomic relation allows for establishing conceptual hierarchies. Example: A man is a human being.
2. PART-OF: this mereological relation indicates that a concept is comprised of other ones. Example: The engine is part of the car.
3. ASSOCIATION: this accounts for any relation between two concepts that is neither taxonomic nor mereological. Example: Hair colour is related to skin colour.
4. INFLUENCE: this is an association relation in which a concept can influence the existence of another concept.

The taxonomic and mereological relations do only exist between two concepts. The remaining relations can exist between any two ontological categories, although a relation cannot be part of another relation.

• Axioms: an axiom is a domain rule that includes a relational operator. For instance, Force = mass * acceleration.

In this tool, ontologies are visualised as trees with three branches: (1) one for concepts, (2) one for relations and (3) another one for axioms. The structure of the ontology can be seen in Figure 1. The tree on the left hand-side of Figure 1 is the ontology, 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. The relations are branches of the "relationships" node, and the instances of the relations can be viewed on the right side of the screen (i.e., the IS-A relation in Figure 1).


Current fragment

Leukemia is a very serious disease which has different clinical form and patients who suffer from Leukemia have a significant mortality risk.

Acquired knowledge

```
Knowledge
  Concepts
    Leukemia
      clinical form
    Person
      age
      name
    Patient
      number
      remission
  Relations
    IS-A
    suffer
  Rules
```

Information

Relation type: IS-A

Expressions that infer rela
(is a)

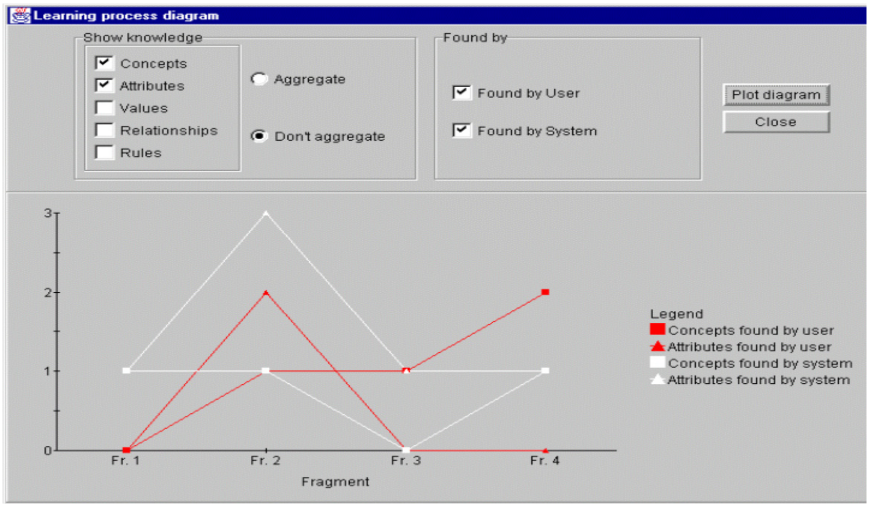
Instances:
a patient is a person

Fig.1 Analysis of a text fragment

5.2 Application

The tool has been applied to a specific domain, namely, leukaemia. Figure 1 displays an analysed fragment of the text belonging to the application domain. The screen shows some acquired knowledge; it displays how “clinical form” has been inferred as an attribute of the concept “leukaemia” and how the relationship “patient suffers from leukaemia” has been inferred from the text “patients who suffer from leukaemia”.

Figure 2 shows a learning diagram and several knowledge options that can be selected. The user can view the learning diagram curve for the ontological entities discussed here (concepts, attributes, relations, rules and values) relative to text fragments analysed. Additionally, the user can also decide whether (s)he wants to see the knowledge associations found/identified by the user, by the system or by both. As a result of the knowledge acquisition process in this domain, an ontology was obtained with the following number of knowledge entities: 169 concepts, 30 (valued) attributes, 165 relationships (113 “is-a” and 52 “part-of”) and 9 axioms (see Appendix).


