ConNeKTion: A Tool for Exploiting Conceptual Graphs Automatically Learned from Text

F. Leuzzi[1], S. Ferilli[1,2], and F. Rotella[1]

[1] Dipartimento di Informatica – Università di Bari
{fulvio.rotella, stefano.ferilli, fabio.leuzzi}@uniba.it
[2] Centro Interdipartimentale per la Logica e sue Applicazioni – Università di Bari

Abstract. Studying, understanding and exploiting the content of a digital library, and extracting useful information thereof, require automatic techniques that can effectively support the users. To this aim, a relevant role can be played by concept taxonomies. Unfortunately, the availability of such a kind of resources is limited, and their manual building and maintenance are costly and error-prone. This work presents ConNeKTion, a tool for conceptual graph learning and exploitation. It allows to learn conceptual graphs from plain text and to enrich them by finding concept generalizations. The resulting graph can be used for several purposes: finding relationships between concepts (if any), filtering the concepts from a particular perspective, keyword extraction and information retrieval. A suitable control panel is provided for the user to comfortably carry out these activities.

1 Introduction

The spread of the electronic technology has had a dramatic impact on the production of documents in all fields of knowledge, and has led to the flourishing of document repositories aimed at supporting scholars and non-technically aware users in carrying out their tasks and satisfying their information needs. However the study, understanding and exploitation of the content of a digital library, and the extraction of useful information thereof, are complex activities requiring automatic techniques that can effectively support the users. In this landscape, a relevant role can be played by concept taxonomies that express both common sense and domain-specific information, including implicit relationships among the concepts underlying the collection. Unfortunately, the availability of such a kind of resources is limited, and their manual building and maintenance are costly and error-prone. A possible solution is the exploitation of Natural Language Processing (NLP) techniques, that work on the textual parts of the documents to extract the concepts and relationships expressed by words. Although the task is not trivial, due to the intrinsic ambiguity of natural language and to the huge amount of required common sense and linguistic/conceptual background knowledge, even small portions of such a knowledge may significantly improve understanding performance, at least in limited domains. This work presents ConNeKTion (acronym for 'CONcept NEtwork for Knowledge representaTION'), a tool for conceptual graph learning and exploitation. It allows to learn conceptual graphs[1] from plain text and to enrich them by finding concept generalizations. The resulting graph can be used for several purposes: finding relationships between concepts (if any), filtering the concepts from a particular perspective, keyword extraction and information retrieval. Specifically, this paper focuses on the graphical control panel provided to the user for exploiting the various functionalities, while technical details and evaluation of the single funcionalities have been already presented in [8, 14, 22].

The paper is organized as follows: the next section describes related works that have some connection to the present proposal, described in Section 3; then, Section 4 describes the tool aimed at supporting end users in managing and exploiting the learned conceptual graph. In the last section, some considerations and future work issues are proposed.

2 Related Work

A primary task in this work is the construction of a conceptual graph starting from an analysis of the plain text contained in the documents that make up a collection. Several techniques are present in the literature generally aimed at building some kind of graph-like structure, that have made the basis on which the state of the art specifically aimed at building taxonomies and ontologies from text has built its approaches. [1] builds concept hierarchies using Formal Concept Analysis: objects are grouped using algebraic techniques based on their descriptive attributes, which are determined from text linking terms with verbs. Different approaches are also available. [16, 15] build ontologies by labeling taxonomic relations only; in our opinion, also non-taxonomic relationships are very important to improve text understanding, such as those associated to actions (and expressed by verbs). [19] builds taxonomies considering only concepts that are present in a domain but do not appear in others; however, one might be interested in collecting and organizing all concepts that can be recognized in a collection, because generic ones may help to frame and connect domain-specific ones. [18] defines a language to build formal ontologies, but this level is very hard to be effectively reached, so a sensible trade-off between expressive power and practical feasibility might better focus on working at the lexical level (at least in the current state of the art).

