Analyzing German Noun Compounds using a Web-Scale Dataset – Report Week 5

TECHNISCHE UNIVERSITÄT DARMSTADT

UIMA Software Project WS 2010/2011
Jens Haase

1 Work done in the last week

1.1 Tree visualization

After refactoring the split algorithm in the last week, the algorithm will now return a tree instead of a list. The tree can now be visualized. For that I created 20,000 split with the current implementation. All trees are converted to a `JSON` structure and stored in a CouchDB Database. With the help of JQuery (`http://jquery.com/`) and Javascript InfoVis Toolkit (`http://thejit.org/`) I created a little web application. It can be see on `http://jenshaase.couchone.com/noun_decompounding/_design/tree/index.html`. The code of the web application can be found in `src/main/couchapp`. To install the application you need to install CouchDB and `node.couchapp.js` from `https://github.com/mikeal/node.couchapp.js`.

1.2 Improving splitting algorithm

In the next step i want to improve the current splitting algorithm. In the last week we had a recall of 0.47 with morphemes. I added a few small improvements to generate more words with different morphemes. Also the list of morphemes was updated. It contains now all morphemes that appear in the evaluation corpus.

These improvements result in a recall of 0.81 with morphemes and 0.89 without morphemes. I think this are good value to start with the ranking algorithm.

1.3 Ranking algorithm

In the next few weeks all splits in the current tree have to be ranked. The split with the highest rank will be the correct split. With the current implementation there are two options to create a ranking. The first and more easy one is to convert the tree to a list an rank each split individual. The other, more complex one, is to evaluate the splits that are added to the tree. This has the advantage that complete subtree can be skipped when the calculated value is to bad. In the following weeks I will focus on the first method. The experiences with this method can help me to build the more complex method.

1.3.1 Frequency-Based Method

The first attembt to build a ranking function should be a frequency based method as described by Alfonseca [ea08]. For each split S with the words s_i we calculate the geometric mean of the frequencies from the `Web1T` corpus:

$$F_s = \prod_{s_i \in S} freq(s_i))^{\frac{1}{|S|}} \tag{1}$$

The split with the highest F_s will be the highest ranked split.

1.3.2 Probability-Based Method

The second method is also described by Alfonseca [ea08]. For each Split S with the words s_i we calculate

$$P_s = \sum_{s_i \in S} -log(\frac{freq(s_i)}{F}) \tag{2}$$

where F is the total amount of frequencies. The split with the lowest P_s will the the highest ranked split.

1.3.3 Mutual Information

The previous methods only focus on the individual word and not on the co-occurrence of the words. For that we add the next method, also described by Alfonseca [ea08]. For a word pair w_1 and w_2 we can calculate the mutual information:

$$M(w_1, w_2) = log_2(\frac{\frac{freq(w_1,w_2)}{F}}{\frac{freq(w_1)}{F} \times \frac{freq(w_2)}{F}}) = log_2(\frac{F \times freq(w_1, w_2)}{freq(w_1) \times freq(w_2)}) \tag{3}$$

For all word pairs in a split S we can calculate the mutual information and average it. The split with the highest averaged mutual information is the highest ranked split.

1.3.4 Combination

A combination of this tree method can help to improve the result.

1.4 Time Tracking

Date	Task	Needed time	Planned time
2010-12-09	Improve splitting algorithm	2	4
2010-12-14	Ranking functions	2	3
2010-12-15	Ranking functions and Report writing	4	4

2 Plan for next week

In the next week I will start to implement a the first ranking method. I will also focus on the evaluation of the final results.

Date	Task	Planned time
2010-12-16	Ranking algorithm	4
2010-12-18	Ranking algorithm	3
2010-12-20	Ranking algorithm and Evaluation	2
2010-12-22	Report Writing	2

References

[ea08] Enrique Alfonseca et al. German decompounding in a difficult corpus. In *Computational Linguistics and Intelligent Text Processing*, 2008.