German Compound Analysis with wfsc

Anne Schiller

Xerox Research Centre Europe
6 chemin de Maupertuis, 38250 Meylan, France
Anne.Schiller@xrce.xerox.com

Abstract. Compounding is a very productive process in German to form complex nouns and adjectives which represent about 7% of the words of a newspaper text. Unlike English, German compounds do not contain spaces or other word boundaries, and the automatic analysis is often ambiguous. A (non-weighted) finite-state morphological analyzer provides all potential segmentations for a compound without any filtering or prioritization of the results.

The paper presents an experiment in analyzing German compounds with the Xerox Weighted Finite-State Compiler (wfsc). The model is based on weights for compound segments and gives priority (a) to compounds with the minimal number of segments and (b) to compound segments with the highest frequency in a training list. The results with this rather simple model will show the advantage of using weighted finite-state transducers over simple FSTs.

1 Compound Construction

A very productive word formation process in German is compounding, which combines words to build more complex words, mainly nouns or adjectives. In a large newspaper corpus the Xerox German Morphological Analyzer [1] identified 5.5% of 9,3 million tokens and 43% of overall 420,000 types[1] as compounds. In other texts, such as technical manuals, the percentage of compound tokens may even increase (e.g. 12% in a short printer manual). This is comparable to the observations of Boroni et al. [2] who found in a 28 million newswire corpus that 7% of the tokens and 46% of the types were compounds.

Regarding the construction of compounds, any adjective or noun (including proper names) may, in principle, appear as head word[2], and any

[1] *Tokens* represent all words of the text, *types* count only different word forms.

[2] Verbal compounds (such as *spazierengehen*) exist, but are much less productive than nouns or adjectives. They are not taken into account in this experiment.

adjective, noun or verb may occur as the left-hand ("modifier") part of a compound.

- *Buchseite* (book page)
- *Großstadt* (big town)
- *grasgrün* (grass green)
- *Goethestück* (Goethe piece)

The Xerox finite-state tool [3] for German morphological analysis [1] implements this general principle without any semantic restrictions and with only a few morphosyntactic constraints concerning the so-called "linking" elements. The advantage of this very general approach is high and robust coverage. The inconvenience is potentially very ambiguous output due to "over-segmentation" of (long) words. A potential source of over-segmentation is the homonymy of compound parts with derivational affixes.

- derivational suffix *-ei* (ing) vs. noun *Ei* (egg), e.g. *Spielerei* (playing) – *Vogel#ei* (bird egg)
- prefix *ein-* (in-) vs. cardinal *ein* (one), e.g. *Einwohner#zahl* (inhabitant number) – *Ein#zimmer#wohnung* (one room apartment)

But other "accidental" homonymy may also lead to over-segmentation:

- *Verbraucher* (consumer) vs. *Verb#Raucher* (verb smoker)
- *Abteilungen* (departments) vs. *Abtei#Lungen* (abbey lungs)

While morphologically correct, most of these "over-segmentations" are semantically impossible—or at least very unlikely, i.e. they require a very special context to make sense. But there also exist some real ambiguities:

- *Hochzeit* (wedding) vs. *Hoch#Zeit* (high time)
- *Gründung* (foundation) vs. *grün#Dung* (green manure)

Another source for multiple analyses is the ambiguity of segments themselves, i.e. one surface form may be analyzed in different ways.

- noun *Alt* (alto) vs. adjective *alt* (old)
- prefix *auto-* (self) vs. noun *Auto* (car)
- masculine noun *Leiter* (leader) vs. feminine noun *Leiter* (ladder)

2 Finite-State Compound Analysis

A simple approach to modeling compound formation by a finite-state transducer can be described by a regular expression like `TRUNC+ HEAD` where `TRUNC` is a compound modifier (i.e. a left-hand side part), including a potential linking element (*"(e)s"* or *"en"*), and `HEAD` is the compound head word (i.e. its right-most part).