Our proposal to learning conceptual graphs from text relies on pre-processing techniques taken from the field of NLP, that provide a formal and structured representation of the sentences on which the actual graph construction and reasoning operators can be applied. Due to the wide range of tools available for English, compared to other languages, we will focus on this language in the following. As regards the syntactic analysis of the input text, the *Stanford Parser* and *Stanford Dependencies* [13, 2] are two well-known tools that can identify the

[1] We use the term ‘conceptual graph’ as a synonym for ‘concept network’, with no reference to Sowa’s formalism.

most likely syntactic structure of sentences (including active/passive and positive/negative forms), and specifically their 'subject' or '(direct/indirect) object' components. They also normalize the words in the input text using lemmatization instead of stemming in order to preserve the grammatical role of the original word (and improve readability by humans).

Also, we need in some steps of our technique to assess the similarity among concepts in a given conceptual taxonomy. A classical, general measure, is the *Hamming distance* [9], that works on pairs of equal-length vectorial descriptions and counts the number of changes required to turn one into the other. Other measures, specific for conceptual taxonomies, are [7] (that adopts a global approach based on the whole set of super-concepts) and [29] (that focuses on a particular path between the nodes to be compared).

Another technology we use is ProbLog [20] to apply probabilistic reasoning on the extracted knowledge. It is essentially Prolog where all clauses are labeled with the probability that they are true, that in turn can be extracted from large databases by various techniques. A ProbLog program $T = \{p1 : c1, ..., pn : cn\}$ specifies a probability distribution over all its possible non-probabilistic subprograms according to the theoretical basis in [27]. The semantics of ProbLog is then defined by the success probability of a query, which corresponds to the probability that it succeeds in a randomly sampled program. Indeed, the program can be split into a set of labeled facts $p_i :: f_i$, meaning that f_i is a fact with probability of occurrence p_i, and a Prolog program using those facts, which encodes the background knowledge (BK). Probabilistic facts correspond to mutually independent random variables, which together define a probability distribution over all ground logic programs $L \subseteq L_T$ (where L_T is the set of all f_i's):

$$P(L|T) = \prod_{f_i \in L} p_i \prod_{f_i \in L_T \setminus L} (1 - p_i)$$

In this setting we will use the term *possible world* to denote the least Herbrand model of a subprogram L together with BK, and we will denote by L both the set of sampled facts and the corresponding world.

A possible exploitation of the learned conceptual graph is for Information Retrieval (IR) purposes, so a quick overview of this field of research may be useful as well. Most existing works that tackle the IR problem are based on the so-called *Vector Space Model* (VSM), originally proposed in [26]. This approach represents a corpus of documents D, and the set of terms T appearing therein, as a $T \times D$ matrix, in which the (i, j)-th cell reports a weight representing the importance of the i-th term in the j-th document (usually computed according to both the number of its occurrences in that document and its distribution in the whole collection). Most similarity approaches [11, 25, 24] and weighting schemes [23, 21, 28] have been proposed. Based on this representation, the degree of similarity of a user query to any document in the collection can be computed, simply using any geometric distance measure (e.g., the cosine measure) on that space. One limitation of these approaches is their considering a document only from a lexical point of view, which is typically affected by several kinds of linguistic tricks, such as synonymy and polysemy. More recently, techniques based on dimensionality reduction have been explored with the aim to map both the documents in the corpus and the queries into a lower dimensional space that explicitly takes into account the dependencies between terms, in order to improve the retrieval or categorization performance. Prominent examples are Latent Semantic Indexing [3] (a statistical method based on Singular Value Decomposition that is capable of retrieving texts based on the concepts they contain, not just by matching terms) and Concept Indexing [12] (that exploits Concept Decomposition [4] instead of Singular Value Decomposition).

