Analyzing German Noun Compounds using a Web-Scale Dataset – Report Week 6

TECHNISCHE UNIVERSITÄT DARMSTADT

UIMA Software Project WS 2010/2011
Jens Haase

1 Work done in the last week

1.1 Splitting algorithm

The last week started with some small improvements for the splitting algorithm. All split elements with less than three letters are not added to the tree. This had no effect on the recall, but in reduces the number of possible splits.

1.2 Frequency-Based Method

The first implementation of a ranking algorithm is the frequency based method. To evaluate the method I added an evaluation class. Running the evaluation currently takes very long. For the evaluation only the first thousands words are used. This takes 30 minutes to complete. At the end we have a recall of 0.415 without morphemes and 0.667 with morphemes. That is not that great but I do not expect such a high value for this easy method.

The bottleneck of the algorithm currently is the search in the lucence index. A single search often takes over one second.

1.3 Probability-Based Method

Next, the probability based method was implemented. For evaluation also only the first 1000 words are used. It has nearly the same run time as the frequency based method above, because of the same bottleneck.

The evaluation has a very bad recall of 0.074 with morphemes and 0.119 without morphemes. This is because the methods favors the splits with the smallest number of parts. Mostly the highest rank word is the word that is not splitted. Maybe this method works better on small corpus sizes.

1.4 Time Tracking

Date	Task	Needed time	Planned time
2010-12-16	Improve splitting algorithm	1	0
2010-12-18	Frequency based ranking	2	4
2010-12-28	Probability based ranking	2	4
2010-12-22	Evaluation and Report Writing	3	3

2 Plan for next week

Before implementing another ranking algorithm the bottleneck should be resolved. For real application it can not take a few seconds to split a word. My attempt to improve the speed is to create separated index for each n-gram type. One for the unigram, one for the bigrams and so one. The algorithm than

can ask each index in parallel. Also the search should be faster on smaller indexes. Another solution could be to use a cache for the search results. This can help for the lookups of a single split, but not necessary for different splits, because the words are mostly very different.

Date	**Task**	**Planned time**
2010-12-27	Resolve bottleneck	3
2010-12-28	Resolve bottleneck	3
2011-01-02	Resolve bottleneck and Report Writing	4