Example 1. Compound Part Lexicon[3]:

```
((
V e r b r a u c h e r "+NmSg":0          # (consumer)
| V e r b r a u c h "+NmSg":0           # (consumption)
| V:v e r b r a u c h e:0 n:0 "+V":0    # (to consume)
| E:e r z "+NnSg":0                     # (ore)
| V e r b "+NnSg":0                     # (verb)
| r a u c h e:0 n:0 "+V":0              # (to smoke)
| R:r a u c h "+NmSg":0                 # (smoke)
| R:r a u c h e r "+NmSg":0             # (smoker)
)
"#":0 )*
(
A:a h l e 0:n "+NfPl":0                 # (awl)
| z a h l e n "+NnSg":0                 # (paying)
| Z:z a h l 0:e 0:n "+NfPl":0           # (numbers)
);
```

Assuming the sample definitions above and a set of rules to cope with upper/lower case restrictions for resulting nouns and adjectives, our compound analyzer would provide the following results for the input word *Verbraucherzahlen* (consumer numbers).

Example 2. Finite-State Compound Analysis

```
Verbraucher+NmSg#Zahl+NfPl               (consumer numbers)
Verbraucher+NmSg#zahlen+NnSg             (consumer paying)
Verbrauch+NmSg#Erz+NnSg#Ahle+NfPl        (consumption ore awls)
Verb+NnSg#Raucher+NmSg#Zahl+NfPl         (verb smoker numbers)
Verb+NnSg#Raucher+NmSg#zahlen+NnSg       ...
Verb+NnSg#Rauch+NmSg#Erz+NnSg#Ahle+NfPl
Verb+NnSg#rauchen+V#Erz+NnSg#Ahle+NfPl
```

A human reader would probably only think of the first solution, while all others segmentations require a very special and artificial context to make sense.

[3] For the examples of regular expressions in this article we use the syntax of wfsc (Xerox Weighted Finite-State Compiler) [4].

The problem of over-segmentation may not affect applications such as part-of-speech tagging where only the category of the complex word is taken into account (which would be *noun* for all of the above decompositions. But a correct segmentation is crucial for applications like machine translation ([5], [6]) or information retrieval ([7]).

3 Weighted Finite-State Transducers

Weighted finite-state transducers (wFSTs) are like ordinary FSTs with input and output symbols on arcs, but they contain, in addition, a weight on every arc and every final state. These weights are combined during traversal of the automaton to compute a weight for each path [4]. To ensure that various operations on wFSTs are well defined, the weight set must correspond to the algebraic structure of a semiring:

semiring S = (K, $\oplus$, $\otimes$, $\bar{0}$, $\bar{1}$)
with

K = a set of weights
$\oplus$ = Collection operation
$\otimes$ = Extension operation
$\bar{0}$ = Identity element for collection
$\bar{1}$ = Identity element for extension

One example of a semiring is $(\Re^{+}, +, *, 0, 1)$, the *real semiring* (positive real weights with addition as the collection operator and multiplication as the extension operator. e.g. for modeling probability calculation, which will also be used for our experiment described in the sections below.

4 Adding Weights to the Lexicon

When looking at the results of the (unweighted) morphological analyzer we realized that the preferred reading for a human reader very often corresponds to the decomposition with the least number of segments. Section 4.1 describes how we can model this observation in a weighted finite-state analyzer that will output higher scores for analyses with lesser segments.

Furthermore we assume that we can use frequency information from a training corpus to obtain weights for segments with different lexical analyses in order to resolve ambiguities which remain after having chosen the “shortest” decomposition. This will be described in section 4.2.

4.1 Compound Segments with Equal Weights

The objective is to prioritize compounds with a minimal number of segments. Using the *real semiring* as shown above weights are multiplied along the path. If all segments have a weight less than 1, then a bigger number of segments leads to a lower overall weight.

With a segment weight of 0.5 our sample gives overall weights as follows:

Example 3. Finite-State Compound Analysis with Weights

```
Verbraucher+NmSg#zahlen+NnSg                <0.25>
Verbraucher+NmSg#Zahl+NfPl                  <0.25>
Verbrauch+NmSg#Erz+NnSg#Ahle+NfPl           <0.125>
Verb+NnSg#Raucher+NmSg#zahlen+NnSg          <0.125>
Verb+NnSg#Raucher+NmSg#Zahl+NfPl            <0.125>
Verb+NnSg#Rauch+NmSg#Erz+NnSg#Ahle+NfPl     <0.0625>
Verb+NnSg#rauchen+V#Erz+NnSg#Ahle+NfPl      <0.0625>
```

4.2 Compound Segments with Weights from Training Corpus

When using weights for segments which are derived from compound analyses in a training list, we must provide a *default* weight for segments which do not occur in the training list. The overall goal is still to give preference to a minimal number of segments. Therefore the multiplication of 2 maximal weights should be less than the minimal (default) weight of a single segment.