3 Provided Functionalities

ConNeKTion aims at partially simulating some human abilities in the text understanding and concept formation activity, such as: extracting the concepts expressed in given texts and assessing their relevance; obtaining a practical description of the concepts underlying the terms, which in turn would allow to generalize concepts having similar descriptions; applying some kind of reasoning 'by association', that looks for possible indirect connections between two identified concepts; identifying relevant keywords that are present in the text and helping the user in the retrieval of useful information. The system takes as input texts in natural language, and process them to build a conceptual network that supports the above objectives. The resulting network can be considered as an intensional representation of a collection of documents. Translating it into a suitable First-Order Logic (FOL) formalism allows the subsequent exploitation of logic inference engines in applications that use that knowledge.

3.1 Graph Learning

ConNeKTion exploits a mix of existing tools and techniques, that are brought to cooperation in order to reach the above objectives, extended and supported by novel techniques when needed.

Natural language texts are processed by the Stanford Parser in order to extract triples of the form ⟨*subject*, *verb*, *complement*⟩, that will represent the concepts (the *subject*s and *complement*s) and relationships (*verb*s) for the graph. Some representational tricks are adopted: indirect complements are treated as direct ones by embedding the corresponding preposition into the verb; sentences involving verb 'to be' or nouns with adjectives contributed in building the sub-class structure of the taxonomy (e.g., "the penguin is a bird" yields *is_a*(*penguin*,*bird*)). Specifically, 'is_a' relationships are exploited to build the taxonomy. The representation formalism was enriched by including the sentence's positive or negative form based on the absence or presence (respectively) of a *negation modifier* for the verb in the corresponding syntactic tree. The frequency of each arc between the concepts in positive and negative sentences were taken into account separately. This made our solution more robust, laying the basis for a statistical approach that inspects the obtained taxonomy by filtering out all portions that do not pass a given level of reliability.

The use of generalizations provides many opportunities of enrichment and/or manipulations on the graph. It can be used to build taxonomic structures, also after the addition of new text (possibly causing the presence of new nodes in the graph); to shift the representation, by removing the generalized nodes from the graph and leaving just their generalization (that inherits all their relationships to other concepts); to extend the amount of relationships between concepts belonging to the same connected component of the graph, or to build bridges between disjoint components that open new reasoning paths (which improves the effectiveness of reasoning 'by association'). Given two concepts G and C, G generalizes C if anything that can be labeled as C can be labeled as G as well, but not *vice-versa* [14]. The generalization procedure is made up of three steps: *Concept Grouping*, in which all concepts are grossly partitioned to obtain subsets of concepts (we group similar concepts if the aim is to enrich the relationships, or dissimilar ones in the bridging perspective); *Word Sense Disambiguation*, that associates a single meaning to each term by solving possible ambiguities using the domain of discourse; *Computation of taxonomic similarity*, in which WordNet [5] is exploited in order to further filter with an external source the groups found in step 1, and to choose an appropriate subsumer.

3.2 Reasoning by Association

We intend 'reasoning by association' in a given conceptual graph as the task of finding a path of pairwise related concepts that establishes an indirect interaction between two concepts [14]. Our tool provides two different strategies for doing this: one works in breadth and returns the minimal path (in the number of traversed edges) between concepts, also specifying all involved relations; the other works in depth and allows to answer probabilistic queries on the conceptual graph.

In more details, the former strategy looks for a minimal path starting two *Breadth-First Search* (BFS) procedures, one for each concept under consideration, until their boundaries meet. It also provides the number of positive/negative instances, and the corresponding ratios over the total, in order to express different gradations (such as permitted, prohibited, typical, rare, etc.) of actions between two objects. While this value does not affect the reasoning strategy, it allows to distinguish which reasoning path is more suitable for a given task. Note that this is different than the standard *spreading activation* algorithm, in that (1) we do not impose weights on arcs (we just associate arcs with symbolic labels expressing their semantics) nor any threshold for graph traversal, (2) we focus on paths rather than nodes, and specifically we are interested in the path(s) between two particular nodes rather than in the whole graph activation, hence (3) it makes no sense in our approach setting the initial activation weight of start nodes, and (4) this allows us to exploit a bi-directional partial search rather than a mono-directional complete graph traversal.

