

# A study about synonym replacement in news corpus

Roxana Angheluta, Marie-Francine Moens  
roxana.angheluta@law.kuleuven.ac.be

**ICRI - Interdisciplinary Center for Law and Information Technology  
Katholieke Universiteit Leuven  
Belgium**

## 1 Introduction

An important issue in many fields dealing with text processing (retrieval, summarization, etc) is synonym recognition. In information retrieval, for example, the query can be augmented with synonym words in order to improve the recall of the retrieved articles. In summarization, the synonyms are to be used for detection of redundant passages in a text. The fact that many words are polysemous adds difficulty to the task, involving word sense disambiguation. While in the case of retrieval, the lack of the context for the query words makes their disambiguation quite difficult, in the case of summarization the words are surrounded by context, which theoretically should help in the process of synonym detection. Having a resource with synonyms, the task is to detect synonym words from the text, taking care of the correct sense of the word. In fact the problem can be seen as a problem of disambiguation using a thesaurus.

## 2 Methods

We used WordNet [7] as the back resource with synonyms. Senses in the WordNet database are represented relationally by synonym sets ('synsets') which are the sets of all the words sharing a common sense. Words of the same category are linked through semantic relations like hyponymy (more specific terms), hypernymy (more general terms), etc. Polysemous words appear in more than one synset. Each

synset has a gloss associated. Using the hyponymy/hypernymy network in WordNet (see figure 1; in the brackets is the sense of the first word from the synset), one can compute the relatedness between each pair of synsets as the length of the path between them. It is a classical search problem, resolved by us with iterative deepening algorithm, with a maximum search depth of  $\langle \text{MAX\_DEPTH} \rangle$ , i.e. words farther away than  $\langle \text{MAX\_DEPTH} \rangle$  were considered not related. We allowed only certain paths between words, taken from literature [6] (e.g. only one change of direction, where hypernymy=upward relation, hyponymy=downward relation) . We used a binary measure of relatedness.

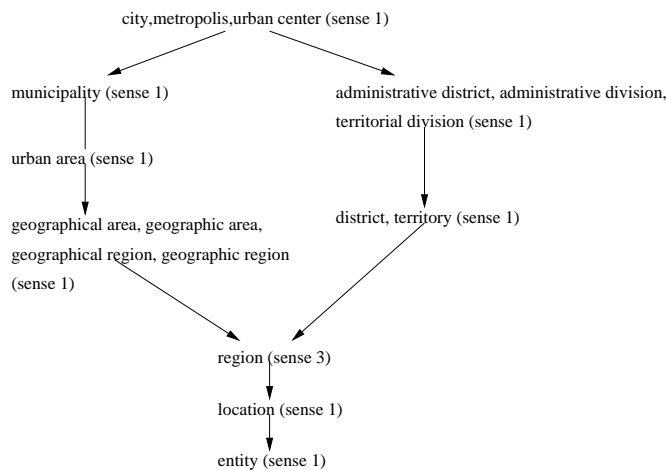


Figure 1: Hypernyms of the first sense of the word *city*

For detection of synonyms we used the algorithm for lexical chains construction described in [1], restricted just to the relation of synonymy. The notion of cohesion was introduced by Halliday and Hasan [3] in 1976 as a device for “sticking together” different parts of text. It is achieved through the use of semantically related terms, reference, ellipsis and conjunctions. The most easily identifiable and the most frequent type is lexical cohesion, created by using semantically related words. An example of a lexical chain is: *technology-science-technology-*

*computer-science-ai-intelligence-field*. The relatedness between chain members is not well defined, but one can agree that at least synonym words should belong to the same chain. Lexical chains have been used for information retrieval [5], for correction of malapropism [6] and for summarization [1], [2].

We consider just the nouns from the text, extracted with a tagger [4]. They are stemmed using the morphological module of WordNet. The algorithm works in two phases: words disambiguation and chain construction (see figure 2).