Learning process diagram

Show knowledge

[✓ Concepts]
[✓ Attributes]
Values
Relationships
Rules

Aggregate
[✓ Don't aggregate]

Found by

[✓ Found by User]
[✓ Found by System]

Plot diagram
Close
3
2
1
0
Fr. 1
Fr. 2
Fr. 3
Fr. 4
Fragment
Legend
🟥 Concepts found by user
▲ Attributes found by user
⬜ Concepts found by system
△ Attributes found by system

Fig.2 The learning diagram

6. Discussion and conclusion

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 fashion. The techniques for acquiring natural language presented in this work offer a different and interesting method within natural language acquisition.

The way we approach knowledge structuring differs from the one presented in Hahn and Schnattinger (1997): our knowledge entities are concepts, attributes, values, relations, and rules whereas in Hahn and Schnattinger (1997), the discussion is about concepts, roles, individuals and axioms. Another difference with Hahn and Schnattinger (1997) is that the concept acquisition process is performed in a different way too: the system's suggestions are hypotheses that the user accepts or rejects, whereas in Hahn and Schnattinger (1997), the process is structured in three phases: (1) generating quality labels for hypotheses; (2) estimating the credibility of the hypotheses; and (3) computing the order of preference of the hypotheses.

The use of ontologies for knowledge acquisition from text is discouraged in Jones and Paton (1997) and O'Leary (1997) 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 system has been used within the leukaemia domain and an ontology with almost two hundred concepts and relationships has been built by applying the framework described in this paper to a set of natural language texts on leukaemia.

We are confident that this approach of recognizing natural language offers some advantages with respect to pure linguistic methods 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.

The way in which the acquisition process has been divided into (search phase and context-in-a-setting phase) allows, in principle, the system to be used for any language. However, some considerations need to be made regarding these two phases. The search phase is hard to be improved, and we prefer to propose some improvements concerning the context-in-a-setting phase regarding different knowledge entities: (a) attributes (i.e., the management of the syntactic form "attribute of the concept"); (b) values (i.e., whenever an attribute does not explicitly appear or no close attribute is found, then the system could search directly in the database); (c) 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); (d) pronouns (these are not dealt with in this work and would be another interesting feature to include).

Acknowledgements: This work has been possible thanks to the financial support of the Fundación Séneca (Centro de Coordinación para la Investigación), Programa Séneca.

References

Hahn, U. and Schnattinger, K. (1997) "An Empirical Evaluation of a System for Text Knowledge Acquisition". *European Knowledge Acquisition Workshop,* Sant Feliu de Guixols, Spain.

Jones, D.M. and R.C. Paton (1997) "Acquisition of Conceptual Structure in Scientific Theory". *European Knowledge Acquisition Workshop*, Sant Feliu de Guixols, Spain.

Martínez-Béjar, R., J. M. Cadenas-Figueredo and F. Martín-Rubio (1997) "Fuzzy Logic in Landscape Assessment". *European Symposium on Intelligent Techniques*, Bari, Italy.

Martínez-Béjar, R. and F. Martín-Rubio (1997) "A Mathematical Function-based Approach for Analysing Elicited Knowledge". *Ninth International Conference on Software Engineering and Knowledge Engineering*, Madrid, Spain.

Musen, M. A. (1998a) "Domain Ontologies in Software Engineering: Use of Protegé with the EON Architecture". *Methods of Information in Medicine*, 37: 540-550.

Musen, M. A. (1998b) "Modern Architectures for Intelligent Systems: Reusable Ontologies and Problem-Solving Methods". *1998 AMIA Annual Symposium*, Orlando (Florida), USA.

O'Leary, D.E. (1997) "Impediments in the use of explicit ontologies for KBs development". *International Journal of Human-Computer Studies,* 46: 327-338.

Sánchez-Carreño, R.I., J. T. Fernández-Breis, R. Martínez-Béjar. R. and P. Cantos-Gómez (2000) "An ontology-based approach to knowledge acquisition from text". *Cuadernos de Filología Inglesa,* 9(1): 191-212.