Since real world data are typically noisy and uncertain, the latter strategy was included, that softens the classical rigid logical reasoning. This is obtained by suitably weighting the arcs/relationships among concepts to represent their likelihood among all *possible worlds*, and using these weights to prefer some paths over others. ProbLog [20] is exploited for this purpose, whose descriptions are based on the formalism $p_i :: f_i$ where f_i is a ground literal having probability p_i. In our case, f_i is of the form *link*(*subject*, *verb*, *complement*) and p_i is the ratio between the sum of all examples for which f_i holds and the sum of all possible links between *subject* and *complement*. Again, this is different than *spreading activation* because the ProbLog strategy is adopted.

3.3 Keyword Extraction

The identification of relevant nodes in the graph may in some sense correspond to selecting keywords that provide indications on the main topics treated in the collection. As a first step, the frequency of each term is computed (its spread through the collection is ignored, to allow the incremental addition of new texts without the need of recomputing this statistics). Then, the EM clustering approach provided by Weka based on the Euclidean distance is applied to row vectors (representing concepts in the graph). Finally, various Keyword Extraction techniques, based on different (and complementary) aspects, perspectives and theoretical principles, are applied on the input texts to identify relevant concepts. We mixed a quantitative approach based on co-occurrences [17], a qualitative one based on WordNet [6] and a novel psychological one based on word positions. Assuming that humans tend to place relevant terms/concepts toward the start and end of sentences and discourses, where the attention of the reader/listener is higher [10], this approach determines the chance of a term being a keyword based on its position in the sentence/discourse. In particular, a mixture model determined by two Gaussian curves, whose peaks are placed around the extremes of the portion of text to be examined, is used. The outcomes of these techniques are exploited to compute a compound *Relevance Weight* for each node in the network. Then, nodes are ranked by decreasing Relevance Weight, and a suitable cut-point in the ranking is determined to distinguish relevant concepts from irrelevant ones. We cut the list at the first item in the ranking such that the difference in relevance weight from the next item is greater or equal than the maximum difference between all pairs of adjacent items, smoothed by a user-defined parameter $p \in [0, 1]$ [8].

3.4 Information Retrieval

After the set of representative keywords for each document has been obtained, it can be considered as a higher-level representation of the digital library's content, and hence keyword extraction also work as a pre-processing step toward Information Retrieval in the library itself [22]. Indeed, to each keyword a corresponding meaning can be associated as follows: each keyword in the document is mapped to a corresponding synset (i.e., the code of a concept) in WordNet, that is taken as its semantic representative, using Word Sense Disambiguation techniques [8]. The output such a step, for each document, is a list of pairs, consisting of keywords and their associated synsets. All these synsets are partitioned into different groups using pairwise clustering. Then, each document is considered in turn, and each of its keywords 'votes' for the cluster to which the associated synset has been assigned. The aim is finding groups of similar synsets that might be usefully exploited as a kind of 'glue' binding together subsets of documents that are consistent with each other. In this perspective, the obtained clusters can be interpreted as intensional representations of specific domains, and thus they can be exploited to retrieve the sub-collection they are associated to. In this setting, a query in natural language is processed in order to recognize the relevant terms, and consequently find the the corresponding synsets. At this point, a similarity evaluation (using the function in [6]) is performed against each cluster (that has a list of associated documents). The best result is used to obtain the list of documents by descending relevance, that can be used as an answer to the user's search.

4 Exploitation Tool

The above functionalities are delivered to the users through a graphical tool that provides a set of controls allowing to explore and analyze the conceptual graph. The learned net is represented through an XML file. The tool can load a file in this format, and draw the corresponding net (automatically organizing the nodes in the best possible way). Different colors are used for nodes depending by their type: subjects and complements have a different color than verbs. Also the relations are filled with a different color depending on the positive or negative valence of the corresponding phrase. Figure 1 shows two screenshots of the main interface of the tool, showing two different perspectives on the same net (an complete overview and a selection thereof, respectively). The main area, on the left, contains the net.