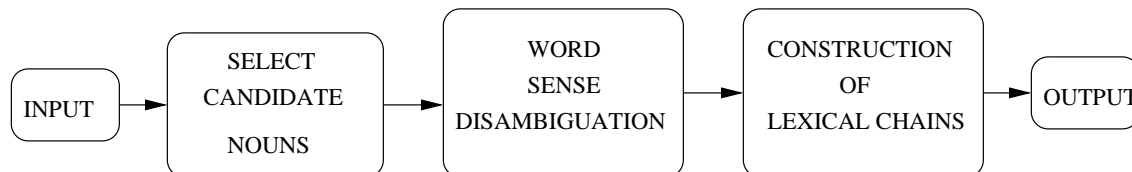


Figure 2: The algorithm for constructing the chains

## 2.1 Disambiguation

For the word sense disambiguation, we used the notion of component. A component is a cluster of related words which contains the possible combinations of senses for the words inside (a vector of synsets), called interpretations. Each interpretation has a power associated, which is the number of links between its synsets. The interpretations are ordered by descending power. When a new word is processed, it is tried to be put in one of the existing components, if it is related with one of the words from that component. Otherwise a new component is created.

```

process(<new_word>){
  inserted=FALSE
  for each existing component <comp>
    for each word <word> from <comp>
      if <new_word> is related with <word>
        then
          insert <new_word> into <comp>
          inserted=TRUE
        endif
  }
}
  
```

```

endfor
endfor
if(inserted==FALSE)
then
  create_a_new_component(<new_word>)
endif
}
  
```

Two words are considered related when at least one sense of the first word is related with a sense of the second word, using the WordNet network. Each time a new word is added in a component, the number of interpretations for that component is updated and the weak ones (the last `<no_interpretation>`-`<MAX_NO_INTERPRETATIONS>` in the ordered list) are removed. The maximum number of allowed interpretations `<MAX_NO_INTERPRETATIONS>` in a component is given as a parameter.

An example of two possible interpretations associated with a component is presented in figure 3.

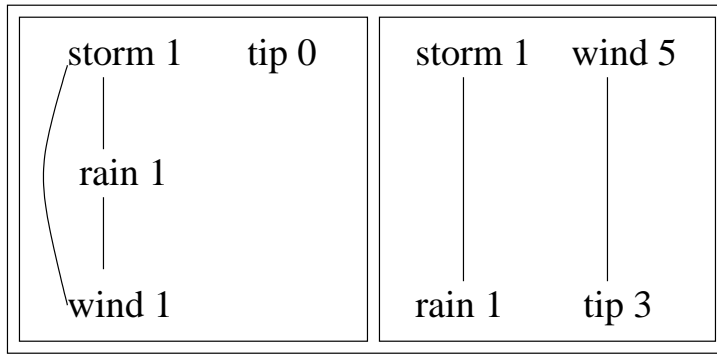


Figure 3: An example of two interpretations associated with a component

In the end, after all the words are processed, each word is assigned the sense from the most powerful interpretation from the component it belongs to.

As one can notice from the above mentioned algorithm, the disambiguation is based on all the words inside a component. The algorithm has a greedy aspect, because each word is assigned to the first component with which the word is related. It is thought not completely greedy, because for each word a number of senses (implicitly set by the parameter `<MAX_NO_INTERPRETATIONS>`) is kept till the end, when the strongest sense is chosen. An exhaustive comparison between each pair of words in a text is unrealistic for large texts.

## 2.2 Chain construction

In our implementation, the chains will consist of synonym words. From each strong interpretation of each component, the synonym words are selected in a vector.

The greedy aspect of the algorithm for disambiguation can lead sometimes to the same word in different components. For example, if one has in the text the words: *hurricane, ..., wind, ..., storm, ... wind*. At its first apparition in the text, the word *wind* has no link with any of the existing components (*wind* and *hurricane* are at a distance of 6, so they are considered unrelated), so a new component is created for it. After, the word *storm* is inserted in the component created for *hurricane* (say component 0) and the next apparition of the word *wind* is linked to it because of the relationship

with word *storm*. And all the following apparitions of *wind* will be put in component 0. So words tend to agglomerate in the first components. This effect is inversely proportional with the power of relatedness: if the relatedness criterion is very strict (just synonym words are considered as being related), then there will be less cases than when also hypernyms/hyponyms are considered as related words. In order to correct a bit this behavior, the algorithm matches in the end chains that have a word in common.

We considered two experiments:

- `<MAX_DEPTH>=0` - consider just synonym words for the disambiguation. Will be further referred as `expl`.
- `<MAX_DEPTH>=4` - consider related words that are separated by a path of length maximum 4 in WordNet. Will be further referred as `exp2`.

