Computational Linguistics Project Proposal and First Report

Ishan Agarwal

16 June 2017

Proposal

We propose to attempt a classification of nouns in WordNet as 'artifacts' in the sense of the term in generative lexicon theory. Furthermore we propose to look at the synset containing 'artifact' in WordNet and all it's hyponyms so as to study the overlap between this set of nouns and the nouns our algorithm classifies as artifacts based on the glosses of their synsets.
Part of the motivation for this is to understand the difference between the concept of an artifact in WordNet and the notion of the same in GL theory. We would further like to develop a satisfactory algorithm that can, from reading glosses, pick out the nouns that fit the GL notion of being artifacts. Note that for the moment we are avoiding the problem of polysemy considering the fact that previous work, for example CoreLex, should be able to resolve this.

Work done so far

As a first step I tried searching through all the synsets of Wordnet and look at their glosses for the word 'used'. As a typifying feature of artifacts is their having a specific purpose, this was chosen as the starting point.
However this leads to the inclusion of certain extraneous synsets for non physical things like one containing say 'atlantic time'. This was easily remedied by restricting our search to hyponyms of the synset containing 'physical object'.
Furthermore as hyponymy is a transitively closed relation I took the closure over hyponyms of the list of synsets I generated. This is important as many synsets deeper down the hierarchy do not have any typical phrase in their glosses but some hypernyms of these synsets do.
We were also able to compare the list we generated to the set of synsets in wordnet consisting of 'artifact' and all it's hyponyms.

Code

This is the python code to generate our artifact list:

```
import nltk
from nltk.corpus import wordnet as wn

stuff=wn.synsets('object')[0]
hypostuff=set([i for i in stuff.closure(lambda s:s.hyponyms())])
hypo=set([])
artifacts=set([])

for synset in list(wn.all_synsets('n')):
        hypo.clear()
        if((synset.definition().count("used")>0)&(synset in hypostuff)):
               artifacts.add(synset)
               hypo=set([i for i in synset.closure(lambda s:s.hyponyms())])
               artifacts=artifacts.union(hypo)

print(artifacts)
```

Here is the code with some trivial modifications to compute precision and recall:

```
import nltk
from nltk.corpus import wordnet as wn

stuff=wn.synsets('object')[0]
hypostuff=set([i for i in stuff.closure(lambda s:s.hyponyms())])
hypo=set([])
artifacts=set([])
wn_artifacts=set([])
overlap=([])
wn_artifact = wn.synsets('artifact')[0]
wn_artifacts = set([i for i in wn_artifact.closure(lambda s:s.hyponyms())])

for synset in list(wn.all_synsets('n')):
        hypo.clear()
        if((synset.definition().count("used")>0)&(synset in hypostuff)):
               artifacts.add(synset)
               hypo=set([i for i in synset.closure(lambda s:s.hyponyms())])
               artifacts=artifacts.union(hypo)

overlap= artifacts.intersection(wn_artifacts)
count_ours=len(artifacts)
count_theirs=len(wn_artifacts)
count_overlap=len(overlap)

precision=count_overlap/count_ours
recall=count_overlap/count_theirs
```

```
print(precision)
print(recall)
```

Results

We were able to generate a list of synsets which are pysical objects and either have the word 'used' in their gloss or are hyponyms of such synsets.
It is unusual that many objects that we considered artifacts were not considered so by wordnet. We obtained a precision of 69.96 percent and recall was only 43.56 percent. While recall was expected to be lower than precision, I expected the precision figure to be significantly higher as we essentially are only counting synsets with the word 'used' in the gloss, which I expected to be an overly stringent criterion that would result in very high precison and quite low recall. I believe that this might have to do with a usage of the word 'used' in the gloss in some manner that I did not forsee and not as denoting that the nouns in the synset were made for some particular use.
I suspect that the word 'used' may be refering to word usage of the nouns in the synset in some cases such as in the kinds of phrases commonly seen in dictionaries like "this word is often used in a negative sense". Here our algorithm will choose the synset where the choice is not justified by the presence of the word use. I plan to manually inspect some of the data to develop heuristics to mitigate this problem.

Next Steps

We propose three further related directions of work:

- We can try to better study the overlap between our list and the artifact synset of Wordnet with all its hyponyms, as has been already discussed. This could lead to several interesting insights such as the suitability of wordnet for a GL framework, a better understanding of the notion of an artifact from the point of view of Wordnet and so on.
- We can try to integrate this tool with other tools to develop an ontology keeping in mind GL principles.
- It is likely that our algorithm is not entirely satisfactory at picking out words that native speakers would call artifacts. We can try to improve this crude algorithm and try to see how much more information can be extracted from glosses to reduce both type 1 and 2 errors that we suspect. A first step for this would be to find a satisfactory way to find errors; we would have to set up a standard against which to test the algorithm.

As is clear the above can all be done to some extent in parallel and do indeeed entail one another to some degree.

References

I studied material from the following to familiarize myself with the Generative Lexicon theory and some basic ideas in computational linguistics:

- The Generative Lexicon ; James Pustejovsky
- The Language of Word Meaning; edited by Pierrette Bouillon and Federica Busa
- Speech and Language Processing ; Daniel Jurafsky and James H. Martin

The following was used as a reference for WordNet:

- WordNet An Electronic Lexical Database ; Christiane Fellbaum

Acknowledgements

I would like to thank

- Nikhil Krishnaswamy who helped me a lot with setting up and learning the new programming environment
- Marc Verhagen who suggested this project idea to me
- Prof. James Pustejovsky for introducing me to his Generative Lexicon theory and giving me an opportunity to work with his group this summer