After loading a conceptual graph, the tool allows to explore it in a graphical intuitive way, using classical mouse-based controls. Since the compound view of the whole graph is typically cluttered and very dense of (often overlapping) nodes and edges (but still useful to grasp the overall shape of the net), scroll, pan and zoom in/out controls allow to focus on specific parts thereof and to have a better insight on them. Single nodes can be dragged as well, and the entire net is automatically rearranged accordingly to best fit the available space. Finally, by selecting a specific node, it is possible to set a neighborhood limit such that all the nodes whose shortest path to the selected node are outside the selected level are filtered out.

All the controls, settings and results are placed in a panel standing on the right of the graph visualization window. Such a panel is in turn divided into several sub-parts (shown in Figure 2). Let us examine the single sub-areas of the control panel in more details. **Distance hops** is in the top part of the panel (shown on the left in Figure 2) and containing a text field in which the user can enter the

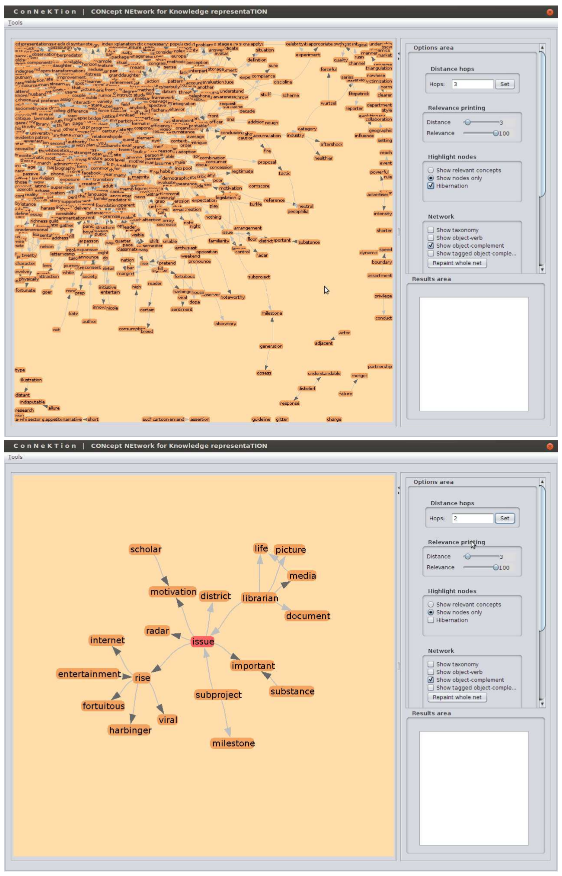
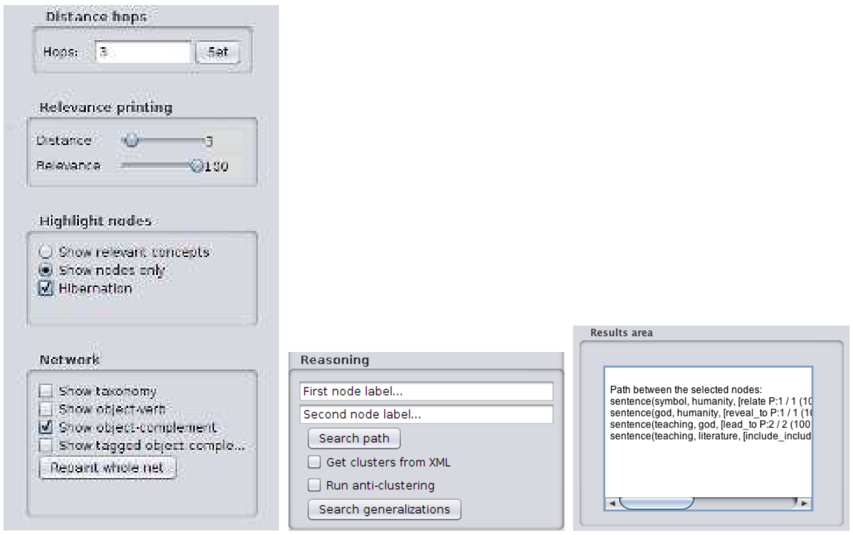