The first experiment has the advantage of simplicity and speed, while the second one increases the accuracy of the synonym words in the chains.

## 2.3 Synonym replacement

Each chain of words is assigned a most representative element of the chain, i.e. the first most frequent word from that chain. All the other words from the same chain are replaced in the original text with the most representative element.

### 3 Evaluation

In order to evaluate the efficiency of the synonym replacement we counted the number of errors obtained after replacement of the words from a chain with the most representative word of that chain. We used news texts from TREC disks employed at DUC 2002. The articles were extracted from: Wall Street Journal 1987-1992, AP newswire 1989-1990, San Jose Mercury News 1991, Financial Times 1991-1994, LA Times from disk 5 and FBIS from disk 5.

We considered a subset of 34 documents (4 clusters of related articles). We did not count capitalization errors, wrong accord and punctuation errors, because they can be easily solved.

We only looked closely to changes of meaning due to the placement in the same chain of words that were not synonyms.

There were two types of errors in the construction of the chains, both generated by the same cause (wrong disambiguation):

1. the same word (stem) appears in a chain more than one time, but with different senses. Will be further referred as error1.
2. different unrelated words are put in the same chain. Will be referred as error2.

A few examples of error 2 are presented in figure 4.

Words put in the same chain	Examples of the contexts
eye-C(c)enter	looks like the eye - National Hurricane Center
world-man	beloved by the world - Renaissance man; Rain Man - around the world
history-Story	other niches in history - West Wide Story
people-Mass.	well-known to people - Tanglewood, Mass.;
	moved so many people - Bernstein's 1971 "Mass"
Hispaniola-Haiti	moved south of Hispaniola - heavy rain on ... Haiti
air-lines	whipping paper through the air - downed power lines;
	a considerable amount air time - ads for public condolences at \$15.85 a line
death-end	shot to death - the end of the checkpoint
biography-life	in a controversial biography - his personal and professional life
(H)honor-award	Legion of Honor - Peabody award
Marx-Groucho	Groucho Marx
office-roles	at the box office - fulfilled their roles

Figure 4: Errors made for different words considered as synonyms

Sometimes it was difficult to decide if the words replaced were synonyms when they were part of an expression or an idiom. For example, when *secretary of state* was replaced by *secretary of country*, we did not count it as an error. Other difficult cases were abstract terms which are synonyms, like *thing-matter*, *scene-aspect*, but which appear in unrelated parts of the text and should not be put in the same chain, because they do not improve coherence. Following the definition of the lexical chains we still considered them related and we did not count them as an error.

One can notice that we tried to be as un-

demanding as possible and that the number of errors we counted is actually the minimum number of errors. The number of errors are presented in figure 5.

Sometimes different parts of speech were wrongly tagged as nouns. In the brackets are the real numbers of errors, not caused by the tagger. The percentages represent the fraction of the replacements that belong to the same stem and different stems, respectively, for rows 2 and 4 and the fraction of errors in each of these two categories, for rows 3 and 5. The precision of the algorithm can be measured as 100%-%errors.

	exp1	exp2
no_replacements	650(639)	504(500)
no_words_same_stem	395(394) 60.7%(61.6%)	406(405) 80.5%(81%)
error1	17(16) 4.3%(4%)	17(16) 4.1%(3.9%)
no_words_dif_stem	255(245) 39.2%(38.3%)	98(95) 19.4%(19%)
error2	142(132) 55.6%(53.8%)	36(33) 36.7%(34.7%)

Figure 5: Synonym replacement evaluation

## 4 Discussion of the results and possible improvements

The first type of errors is not very frequent when considering common nouns, as a study made in [8] concludes. Yarowsky counted for a few ambiguous words the percentage with which they appear with the same sense in a document and he got very high figures, which proves that usually a word appears with the same sense throughout a discourse. Notable cases in this category are the elements of proper names, like in *Mexico City* and *city* or *cities*. We did not count these cases as errors, unless the sense was obviously different, like in *National Hurricane Center* and *the center of the storm*, *New York Times* and *New York time*.

