6.863 Final Project Writeup

Mark Richardson
(Dated: May 14, 2010)

I. SYSTEM DESIGN

The goal of this system is to provide a measure of semantic similarity between any two verbs. The VerbOcean dataset contains various semantic relationships for a set of 3477 unique verbs. The pairwise relationships between verbs contained in VerbOcean were used to create a graph of semantic relations between many verbs. A tree traversal algorithm was implemented so that, given a starting word and a target word, various semantic paths from the starting word to the target word could be found. The VerbOcean dataset was generated by searching for the number of web pages which contained pairs of verbs in various simple syntaxes. The relationships incorporated into VerbOcean are similarity, strength, antonymy, enablement, and precedence. Similarity between two verbs was found by searching for instances of 'to X and Y' or 'Xed and Yed', while precedence was found by searching for 'to X and then Y' or 'Xed and then Yed'. The number of results found (using Google) corresponded to the strength of a particular relationship between two verbs. The score of the shortest semantic path between two verbs intuitively gives a measure of how closely related those two verbs should be.

II. IMPLEMENTATION

The VerbOcean dataset is contained in a line based text file, where each line contains two verbs, the relation between them, and the strength of that relation. The text file is imported into python by issuing the command `semanticnet.build_net()` with the file `verbocean.unrefined.2004-05-20.txt` in the python path. This constructs a dictionary of all relations between any two verbs, and for relations which are symmetric, both the original relation and the reversed relation are inserted. The command `semanticnet.relation(word1,word2,N,relations)` searches for the `N` best semantic paths between `word1` and `word2`. The types of semantic relations included in the search for paths between words are listed in the variable `relations`. When `relations` is an empty list, the system defaults to using all available relations. When `relations` is nonempty, it must be a list containing one or more of the following strings which correspond to semantic relationships:

```
'happens-before'
'happens-after'
'similar'
'opposite-of'
'results-in'
'results-from'
'stronger-than'
'weaker-than'
```

The semantic relationships always operate in the direction `word1 -> word2` so that the relation `'stronger-than'` would allow the pair `abhor -> hate` while the relation `'weaker-than'` would allow the pair `hate -> abhor`. The search function uses best first search, where the cost associated with each path is given by the negative log of the score listed in the VerbOcean dataset. The similarity between two words is given by the score of the last relationship sequence returned by this script (the most positive score). This score corresponds to the shortest path between the two verbs given.

A transcript of the output from this script:

```
>>> import semanticnet
```

```
>>> semanticnet.build_net()
>>> semanticnet.relation('invent','manufacture',4,['happens-before','similar'])
invent [similar] patent [happens-before] manufacture :: Score -1.74229912807
invent [happens-before] patent [happens-before] manufacture :: Score -1.7105907517
invent [similar] patent [similar] manufacture :: Score -1.66432466812
invent [happens-before] patent [similar] manufacture :: Score -1.63261629174
```

III. STRENGTHS AND WEAKNESSES

The VerbOcean dataset captures a number of semantic relationships between words which extend beyond a simple notion of equivalence or opposition. For example, the verbs `hate` and `abhor` are synonyms with one another so their relationship is obvious. The relationship between the verbs `drive` and `crash` is not as obvious in terms of equality, but intuitively there is a relationship between the verbs. Because the VerbOcean data captures temporal relationships between verbs (drive and then crash) as well as facilitative relationships (one must drive in order to crash).

On the other hand, the VerbOcean dataset does not distinguish between various forms of the same word. For example, the verb `run` can mean 'to use' as well as 'to flee', but within the context of VerbOcean they are all the same. Another popular semantic network of relationships between words is WordNet. WordNet is able to capture the difference between different meanings of the same word by clustering words into 'synsets', groups of synonymous words. There exist multiple versions of words which have multiple meanings, each of which corresponds to a different synset. While WordNet captures these semantic differences within the context of a single word, the relationships between words that WordNet includes are limited to hierarchical clustering such as 'perceive' is a kind of 'listen', as well as facilitative relationships such as 'sleep' enables 'snore'. The temporal relationships that VerbOcean offers are absent from WordNet.

In calculating the similarity between two words, both WordNet and VerbOcean have their own advantages. Compared to WordNet, VerbOcean is able to find relationships between verbs when there is no hierarchical relationship between them. On the other hand, the clusters of synonymous words that exist in WordNet allow for the calculation of distance between verbs not only by their hierarchical distance from one another, but also taking into account the overlap of their synonym clusters.

IV. EVALUATION

WordNet's ability to distinguish between various forms of a particular verb allow it to avoid many of the issues that arise within the VerbOcean based implementation. The issue of ambiguity in the semantic meaning of a verb extends beyond just the start and target words. For many pairs of verbs the VerbOcean based search will find a semantic path relating the two verbs when it seems that no such path should exist. For example, for the verbs `love` and `read` one would expect there to be either a very long path connecting the two or no path at all, but instead we find that the system behaves quite strangely:

```
>>> semanticnet.relation('love','read',1,[])
love [results-from] find [weaker-than] read :: Score -1.92219677788
```

On closer inspection, the system is simply drawing pairwise relations based on particular semantic meanings of verbs, and then other pairwise relations involving the same words but different semantic meanings. In the above example we see that `love` results from `find` (searching), but `find` is also a weaker form of obtaining information than `read`. The semantic meaning of the verb `find` is different in the first pairwise relation than in the second, leading to semantic paths between verbs which are seemingly illogical. The VerbOcean system, however, is able to determine many valid relationships which WordNet is not. For example, the verbs `start` and `stop` have several relationships to one another. The VerbOcean script is able to capture most of these relationships:

```
>>> semanticnet.relation('start','stop',3,[])
start [opposite-of] end [similar] stop :: Score -2.01411318797
start [similar] stop :: Score -1.04520657567
start [happens-before] stop :: Score -0.846385444933
```

On the other hand, WordNet is unable to determine any semantic relationship between start and stop since WordNet captures neither antonym relationships nor temporal relationships.

```
>>> start=wordnet.synset('begin.v.03')
>>> stop=wordnet.synset('end.v.01')
>>> start.path_similarity(stop)
-1
```

Both WordNet and VerbOcean hold unique information which alleviates a different range of problems. While WordNet is able to avoid the issue of semantic ambuguity using clusters of synonyms, VerbOcean captures far more information regarding semantic pathways between verbs by incorporating temporal relationships. Ultimately a superior system would incorporate features from both systems, maintaining the synsets that exist in WordNet and incorporating the temporal and antonymic relationships between verbs that are contained in VerbOcean.