Fig. 1. The main control panel

desired level up to which nodes are to be shown, starting from the selected one (i.e. the user can set the maximum neighborhood level). **Relevance filtering** contains two sliding bars: the former is *Distance*, it is aimed to providing the same functionality as the *Distance hops* area, but bound in $[0, maxHops(net)]$, where $maxHops(\cdot)$ is a function that returns the diameter of a given net; the latter is *Relevance*, it allows to set the relevance parameter that determines which relevant nodes are to be highlighted. **Highlight nodes** is a radio button, it allows to select a visualization that highlights relevant nodes or a classical one. Choosing the relevant nodes perspective enables access to the advanced functionality of relevant node recognition, useful for a deeper analysis of the collection. Moreover, using the *Hibernation* check box the study of the net is made more comfortable for the human reader. This issue may require further clarification. In standard mode, the research for a balanced placement of nodes within the used space is always 'alive', so that the nodes automatically rearrange their position


Distance hops

Hops: 3 Set

Relevance printing

Distance 3

Relevance 100

Highlight nodes

( ) Show relevant concepts
(●) Show nodes only
[✓] Hibernation

Network

[ ] Show taxonomy
[ ] Show object-verb
[✓] Show object-complement
[ ] Show tagged object comple...

Repaint whole net

Reasoning

First node label...

Second node label...

Search path

[ ] Get clusters from XML

[ ] Run anti-clustering

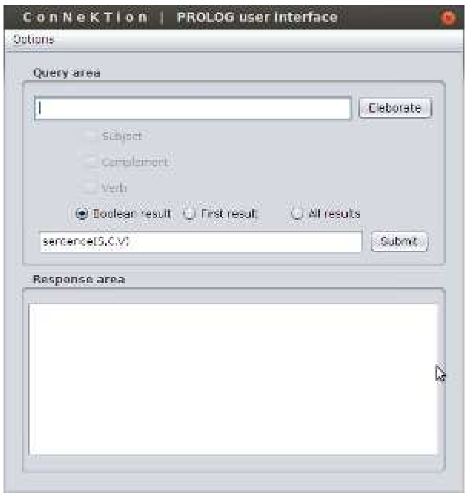
Search generalizations

Results area

```
Path between the selected nodes:
sentence(symbol, humanity, [relate P:1 / 1 (10
sentence(god, humanity, [reveal_to P:1 / 1 (1(
sentence(teaching, god, [lead_to P:2 / 2 (100
sentence(teaching, literature, [include_includ
```

Fig. 2. (left) Set of parameters placed in the right panel - (center) Reasoning operators parameters - (right) Textual area dedicated to results

in the screen after the perturbations introduced by the user when he moves some elements to study them more comfortably. Since the continuous movement of the nodes makes the visual analysis of the net difficult, the possibility to stop the net in order to explore it (through reading and manual rearrangements of single nodes) was introduced. **Network** embeds four options, and specifically: *Show taxonomy*, that adds taxonomic relations to the network; *Show object-verb*, that adds verbs as nodes, and edges $< subject, verb >$ and $< verb, complement >$; *Show object-complement*, that adds direct relations $< subject, complement >$ (regardless of the verbs connecting them); *Show tagged object-complement*, that enables the tagging of the relations $< subject, complement >$ with verbs and associated (positive or negative) valence as reported in the XML file (so, the visual outcome is the same as for *Show object-complement*, but the tagged relations in the XML can be used for further functionalities). **Reasoning** is devoted to the reasoning operators (shown in the middle of Figure 2). In particular, it contains two text fields in each of which a concept (label of a node) can be entered, so that pressing the *Search path* button starts the Reasoning by Association functionality to obtain a plausible complex relation between the specified concepts. This sub-area also contains a button (named *Search generalizations*) that starts the search for *Generalization*; its behavior can be modified by acting on two checkboxes, *Get clusters from XML* (that avoids computing the clusters if they have already been computed and stored in dedicated XML files), and *Run anti-clustering* (that starts the technique to build bridges between different components of the net [14]). **Results** appears in the bottom area in the panel



sentence(S,C,V)

