Using WordNet for Word Sense Disambiguation to Support Concept Map Construction[1]

Alberto J. Cañas[1], Alejandro Valerio[1], Juan Lalinde-Pulido[1,2], Marco Carvalho[1], Marco Arguedas[1]

[1]Institute for Human and Machine Cognition
40 South Alcaniz St., Pensacola, FL 32502
{acanas, marvalho, avalerio, marguedas}@ihmc.us
www.ihmc.us

[2]Universidad EAFIT
Medellín, Colombia
jlalinde@eafit.edu.co
www.eafit.edu.co

Abstract. The construction of a concept map consists of enumerating a list of concepts and —a more difficult task— determining the linking phrases that should connect the concepts to form meaningful propositions. Appropriate word selection, both for concepts and linking phrases, is key for an accurate knowledge representation of the user's understanding of the domain. We present an algorithm that uses WordNet to disambiguate the sense of a word from a concept map, using the map itself to provide its context. Results of preliminary experimental evaluations of the algorithm are presented. We propose to use the algorithm to (a) enhance the "understanding" of the concept map by modules in the CmapTools software that aide the user during map construction, and (b) sort the meanings of a word selected from a concept map according to their relevance within the map when the user navigates through WordNet's hierarchies searching for more appropriate terms.

1. Introduction

Concept mapping is a process of meaning-making. It implies taking a list of *concepts* – a concept being a perceived regularity in events or objects, or records of events or objects, designated by a label [1], – and organizing it in a graphical representation where pairs of concepts and linking phrases form propositions. Hence, key to the construction of a concept map is the set of concepts on which it is based. Coming up with an initial list of concepts to include in a map is really just an issue of retrieving from long-term memory. In fact, rote learners are particularly good at listing concepts.

 Paper presented at SPIRE 2003 – 10th International Symposium on String Processing and Information Retrieval, October 2003, Manaus, Brazil.

A more difficult task during concept map construction is finding the "linking phrase" that appropriately expresses the relationship between two concepts to form a meaningful proposition.

Often, while constructing a concept map, users –whether elementary school students, scientists or other professionals– pause and wonder what additional concepts they should include in their map, or what words to use to clearly express the relationship

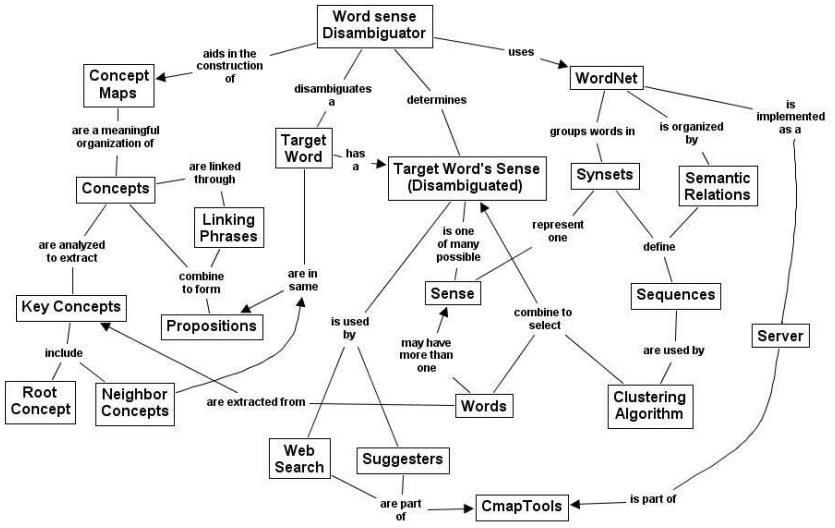

Fig. 1. A concept map on word sense disambiguating during concept map construction

between two concepts. Even though they know well the domain they are modeling, they cannot "remember" what other concepts are relevant, can't think of the "right word", or sometimes they need to "refresh" their knowledge about a particular sub-domain of the concept map.

At the Institute for Human and Machine Cognition (IHMC) we have developed CmapTools [2, 6], a widely-used software program that supports the construction of concept maps, as well as the annotation of the maps with additional material such as images, diagrams, video clips and other such resources. It provides the capability to store and access concept maps on multiple servers to support knowledge sharing across geographically-distant sites.

This paper describes an effort to use WordNet[13] to disambiguate the sense of words in concept maps, whether they are part of a concept or a linking phrase. By exploiting the topology and semantics of concept maps, the algorithm tries to determine which of the senses in WordNet best matches the context of the concept map. If effective, word disambiguation could then be used by the other tools to more precisely search

the Web and CmapTools servers. Additionally, a WordNet server is being implemented that allows the user to lookup words and browse through the broad information that WordNet provides as an aide during concept mapping.