Schreiber, A. T., J. M. Akkermans, A. A. Anjewierden, R. De Hoog, N. R. Shadbolt, W. Van de Velde, and B. J. Wielinga (1999) "CommonKADS. Engineering and Managing Knowledge. The CommonKADS Methodology". The MIT Press: Cambridge, Massachusetts.

Van Heijst, G., A. T. Schreiber, and B. J. Wielinga (1997) "Using explicit ontologies in KBS development". *International Journal of Human-Computer Studies,* 45: 183-292.

Appendix

Concepts (totalling 169)

Acid	Agent	ALL-1
Allel	ALL-I	AminoAcid
AML-1	AML1/EVI-1	AML-193
Anemia	Antioxidant	AntisenseOligo
Blast cell	Bleeding	Blood
Blood Vessel	Bone Marrow	Ca++
Cancer	Cause	CD-14
CDKN2	Cell	Cell Killing
Cellular mechanism	Cellular superoxide production	Chemeotherapy
Chemical	Children	Chimaeric Gene
Chromosomal abnormality	Chromosome	Clone
Clone 707	Clonogenic	Cluster
CSF	Culture	CyclinA
Cytotoxic	D3	Diet
Differentiation pattern	Disease	DNA
DNA Damage	DNA-PCR	E Acetate
E Succinate	Ed-aSynthase	Effect
Environment Exposure	Epsilon	Erythoid Marker
Esterase	Estimulating factor	Eta
EVI-1	Exons	Exposure
Father	Fell Cell	Fungal infection
Gamma-globina	GATA-1	GATA-2
Gene	Gene Expression	Genotoxic
GM	GM-CSF	Group
gst-pi	Hematological disorder	Heterogenous disease
History	HL-60 cell	Homogenous Disease
Hypergranular	IFN	Inducer
Infection	Interferon	isozyme alpha
isozyme beta1	isozyme beta2	k562 Cell

Karyotipic Aberration	Laser Irradiation	Leukaemia
Leukaemia Cell	Leukcocyte	Lipoic
Lipopolysacharide	liquid suppension	Lympho Glande
Lymphocitic Leukaemia (ALL)	Lymphocitic system	M7
Mammalian Cell	Marker	Maternal Exposure
M-CSF	mdr1	Mechanism
Megakyocity	Membrane	Microgranular
Mitocondrial Marker	Mollecular mechanism	Monocytic Leukemia (AML)
Mother	MRNA	Mrp
MTAP	Myelocytic leukemia (AML)	Nation
NBT Cell	NFkB	nococytic cell
Nocodazole	Oligonucleotide	P53
P65	Parent	Paternal Exposure
Patient	Person	Phenotype
Physician	PKC isozyme	PLZF
PML	PML-RARAlpha	Potential
Precursor	Progenitor	Promyelocytic Leukaemia (APL)
Protein	RA	Radiation
RAR-Alpha-PML	RedBloodCell	RNA
Skin	Spleen	Subclone 707BUF
Symptoms	Therapy	Theta
Thiobarbituric	Thymidine	TMD
Topoisomerase	Topoisomerase I	Topoisomerase II alpha
Topoisomerase II beta	Trans Factor	Transretinoic Acid
Treatment	Trisomy	tRNA
Tumor	USA	VDJRecombinase
Virus	Vit	World
X-Ray		

Attributes (totalling 30)

Concept	Attribute
Chromosome	Number
Clone	Size
Culture	FCS(rich,free)
Diet	AmountAntioxidants
Exposure	to
Exposure	when
Exposure	how
Fungal Infection	MDS
Fungal Infection	AML
Group	Name
Group	Number
Laser Irradiation	Intensity
Laser Irradiation	Wave length
Laser Irradiation	Power
Laser Irradiation	Frequency
Laser Irradiation	Doses
Leukaemia	ClinicalForm
Mammalian Cell	TK
NBT Cell	sign
Patient	Cured
Patient	Remission
Patient	Number
Person	Race
Person	Name
Person	Age
PML-RARAlpha	Type(S,L)
Promyelocytic Leukaemia	t
Trisomy	Number
Usa	Population
X-Ray Exposure	where