Fig. 3. The PROLOG knowledge base query tool

(shown on the right in Figure 2). It is dedicated to textual results, consisting of paths where each row reports in square brackets (the labels of) the relations that exist between two nodes. In particular, each relation (verb) is associated to the number of positive and negative instances in which it occurred, expressing its valence. This also provides an indication of the degree of reliability of the path sub-parts. As an example, the screenshot in Figure 2 shows the resulting path between nodes 'symbol' and 'literature':

sentence(symbol, humanity, [relate P:1 / 1 (100.0%), N:0 / 1 (0.0%)])
sentence(god, humanity, [reveal_to P:1 / 1 (100.0%), N:0 / 1 (0.0%)])
sentence(teaching, god, [lead_to P:2 / 2 (100.0%), N:0 / 2 (0.0%)])
sentence(teaching, literature, [include_including P:4 / 4 (100.0%), N:0 / 4 (0.0%)])

which can be interpreted as: "Humanity can relate by means of symbols. God reveals to humanity. Teaching (or education) leads to God, and includes the use of literature.". Here only one relation per row is present, and there are no sentences with negative valence.

Finally, an additional functionality concerns the possibility of querying the ProLog knowledge base expressing the content of the net, which allows more complex kinds of reasoning than simple reasoning by association on the graph. It can be accessed from menu *Tools* in the main window, using the *PROLOG user interface* item. A window like that in Figure 3 is opened, that allows to enter a ProLog query to be answered using the knowledge base (e.g., "what does a dog eat?" might be asked in the form *eat(dog,X)*). The ProLog representation of the net can be obtained and saved from the same window, by choosing the *Create new K.B.*) item in the *Options* menu.

5 Conclusions

Studying, understanding and exploiting the content of a digital library, and extracting useful information thereof, are complex and knowledge-intensive activities for which the user needs the support of effective automatic techniques. To this aim, a relevant role can be played by concept taxonomies. Unfortunately, the availability of such a kind of resources is limited, and their manual building and maintenance are costly and error-prone. ConNeKTion is a tool that allows to learn conceptual graphs from plain text and to enrich them by finding concept generalizations. The resulting graph can be used for several purposes: finding relationships between concepts (if any), filtering the concepts from a particular perspective, keyword extraction and information retrieval. A suitable control panel is provided for the user to comfortably carry out these activities.

As future work, we plan to improve the natural language text pre-processing using anaphora resolution in order to replace, where possible, pronouns with the explicit concept they express. We also wish to extend the reasoning operators by adding an argumentation operator, that could exploit probabilistic weights, intended as a rate of reliability, to provide support or attack to a given statement.

References

[1] P. Cimiano, A. Hotho, and S. Staab. Learning concept hierarchies from text corpora using formal concept analysis. *J. Artif. Int. Res.*, 24(1):305–339, August 2005.

[2] M.C. de Marneffe, B. MacCartney, and C. D. Manning. Generating typed dependency parses from phrase structure trees. In *LREC*, 2006.

[3] S. Deerwester. Improving Information Retrieval with Latent Semantic Indexing. In Christine L. Borgman and Edward Y. H. Pai, editors, *Proceedings of the 51st ASIS Annual Meeting (ASIS '88)*, volume 25, Atlanta, Georgia, October 1988. American Society for Information Science.

[4] I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. In *Machine Learning*, pages 143–175, 2001.

[5] C. Fellbaum, editor. *WordNet: An Electronic Lexical Database.* MIT Press, Cambridge, MA, 1998.

[6] S. Ferilli, M. Biba, T.M. Basile, and F. Esposito. Combining qualitative and quantitative keyword extraction methods with document layout analysis. In *Post-proceedings of the 5th Italian Research Conference on Digital Library Management Systems (IRCDL-2009)*, pages 22–33, 2009.