This paper begins with a short description of concept mapping. It then presents CmapTools, and the concept mapping aides that would take advantage of word disambiguation. Section 4 describes WordNet within the context of word disambiguation. In Section 5 we present the algorithm used to disambiguate words in a concept map. Finally, results from an experiment where we compare the word sense that the algorithm recommends with that of subjects is presented and discussed in Sections 6-8. Figure 1 shows a concept map summarizing the purpose and function of word disambiguating during concept mapping.

2. Concept Maps and Concept Mapping

Concept maps, developed by Novak [1], are tools for organizing, representing and sharing knowledge, and were specifically designed to tap into a person's cognitive structure and externalize concepts and propositions. A concept map is a two-dimensional representation of a set of concepts constructed so that the interrelationships among them are evident.

From the education perspective, there is a growing body of research that indicates that the use of concept maps can facilitate meaningful learning. During concept map construction, meaning making occurs as the learner makes an effort to link the concepts to form propositions. Additionally, concept maps have been demonstrated to be an effective means of representing and communicating knowledge during the construction of expert systems [3] and performance support systems [4] or as means of capturing and sharing experts' knowledge [5],

3. CmapTools and Concept Mapping Aides

Software programs like CmapTools make it easier for users to construct and share their knowledge models based on concept maps. In CmapTools we have extended the use of a concept maps to serve as the browsing interface to a domain of knowledge. The program facilitates the linking of a concept to other concept maps, pictures, images, audio/video clips, text, Word documents, Web pages, etc., as a means to provide access to auxiliary information on the concept. The software is based on a client-server architecture, which allows the linked media resources and concept maps to be located anywhere on the Internet.

In collaboration with D. Leake, A. Maguitman, and T. Reichherzer from Indiana University, we have developed a number of methods to aide the user during the process of construction of concept maps. These aides are based on the following

observations: users often stop and wonder what other concepts they should add to the concept map they are working on; frequently, they spend time looking for the right word to use in a concept or linking phrase; they search for other concept maps that may be relevant to the one they constructing; they spend time searching through the Web for resources (Web pages, images, movies, etc.) that could be linked to their concept maps; and they search through the Web looking for additional material that could help them enhance their maps. The methods developed analyze a concept map under construction and seek useful information from both distributed concept maps and from the Web. For this, we have developed retrieval methods to exploit the semantics, topology, and context of concept maps for concept map indexing and retrieval, using methods such as topological analysis [7] to summarize structural characteristics, latent semantic analysis [8] to identify topics, and specially-developed indexing methods to capture relationships between concepts.

During concept map construction, the methods will proactively mine the web to suggest concepts that could enhance the map [9] and suggest topics for new concept maps that would complement the one being built [11], and suggest propositions and other concept maps from CmapServers that are relevant to the map being constructed [10]. Additionally, the user can, on-demand, search for concept maps, other resources, and Web pages that are relevant to the map [12].

4. WordNet

WordNet is a freely available lexical database for English whose design is inspired by current psycholinguistic theories of human lexical memory [13]. English words are organized into synonym sets, so-called synsets, and each representing one underlying lexical concept. A synset may have many words (synonyms) and one word can be a member of many synsets, one for each different sense. Relations between synsets are semantical relationships and relations between words are lexical relationships. WordNet represents both.

The literature shows that WordNet has been used successfully in word sense disambiguation algorithms in other contexts, particularly text. Li *et al.* [14] report using it as the source information for disambiguation with correct solutions up to 57% using only the sense ranked as first and 67% when considering the top two senses. Mihalcea and Moldovan[15] report better results when WordNet is combined and cross-checked with other sources, improving up to 92% when the algorithm is allowed not to give an answer when the confidence is low [16]. When using a small but representative set of words to determine the context, Nastase and Szpakowics [17] obtained an average 82% accuracy when allowing the algorithm not to give an answer.

5. Disambiguating Word Sense in Concept Maps with WordNet

The algorithm presented in this paper tries to resolve the correct sense of a polysemic (multiple meaning) word, using a concept map as its context. The selection of the appropriate words from the concept map to be used in the algorithm is crucial. The algorithm exploits the topology of the map, by including only the words of key concepts as part of the disambiguation process. Other algorithms based on text analysis (e.g. [18]) have the problem of selecting the key words, which is often difficult because there is no particular structure, and the relation between the words is not clear. We use the senses and semantic relations provided by WordNet to perform the disambiguation.

Description

The algorithm starts by selecting key concepts from the map which will be included in the process of determining the sense of a word *w*. Once these concepts are selected, the senses of the words within the concepts are found using WordNet after applying morphological transformations where needed.