With a default weight of 0.5 for unseen segments, the maximal weight for training segments should then be 0.7 in order to stay below 0.5 when multiplying the weights of 2 segments (as $0.7 * 0.7 = 0.49$). Therefore we choose the following formula for the weight of a segment which consists of a lexical form *lex* and a surface realization *srf*:

$$weight(lex : srf) = 0.5 + freq(lex : srf)/(freq(: srf) + 1)/5$$

where $freq(: srf)$ is the frequency of all segments with surface realization srf.

In regular expressions, weights are enclosed in angle brackets as shown in Example 4.

Example 4. Compound Part Lexicon with Weights

```
semiring < realpos_sum_times > (
( V e r b r a u c h e r "+NmSg":0 <0.7>
| V e r b r a u c h "+NmSg":0 <0.65>
| V:v e r b r a u c h e:0 n:0 "+V":0 <0.55>
| E:e r z "+NnSg":0 <0.7>
| V e r b "+NnSg":0 <0.7>
| r a u c h e:0 n:0 "+V":0 <0.55>
| R:r a u c h "+NmSg":0 <0.65>
| R:r a u c h e r "+NmSg":0 <0.7>
)
"#":0 )*
(
A:a h l e 0:n "+NfPl":0 <0.7>
| z a h l e n "+NnSg":0 <0.55>
| Z:z a h l 0:e 0:n "+NfPl":0 <0.65>
);
```

Example 5. Finite-State Compound Analysis with Weights

```
Verbraucher+NmSg#Zahl+NfPl                  <0.455>
Verbraucher+NmSg#zahlen+NnSg                <0.385>
Verbrauch+NmSg#Erz+NnSg#Ahle+NfPl           <0.3185>
Verb+NnSg#Raucher+NmSg#Zahl+NfPl            <0.3185>
Verb+NnSg#Raucher+NmSg#zahlen+NnSg          <0.2695>
Verb+NnSg#Rauch+NmSg#Erz+NnSg#Ahle+NfPl     <0.22295>
Verb+NnSg#rauchen+V#Erz+NnSg#Ahle+NfPl      <0.18865>
```

5 Training and Test Data

The experiment focuses on disambiguation of compound analysis. For training and testing we created manually disambiguated compound lists. We analyzed word lists built from large corpora and selected all words that were recognized as compounds, not taking into account hyphenated compounds (such as *US-Präsident* or *Olympia-Sieger*) which bypass the problem of segmentation.

- Corpus used for training:
  - **LNG:** 37,362 compounds (types) from various sources law, newspapers, manuals, ...)
  - **SPG:** 151,22 compounds from the weekly “Der Spiegel” (years 1994 and 1995)

- Corpus used for evaluation::
  - **NZZ:** 30,891 compounds from the daily "Neue Zürcher Zeitung" (year 1994)
  - **MED:** 26,196 compounds from medical texts (from European project "Muchmore")

The different texts are independent, but the extracted lists are not fully disjunctive, i.e. the test corpora share around 17% of the compounds with the training list SPG.