[7] S. Ferilli, M. Biba, N. Di Mauro, T.M. Basile, and F. Esposito. Plugging taxonomic similarity in first-order logic horn clauses comparison. In *Emergent Perspectives in Artificial Intelligence*, Lecture Notes in Artificial Intelligence, pages 131–140. Springer, 2009.

[8] S. Ferilli, F. Leuzzi, and F. Rotella. Cooperating techniques for extracting conceptual taxonomies from text. In *Proceedings of The Workshop on MCP at AI*IA XIIth Conference*, 2011.

[9] R.W. Hamming. Error detecting and error correcting codes. *Bell System Technical Journal*, 29(2):147–160, 1950.

[10] R. Jay and A. Jay. *Effective Presentation: How to Create and Deliver a Winning Presentation.* Prentice Hall, 2004.

[11] W. P. Jones and G. W. Furnas. Pictures of relevance: A geometric analysis of similarity measures. *Journal of the American Society for Information Science*, 38(6):420–442, 1987.

[12] G. Karypis and Eui-Hong (Sam) Han. Concept indexing: A fast dimensionality reduction algorithm with applications to document retrieval and categorization. Technical report, IN CIKM00, 2000.

[13] D. Klein and C. D. Manning. Fast exact inference with a factored model for natural language parsing. In *Advances in Neural Information Processing Systems*, volume 15. MIT Press, 2003.

[14] F. Leuzzi, S. Ferilli, C. Taranto, and F. Rotella. Improving robustness and flexibility of concept taxonomy learning from text. In *Proceedings of The Workshop on NFMCP at ECML-PKDD 2012 Conference*, 2012.

[15] A. Maedche and S. Staab. Mining ontologies from text. In *EKAW*, pages 189–202, 2000.

[16] A. Maedche and S. Staab. The text-to-onto ontology learning environment. In *ICCS-2000 - Eight ICCS, Software Demonstration*, 2000.

[17] Yutaka Matsuo and Mitsuru Ishizuka. Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools*, 13:2004, 2003.

[18] N. Ogata. A formal ontology discovery from web documents. In *Web Intelligence: R.D., First Asia-Pacific Conf. (WI 2001)*, number 2198 in Lecture Notes on Artificial Intelligence, pages 514–519. Springer-Verlag, 2001.

[19] A. Cucchiarelli P. Velardi, R. Navigli and F. Neri. Evaluation of OntoLearn, a methodology for automatic population of domain ontologies. In *Ontology Learning from Text: Methods, Applications and Evaluation.* IOS Press, 2006.

[20] L. De Raedt, A. Kimmig, and H. Toivonen. Problog: a probabilistic prolog and its application in link discovery. In *In Proceedings of 20th IJCAI*, pages 2468–2473. AAAI Press, 2007.

[21] S. E. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, and M. Gatford. Okapi at trec-3. In *TREC*, pages 0–, 1994.

[22] F. Rotella, S. Ferilli, and F. Leuzzi. A domain based approach to information retrieval in digital libraries. In *Digital Libraries and Archives - 8th Italian Research Conference IRCDL 2012 - Revised Selected Papers*, volume 354 of *CCIS*, 2012.

[23] G. Salton. *The SMART Retrieval System—Experiments in Automatic Document Processing.* Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1971.

[24] G. Salton. Automatic term class construction using relevance–a summary of work in automatic pseudoclassification. *Inf. Process. Manage.*, 16(1):1–15, 1980.

[25] G. Salton and M. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill Book Company, 1984.

[26] G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Commun. ACM*, 18:613–620, November 1975.

[27] T. Sato. A statistical learning method for logic programs with distribution semantics. In *Proceedings of the 12th ICLP 1995*, pages 715–729. MIT Press, 1995.

[28] A. Singhal, C. Buckley, M. Mitra, and A. Mitra. Pivoted document length normalization. pages 21–29. ACM Press, 1996.

[29] Z. Wu and M. Palmer. Verbs semantics and lexical selection. In *Proceedings of the 32nd annual meeting on ACL*, pages 133–138, Morristown, NJ, USA, 1994. ACL.