The synsets are clustered using the hypernym distance based on WordNet's hypernym relation in such a way that only one synset per word is allowed in each cluster. Several clusters will result, each with a different weight depending on the number of words in the cluster and the hypernym distance. The cluster with the highest weight that contains a synset *s* of *w*, is the selected cluster, and *s* is chosen as the sense of *w*.

Step 1. Selection of key concepts

The topology of the map presents a strong aide in determining the key words. Based on it, these are the selected words: (a) Words in concepts with two linking phrase distance from the concept where *w* is found. That is, words in concepts that are in the same proposition as *w*; (b) Words in the root concept of the map. (The root concept of the map is usually a good representation of the overall topic of the map); (c) Other words in the concept to which *w* belongs. (Words within the same concept have a strong relation between them, therefore there words are included). These criteria determine the words to be used in the following steps.

Step 2. Relating words to synsets

A synset is the set of synonym words representing a concept in WordNet. Therefore, each word belongs to one or more synsets (in case of a polysemic word). In order to relate the words to the WordNet collection, we use a variation of the original morphological transformation proposed by the WordNet team in Princeton [13], making some additional validations to remove stop words and a stronger suffix and

prefix analysis. At the end of this step, each word is related to the set of synsets to which it belongs.

Step 3. Hypernym sequences creation

Once the set of synsets for each word have been found, we construct all the possible hypernym sequences whose last element is a synset in one of those sets. We call a hypernym sequence an indexed collection of synsets (a list of synsets), in which the n_i element of the sequence is a hypernym of the n_{i+1} element, and n_0 is a synset with no hypernyms. The hypernym relation is transitive and asymmetrical, so it is guaranteed that there will be no repetitions and no cycles in the hypernym sequences. In WordNet, a synset can have more than one hypernym, so there can be more than one hypernym sequence for a synset. Now we have all hypernym sequences for all words participating on the process.

Step 4. Cluster creation

For implementation purposes, an optimization is done at this point, sorting the sequences in such a way that sequences with the largest common prefix are together. This is important to reduce the cluster construction time.

With the set of sequences, the cluster creation step follows. In the context of the algorithm we define cluster as a tuple (*C, l, S*), where *C* is a hypernym sequence, *l* is a positive integer, *S* is the set of hypernym sequences belonging to the cluster and all elements of *S* have its first *l* elements equal to the first *l* elements of *S*.

For each sequence *q,* whose last element is a synset *s* that contains *w*, we calculate the possible clusters using *q* as centroid. We begin creating the first cluster which is formed by (*q*, *lengthOf* (*q*), {*q*}) and is added as to the resultant clusters. Now an iterative procedure begins: We create a new cluster grouping those sequences with *lengthOf* (*q*)-1 elements in common with *q*, then a cluster with sequences with *lengthOf* (*q*)-2 elements in common with *q*, and so forth until *l* is equal to 1.

Step 5. Best cluster selection

For all the clusters produced in step 4, their weight is calculated. The cluster with the highest weight is selected as the recommended one. In case two or more of them have the maximum weight, they are all selected.

The weight of each cluster is calculated as follows: Given a cluster $H = (C, l, S)$, P_i is the length of the sequence s_i, belonging to *S*, that is not common with C. In other words, P_i tells us in how many elements s_i differ from *C*, and give us a measure of distance between them. So the weight of the cluster *H* is $\frac{1}{\sum P_i}$.

Step 6. Word sense resolution

If there is only one cluster $H = (C, l, S)$ with the maximum weight, then the last synset s of C is the disambiguated sense of the word. If more than one cluster has the maximum weight, then for each of these clusters, the last synset s of C with the maximum frequency of use according to the WordNet collection is selected as the disambiguated sense of w.

An Example

To clarify the algorithm, we will use as an example the concept map in Figure 1. Let's assume the concept to disambiguate is *sense.* The algorithm first selects the words from the root concept and neighboring concepts: *words, target, synset, clustering, algorithm, disambiguator, WordNet, sequences, web, search, key, concept, suggesters, semantic,* and *relations*. Next, it checks whether any of these words does not exist in WordNet, making the morphological transformations. As the algorithm deals with nouns and the hypernym hierarchy, auxiliary WordNet relations are used to transform possible adjectives to nouns which in this case are none. To complete this step, the set of synsets for each word is determined. In the case of the word *sense*, 5 senses are found: 1-(a general conscious awareness), 2-(the meaning of a word or expression), 3-(the faculty through which the external world is apprehended), 4-(sound practical judgment), and 5-(a natural appreciation or ability). In the next step, the clusters are made using the hypernym hierarchy, resulting in 2076 paths constructed, 184 belonging to the word *sense*. This means that from 5 synsets there are 184 possible different routes from one of the *sense*'s synset to a hierarchy root. At this point, the clustering algorithm begins, resulting in 917 clusters with an average 4.9 clusters per path. The cluster with the highest weight is the one formed with paths ending with the following synsets: {*sense*(2), *word*(1), *key*(8), *wordnet*(1)} with a weight of *14.1*. This cluster is selected, with the sense *sense*(2) (the meaning of a word or expression), which is the correct sense of the word in this context.