Relations (totalling 165)

Type “is-a” (totalling 113)

Agent	IS A	Antioxidant
ALL-1	IS A	Cause
ALL-I	IS A	Gene
AML-1	IS A	Gene
AML1/EVI1	IS A	Protein
AML-193	IS A	Leukaemia Cell
Anemia	IS A	Symptom
Antioxidant	IS A	Inducer
AntisenseOligo	IS A	Antioxidant
Blast Cell	IS A	Cell
Bleeding	IS A	Symptom
C-14	IS A	Antigen
Ca++	IS A	PKC isozyme
Cancer	IS A	Disease
CDKN2	IS A	Gene
Cell Killing	IS A	Effect
Cellular Superoxide	IS A	Differentiation Marker
Chemical	IS A	Cause
Chemotherapy	IS A	Therapy
Children	IS A	Person
Chromosomal abnormality	IS A	Cause
Clone	IS A	Cell
Clone 707	IS A	Mammalian Cell
Clonogenic	IS A	Culture
CSF	IS A	Estimulation Factor
Cyclin A	IS A	Gene Expression
Cytotoxic	IS A	Potential
D3	IS A	Vit
Differentiation Marker	IS A	Marker
Differentiation pattern	IS A	Substance
DNADamage	IS A	Effect
E Acetate	IS A	Vit
E Succinate	IS A	Vit
Ed-aSynthase	IS A	Substance

Effect	IS A	Potential
Environment Exposures	IS A	Exposures
Epsilon	IS A	PKC isozyme
Erythoid Marker	IS A	Marker
Esterase	IS A	Substance
Eta	IS A	PKC isozyme
Exposure	IS A	Cause
Father	IS A	Parent
Fell Cell	IS A	Mammalian Cell
Fungal Ingections	IS A	Infections
GATA-1	IS A	Substance
GATA-2	IS A	Substance
Genotoxic	IS A	Potential
GM-CSF	IS A	Estimulation Factor
gst pi	IS A	Gene Expression
Hematological disorder	IS A	Symptoms
HL-60 Cell	IS A	Leukaemia Cell
IFN	IS A	Gene
Infection	IS A	Symptom
Interferon	IS A	Therapy
Isozyme alpha	IS A	PKC isozyme
Isozyme beta1	IS A	PKC isozyme
Isozyme beta2	IS A	PKC isozyme
k562	IS A	Leukaemia Cell
Karyotipic Aberration	IS A	Chromosomal abnormality
Laser Irradiation	IS A	Treatement
Leucocyte	IS A	Cell
Leukaemia	IS A	Cancer
Leukaemia Cell	IS A	Cell
Lipoic	IS A	Acid
Liquid Suspension	IS A	Culture
Lymphocytic Leukaemia	IS A	Leukaemia
M7	IS A	Leukaemia
Mammalian Cell	IS A	Cell
Maternal Exposure	IS A	Environment Exposure
M-CSF	IS A	Estimulation Factor
mdr1	IS A	Gene Expression
Megakyocity	IS A	Marker