The second type of errors appears mainly because of the lack of the context. When disambiguating a word, we compare for each sense the context in which that sense occurs (the rest of the words from the same synset and the words from related synsets) with the context in the text (other nouns in the text). When there is not enough evidence for a certain sense, the word can be wrongly disambiguated. For example, if in the same text the words *side* and *English* appear and no other *side*-related words are encountered (e.g. slope, incline, position, face), then they are considered synonyms, like in the synset:

*English, side -- ((sports) the spin given to a ball by striking it on one side or releasing it with a sharp twist)*

One can notice that in the second experiment the number of errors for different stems is highly reduced, because we enlarged the context in the dictionary with all words from related synsets. Still, around 66% precision (100%-34.7%) is not a very high number and one should experiment carefully before deciding to detect synonyms based on WordNet.

In the results from figure 5 only the number of errors is computed (and implicitly the precision). We did not measure the recall of the correct synonyms detected. Usually all the words detected as synonyms in the second experiment were also detected in the first one. Sometimes, correct synonyms detected in the first experiment *style-flair*, *candidates-nominees*, *director-conductor* were not detected in the second one, which make us hypothesize that the recall is lower in the second experiment. Probably in most applications the recall is not so important as precision, but one should be aware of the trade-off between these two measures.

Proper names should not be treated in the same way as the common ones. Especially in news corpora, where they appear a lot, they are largely responsible for the first type of errors. But if they are put in separate chains, then they will lower the recall. Without real coreference resolution the problem cannot be properly solved.

It has been mentioned in the literature that WordNet is too fine-grained for being used as a thesaurus. We did not feel this as the main

problem for our task, but rather the lack of the context needed for disambiguation. A learning approach might lead to better results than a thesaurus-based one, but the need of a training corpus counter-balance the advantage of the technique. An approach similar with the one of Yarowsky [8] - which learns contexts starting with a “seed” context possibly taken from a dictionary - might yield good results for senses that are clearly separated (like plant in the sense of living thing and plant in the sense of manufacture) which appear in distinctive contexts, but one should test if it resolves more difficult cases.

## 5 Conclusions

Although synonym replacement in a text is conceptually a simple problem, there are no simple techniques to resolve it accurately. The bottleneck is due to the necessity of word sense disambiguation, which is a difficult task. In our experiment, we detected synonyms using the algorithm for lexical chains construction, with WordNet as a thesaurus. We counted the number of errors introduced by placement in the same chain of words that were not synonyms. The results showed that while in the case of the same stem, the error rate was quite low, in the case of different stems it was much higher. If one would like to use WordNet as a thesaurus for synonym replacement, one should be careful not to introduce more noise and to study the influence of wrong disambiguation on the final results.

## References

- [1] Barzilay R. and Elhadad M. (1997) *Using Lexical Chains for Text Summarization*. <http://citeseer.nj.nec.com/324597.html> (visited September, 5th, 2002).
- [2] Brunn M., Chali Y. and Pinchack C.J. (2001) *Text Summarization Using Lexical Chains*. In Proceedings of Document Understanding Conference, New Orleans, USA, September 13, 2001, pp. 135-140.
- [3] Halliday M.A.K. and Hasan R. (1976) *Cohesion in English*, London: Longman.
- [4] Mikheev A. (1998) *Part-of-Speech Guessing Rules: Learning and Evaluation*. <http://www.ltg.ed.ac.uk/software/pos/> (visited September, 5th, 2002).
- [5] Stairmand M. (1996) *A Computational Analysis of Lexical Cohesion with Applications in Information Retrieval*. PhD thesis, Center for Computational Linguistics, UMIST, Manchester.
- [6] St-Onge D. (1995) *Detecting and Correcting Malapropisms with Lexical Chains*. Department of Computer Science, University of Toronto, Toronto, Canada. <http://www.cs.toronto.edu/compling/Publications/Abstracts/Theses/StOnge-thabs.html> (visited September, 5th, 2002).
- [7] Miller G.A., Beckwith R., Fellbaum C., Gross D. and Miller K.J. (1990) *Introduction to WordNet: An on-line Lexical Database*. International Journal of Lexicography (special issue), 3(4):235-312.
- [8] Yarowsky D. (1995) *Unsupervised Word Sense Disambiguation Rivaling Supervised Methods*. In Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics. Cambridge, MA, pp. 189-196.