6. Experimental Procedure

Before proceeding any further with the integration of the algorithm into CmapTools, we examined its effectiveness by running an experiment designed to compare the algorithm's designation of the sense of words from concepts in concept maps with the designation by a group of subjects. We started by asking a person with many years of experience in concept mapping to prepare a collection of 50 "relatively good" concept maps from a public CmapTools server where thousands of concept maps are stored by users from around the world. The maps needed to be in English, and "relatively good", because the server contains all kinds of "concept maps" – some of which have little resemblance to a concept map, consist of just a couple of concepts, or would be unusable for some other reason. Next, we randomly selected 10 concept maps from

this set. For each of these maps, we randomly selected two of the one-word concepts in the map that had more than two senses in the WordNet collection.

For each of the 20 concepts, we printed all the senses that WordNet presents for the word in random order. We presented each concept map with the concept highlighted and the list of senses for the word, with the instructions to the subject to select the sense that was the most relevant for the word in the context of the concept map. We then refined the set of concepts by running the experiment through a small group of subjects with the only intention of eliminating those where they did not agree on the top senses for the word, to eliminate ambiguous concepts in the final selection. In this process, four concepts where dropped. The 16 concepts left where represented to each of 27 subjects, individually, asking them to select the top two senses from the list presented with each concept.

Next, we applied the algorithm to disambiguate each of the words within the context of the concept map from which it was extracted. We then compared, for each word, the sense selected by the subjects with the sense recommended by the algorithm.

7. Experimental Results

For 4 of the 16 concepts, less than 70% of the subjects agreed on the most relevant sense of the concept. These cases where dropped from the analysis because the context of the word within the concept map was not clear, and it would be impossible for the algorithm to agree with the subjects if they didn't agree among themselves..

From the 12 resulting words, the algorithm's proposed sense agreed with the sense selected by the subjects in 9 cases, giving a success rate of 75%.

The average number of concepts in the concept maps is 22.75 concepts, with a standard deviation of 9.91. The average number of concepts used by the algorithm was 9.37, with a standard deviation of 4.15.

8. Discussion

If few subjects agree on what the sense of the word is, it is impossible for the algorithm to select a sense that will be relevant to the subjects. Therefore, those words where less than 70% of the subjects agreed were excluded from the experiment. Additionally, only words with more than two senses were selected in order to eliminate the possibility of the algorithm choosing the correct sense by chance.

The results seem to indicate that it is feasible for the algorithm to obtain a result that matches the sense assigned by the subjects 75% of the time. When compared to similar efforts, the experiment's result is encouraging. Analyzing our results against previous experiments with similar conditions, Li *et al.* [14] obtained 57% correct

solutions working over short text analysis and using 20 words as the size for the text window, compared to the 6-word average used in our algorithm. Nastase *et al*. [17] had 57.27% accuracy using a combined approach with Roget's Thesaurus on disambiguating nouns. Mihalcea and Moldovan [16] reported 92.2% of accuracy in nouns, but they were able to avoid suggesting a sense when the confidence level was low, which would not make sense in our intended application.

Although it is apparent that a reduced number of words can be successfully used to disambiguate the context, the correct selection of the words is crucial for the algorithm to be effective. In the case of a concept map, we exploit the topology of the map itself to define the heuristics by which the set of terms is determined.

However, the algorithm can be easily confused if the neighbor concepts are not part of the word context, which is the case in a poorly constructed map. Even though this has not been formally tested, it will most likely result on the selection of a wrong sense of the word or on constructing clusters with low coherence. Since the intended use of the algorithm always requires an answer, there is not that can be done in this case. A possible approximation that may intuitively work is returning the most common use of the word according to the WordNet collection when the weight of all clusters is under a given threshold.

We are confident that we can improve the algorithm presented by further leveraging on the map's topology and on the type of linking phrases that connect the concept to be disambiguated to other concepts. Further research on this aspect of the algorithm may improve its effectiveness.