Mitocondrial Marker	IS A	Marker
Monocytic Leukaemia (AML)	IS A	Leukaemia
Mother	IS A	Parent
MRNA	IS A	RNA
Mrp	IS A	Gene Expression
MTAP	IS A	Gene
Myelocytic Leukaemia	IS A	Leukaemia
NBT Cell	IS A	Leukaemia Cell
NFkB	IS A	Trans Factor
NHLymphoma	IS A	Cancer
Nocodazole	IS A	Substance
Parent	IS A	Person
Paternal Exposure	IS A	Environment Exposure
Patient	IS A	Person
Physician	IS A	Person
PKC isozyme	IS A	Gene Expression
PML-RARAlpha	IS A	Chimaeric Gene
Precursor	IS A	Cell
Progenitor	IS A	Cell
Promyelocytic Leukaemia	IS A	Leukaemia
Radiation	IS A	Cause
RAR-Alpha-PML	IS A	Chimaeric Gene
Red Blood Cell	IS A	Cell
Subclone 707BUF	IS A	Mammalian Cell
Therapy	IS A	Treatement
Theta	IS A	PKC isozyme
Thiobarbituric	IS A	Acid
TMD	IS A	Disease
Topoisomerase	IS A	Gene Expression
Topoisomerase I	IS A	Topoisomerase
Topoisomerase II alpha	IS A	Topoisomerase
Topoisomerase II beta	IS A	Topoisomerase
Transplantation	IS A	Treatement
Transretionic Acid	IS A	Acid
TRNA	IS A	RNA
Trysomy	IS A	Karyotipic Aberration
USA	IS A	Nation
VDJRecombinase	IS A	Substance

Virus	IS A	Cause
Vit	IS A	Antioxidant
X-Ray Exposure	IS A	Exposure

Type “part-of” (totalling 52)

Acid	PART OF	Diet
Allel	PART OF	Gene
Aminoacid	PART OF	Protein
Antigen	PART OF	Membrane
Antioxidant	PART OF	Diet
Back	PART OF	Patient
Blood	PART OF	Patient
Blood Vessel	PART OF	Patient
Bone Marrow	PART OF	Patient
Carbon	PART OF	Aminoacid
Causes	PART OF	Leukaemia Study
Cell	PART OF	Patient
Cell	PART OF	Cluster
Chimaeric Gene	PART OF	Gene
Chromosome	PART OF	Cell
Culture	PART OF	Leukaemia Study
Diaphragm	PART OF	Patient
Diet	PART OF	Leukaemia Study
Disease	PART OF	Leukaemia Study
DNA	PART OF	Gene
Estimulation Factor	PART OF	Culture
EVI-1	PART OF	Gene
Exons	PART OF	Gene
Gene	PART OF	Chromosome
Gene Expression	PART OF	Gene
History	PART OF	Leukaemia Study
Hydrogen	PART OF	Aminoacid
Lympho Glands	PART OF	Lymphotic System
Lymphotic System	PART OF	Patient
Marker	PART OF	Leukaemia Study
Membrane	PART OF	Cell
Nation	PART OF	World

Oxygen	PART OF	Aminoacid
P53	PART OF	Gene
P65	PART OF	NFkB
Patient	PART OF	Group
Person	PART OF	World
Person	PART OF	Leukaemia Study
PLZF	PART OF	Gene
Potential	PART OF	Treatement
Protein	PART OF	Cell
RNA	PART OF	Gene
Sample	PART OF	Leukaemia Study
Skin	PART OF	Patient
Spleen	PART OF	Patient
Substances	PART OF	Leukaemia Study
Sulfur	PART OF	Aminoacid
Symptoms	PART OF	Leukaemia Study
Trans Factor	PART OF	HL-60 Cell
Treatement	PART OF	Leukaemia Study
Truck	PART OF	Patient
Tumor	PART OF	Patient

Axioms (totalling 9)

$\forall x : LaserIrradiation \to x.doses \geq 0$

$\forall x : LaserIrradiation \to x.Frecuency \geq 0$

$\forall x : LaserIrradiation \to x.WaveLength \geq 0$

$\forall x : LaserIrradiation \to x.Power \geq 0$

$\forall x : Patient, y : Date / Death(x, y) \Rightarrow y \leq \ CurrentDate$

$\forall x : Patient \to x.remission \geq 0$ $\forall x : chromosome \Rightarrow x.number \in (1.\,23\ )$

$\forall x : Child \Rightarrow x.age > 0$ $\forall x : Patient \Rightarrow x.age > 0$