Corpus	compounds	nb. of chars			nb. of segments			ambiguity		
		min.	max.	avg.	min.	max.	avg.	1*	max.	avg.
LNG	37,362	6	46	22.9	2	6	2.5	39.0%	113	3.0
SPG	151,220	10	40	14.6	2	7	2.1	48.5%	55	2.1
NZZ	30,891	19	40	22.9	2	5	2.4	42.1%	120	2.8
MED	26,196	6	45	16.3	2	5	2.2	49.8%	80	2.0

(*) percentage of compounds with single analysis

The longest compounds found in each compound list are

- **LNG:** Verkehrsinfrastrukturfinanzierungsgesellschaft
- **SPG:** Betäubungsmittelverschreibungsverordnung
- **NZZ:** Betäubungsmittelbeschaffungskriminalität
- **MED:** Normalgewebekomplikationswahrscheinlichkeiten

6 Tests and Results

The following process was applied for the evaluation:

- Take only "best scored" analysis results (i.e. all results for unweighted FSTs)
- Compare the FST result to the manually disambiguated lists:
  - *true positives* are the analyses which match the manual choice
  - *false positives* are all other results
  - Count a *false negative* if the manual choice is not among the results.
- The overall precision P is computed as
  P = #(true positives) / #(true positives + false positives)
- The overall recall R for this experiment is
  R = #(true positives) / #(true positives + false negatives)
- The F-score combines the two previous measures
  F = (2*precision*recall)/(precision+recall)

	NZZ			MED		
Test	P (%)	R (%)	F (%)	P (%)	R (%)	F (%)
baseline FST[4]	35.94	100.00	52.87	49.09	100.00	65.85
wFST with equal weights	66.24	99.91	79.67	63.28	99.90	77.48
wFST with weights from LNG	97.05	99.63	98.32	88.69	98.68	93.42
wFST with weights from SPG	97.73	99.06	98.39	95.90	98.32	97.09

7 Conclusion

Using weighted finite-state transducers for compound analysis with the suggested model provides different scores for different segmentations, and thus selecting results with best scores only filters out very unlikely readings. Most of the small number of remaining "real" ambiguities, however, may only be resolved with the full (semantic and syntactic) context of the compound, but a great majority of compounds were found to be unambiguous for a human reader even without context, and can also be disambiguated with a rather simple model of weights associated with compound parts.

We intend to conduct some additional experiments which include probabilities for segment pairs. This should improve the results for cases of high segment ambiguity (e.g. *Auto* (car) vs. *auto* (self)), but given the quite high precision with the simple model, we cannot expect a substantial increase of the overall precision.

Future work will show if similar results can be obtained for other compounding languages, such as Dutch, Swedish, Finnish, Hungarian, Greek, etc.

References

1. Schiller, A.: Xerox Finite-State Morphological Analyzer for German. on-line demo: http://www.xrce.xerox.com/competencies/content-analysis/demos/german (2004)
2. Baroni, M., Matiasek, J., Trost, H.: Predicting the Components of German Nominal Compounds. In: Proceedings of ECAI-2002, Amsterdam, IOS Press (2002) 470–474
3. Beesley, K.R., Karttunen, L.: Finite State Morphology. CSLI Studies in Computational Linguistics. CSLI Publications (2003)

[4] The 100% recall with the baseline FST results from the construction of the test and training data obtained by applying this FST on a word list and then disambiguated by hand.

4. Kempe, A., Baeijs, C., Gaál, T., Guingne, F., Nicart, F.: WFSC - A new weighted finite state compiler. In: Proceedings of CIAA-03. Volume 2759 of Lecture Notes in Computer Science., Santa Barbara, CA, USA, Springer Verlag (2003) 108–119
5. Rackow, U., Dagan, I., Schwall, U.: Automatic Translation of Noun Compounds. In: Proceedings of COLING-92, Nantes (1992)
6. Koehn, P., Knight, K.: Empirical Methods for Compound Splitting. In: Proceedings of ECAI-2003, Budapest, Hungary (2003)
7. Monz, C., de Rijk, M.: Shallow Morphological Analysis in Monolingual Information Retrieval for Dutch, German and Italian. In Peters, C., ed.: Proceedings of CLEF 2001. LNCS, Springer (2002)
8. Mohri, M., Pereira, F., Riley, M.: Weighted Automata in Text and Speech Processing. In: Proceedings ECAI-96, Workshop on Extended finite state models of language, Budapest, Hungary (1996)
9. Karttunen, L.: Applications of Finite-State Transducers in Natural Language Processing. In: Proceedings of CIAA-2000, Springer Verlag (2000)
10. Kempe, A.: NLP Applications based on weighted multi tape automata. In: Proceedings of 11th Conference TALN, Fes, Morocco (2004)