9. Conclusions

Key to providing intelligent tools that aide the user in the construction of concept maps, is for the tools to "understand" to the extent possible the context and content of the map being constructed. Elsewhere we have reported on previous research that has shown the feasibility of using the topology and semantics of the concept map itself as the basis to find and propose new concepts, propositions, topics for new concept maps, and relevant Web pages to the user for improvement of the partially built map. In this paper, we presented the possibility of using an algorithm that exploits WordNet to disambiguate the sense of a word that is part of a concept or linking phrase in a concept map. The results shown are encouraging, and suggest more research be done to improve the algorithm. The word-disambiguating algorithm will be used within the CmapTools software suite to (a) provide context that will enhance the understanding of the concept map by other modules in the toolkit, and (b) display the most approximate sense of a word – in the context of the map being constructed-- when the user navigates through the WordNet hierarchies looking for better terms.

References

1. Novak, J. D. and D. B. Gowin, *Learning how to Learn*. NY: Cambridge Univ. Press, 1984.

2. Cañas, A. J., K. M. Ford, J. W. Coffey, T. Reichherzer, N. Suri, R. Carff, D. Shamma, G. Hill, and M. Breedy, Herramientas para Construir y Compartir Modelos de Conocimiento Basados en Mapas Conceptuales, Rev. de Inf. Educativa, Vol. 13, No. 2, pp. 145-158, 2000.

3. Ford, K. M., J. Coffey, A. J. Cañas, E. J. Andrews, C. W. Turner, *Diagnosis and Explanation by a Nuclear Cardiology Expert System*, Int. J. of Expert Systems, 9, 499-506, 1996.

4. Cañas, A. J., J. Coffey, T. Reichherzer, N. Suri, R. Carff, G. Hill, *El-Tech: A Performance Support System with Embedded Training for Electronics Technicians,* Proc. of the 11th FLAIRS, Sanibel Island, Florida, May 1997.

5. Hoffman, R.R., J. W. Coffey, and K. M. Ford, A Case Study in the Research Paradigm of Human-Centered Computing: Local Expertise in Weather Forecasting. *Unpublished Technical Report, National Imagery and Mapping Agency*. Washington, D. C., 2000.

6. Cañas, A. J., G. Hill, R. Carff, N. Suri, *CmapTools: A Knowledge Modeling and Sharing Toolkit,* Tech. Rep. IHMC CmapTools 93-01, Inst. for Human & Machine Cognition, 2003.

7. Kleinberg, J., *Authorative Sources in a Hyperlink Environment*, JACM 46(5),604-632, 1999.

8. Deerwater, S., S. T. Dumai, G.W. Furnas, T. K. Landauer, and R. Harshman, *Indexing by Latent Semantic Snalysis*. J. Am. Soc. Inf. Sci., 41(6), pp. 391-407, 1990.

9. Cañas, A. J., M. Carvalho, M. Arguedas, Mining the Web to Suggest Concepts during Concept Mapping: Preliminary Results, Proceedings of the XIII SBIE, Brazil, 2002.

10. Leake, D. B., A. Maguitman, A. J. Cañas, *Assessing Conceptual Similarity to Support Concept Mapping*, Proc. of the Fifteenth FLAIRS, Pensacola, FL (May 2002).

11. Leake, D., A. Maguitman, and T. Reichherzer, *Topic Extraction and Extension to Support Concept Mapping*, Proc. of the Sixteenth FLAIRS, 2003.

12. Carvalho, M., R. Hewett, A. J. Cañas, *Enhancing Web Searches from Concept Map-based Knowledge Models*, SCI 2001, Orlando, FL (July 2001).

13. Fellbaum, C. ed., *WordNet – An Electronic Lexical Database*, MIT Press, 1998.

14. Li X., S. Szpakowics, S. Matwin, A WordNet-based Algorithm for Word Sense Disambiguation Proceedings of IJCAI-95. Montréal, Canada, 1995.

15. Mihalcea, R., D. Moldovan, *A Method for Word Sense Disambiguation of Unrestricted Text*. Proc. of ACL '99, pp.152-158, Maryland, NY, June 1999.

16. Mihalcea, R., D. Moldovan, *An Iterative Approach to Word Sense Disambiguation*, Proc. of Flairs 2000, pp. 219-223, Orlando, FL, May 2000.

17. Nastase, V., S. Szpakowics, *Word Sense Disambiguation in Roget's Thesaurus Using WordNet*, Proc. of the NAACL WordNet and Other Lexical Resources Workshop. Pittsburgh, June 2001.

18. Fellbaum C., M. Palmer, H. Dang, L. Delfs, S. Wolf, *Manual &Z Automatic Semantic Annotation with WordNet*. Workshop on WordNet & other Lexical Resources. NAACL-01, 2001.