Automated Morphological Segmentation and Evaluation

Uwe D. Reichel, Karl Weilhammer

Department of Phonetics and Speech Communication
University of Munich, Schellingstr.3, 80799 Munich, Germany
{reichelu, weilkar}@phonetik.uni-muenchen.de

Abstract

In this paper we introduce (i) a new method for morphological segmentation of part of speech labelled German words and (ii) some measures related to the MDL principle for evaluation of morphological segmentations. The segmentation algorithm is capable to discover hierarchical structure and to retrieve morphemes not present as simplex forms in the data. It achieved 87 % recall and 98 % precision. Regarding MDL based evaluation, a linear combination of vocabulary size and size of reduced deterministic finite state automata matching exactly the segmentation outputs turned out to be an appropriate measure to rank segmentation models according to their quality.

1. Introduction

Morphological segmentation systems can roughly be divided in three major classes: (i) handmade systems, (ii) models that have been learned under supervision, and (iii) models induced without supervision.

Hand-crafted systems work on the basis of linguistic knowledge provided by an expert. A possible implementation of such systems is given by finite-state transducers in the framework of two-level morphology (Koskenniemi, 1983) which relates letter sequences on the surface level to more abstract morpheme sequences on the lexical level.

Supervised learning methods are trained on already segmented words. An example would be the connectionist approach of Rumelhart et al. (1986).

Finally, unsupervised induction is provided by raw data to be structured by certain principles depending on the respective approach, that might for example be based on minimum description length (Goldsmith, 2001).

For German there exists already a variety of morphology systems. An overview over some of these systems that participated at the first *Morpholympics*, can be found in Hausser (1996).

2. Goal of the paper

In the following we will present a method for morphological segmentation for German. It is based on knowledge about inflection, derivation, and morphotactics, and part of speech (POS) information. Apart from POS the knowledge is prepared manually, but opposed to fully hand-crafted systems, this preparation is much less time consuming and can easily be carried out by consulting some standard grammar. In contrast to most of the supervised and unsupervised approaches, that can just cope with certain aspects of morphology, it provides full morphologic segmentation. It includes the capabilities to generate allomorphs, to deal with hierarchical structure, and to retrieve morphemes not given in isolation in the input data.

Further, measures were adopted from unsupervised morphology induction within the MDL framework in order to evaluate several segmentations of different qualities.

3. Data

The input of our segmentation method consists of automatically POS labelled[1] text material. We used the SI1000P corpus, which is part of the *Bavarian Archive for Speech Signals*[2] and contains 1000 German broadcast sentences (5708 types[3]).

4. A method for automatic morphological segmentation

German as a highly inflectional language uses both bound morphemes and stem modifications for inflection and derivation. In addition it is very productive in combining words to compounds that can have a (theoretically infinite) degree of complexity. Our algorithm that is described below tries to cope with these difficulties. It consists of two main steps: lexicon construction and segmentation.

4.1. Lexicon construction

The lexicon initially comprises bound morphemes such as inflectional and derivational suffixes, prefixes and linking morphemes as being given by a standard grammar (Klosa et al., 1998). It is then augmented by the input data, applying stemming and allomorph generation whenever possible. All types as well as the retrieved stems and allomorphs are stored in the lexicon together with their corresponding POS and morpheme class respectively. The storage of allomorphs in the lexicon simplifies segmentation later on, because it can be achieved within the limits of simple concatenative morphology (i.e. connecting morphemes without taking care of the context sensitivity of their realization, as it would be the case in two-level morphology).[4]

Stemming In order to reduce the amount of errors resulting from our partly string based stemming operations (some of them are described here), a potential stem must contain at least three letters including at least one vowel in all cases.

[1] http://www.ims.uni-stuttgart.de/projekte/corplex

[2] http://www.phonetik.uni-muenchen.de

[3] Types with different POS are distinguished.

[4] The disadvantage of such a simplification is of course the system's incapability for morpheme sequence **generation** due to overgeneralization.

Verbs Inflectional suffixes for verbs are separated by simple greedy pattern matching. The longest suffix stored in the lexicon is tested first, if it is a final substring of the verb. If a matching potential suffix starts with a 't', it is only accepted if no infinite form in the corpus indicates, that this letter is part of the stem (e.g. in *betet* (*prays*) the first 't' remains part of the stem, if *beten* (*to pray*) is found in the data, that is clearly divided into *bet+en*).

Adjectives Apart from inflectional suffixes for adjectives, comparative morphemes are also separated.

Nouns Concerning nouns, stemming is more complicated (e.g. *en* in *Bahnen* (*tracks*) indicates plural, whereas in *Rasen* (*lawn*) it is part of the stem). Therefore we only define an initial substring as a stem, if it cooccurs with at least two different inflectional morphemes (not starting with the same letter) of the same declination class (e.g. *Bild**s**, Bild**er***). For some declination classes also umlaut allomorphs are considered.

Derivational variants Initial substrings coocurring with at least two different derivational suffixes were also included in the lexicon and classified as verb stems.

Allomorph generation For comparative adjectives containing an umlaut, allomorphs with the corresponding vowels are added to the lexicon. Furthermore some of the ablaut paradigms for strong verbs (that can be identified by their endings in perfect participle and some past forms) allow secure allomorph generation by vowel replacement (e.g. *ge+l**au**f+en* $\longrightarrow$ *l**ie**f*)

4.2. Segmentation

Basic algorithm (cf. Figure 1) Each type *w* of the input text is recursively divided into prefixes and suffixes from left to right until a permitted segmentation is achieved or until the end of *w* is reached.

In the course of the recursion a boundary dividing the current string in prefix and suffix is accepted if (i) the prefix is found in the lexicon, (ii) there exists a permitted segmentation for the suffix or (if not) the suffix is found in the lexicon, (iii) the sequence 'prefix class + class of first suffix segment' is not in conflict with German morphotactics (cf. Table 1) and (iv) the class of the last morpheme is in correspondence with w's POS (cf. Table 1).

noun suffix	linking morpheme, noun suffix, noun inflection
finite verb	verb inflection, verb stem

Table 1: Sample entries for morphotactics (top: right hand morpheme classes can follow left hand ones) and POS–class(last suffix) compatibility (bottom: types with left hand POS must end with a morpheme of right hand class)

Hierarchical structure If a segmentation was successful, it is recursively reapplied for each found segment. This second, finer grained segmentation can be used to discover hierarchical structure. For example applying the function 'segmentation' (cf. Figure 1) to the word *Samstagnachmittag* (*Saturday afternoon*) leads to the segments *Samstag+nachmittag*, leaving *nachmittag* undivided because

```
global list morphs := [ ]
function segmentation(str) ≡
  for i:=2 to length(str)-1
    [ prfx, sfx ] := split(str) at position i
    if prfx ∈ lexicon
      if (segmentation(sfx) and
          morphotactics_ok(class(prfx), class(first_sfx)))
            morphs := [prfx, morphs]
            return 1
      elseif (sfx ∈ lexicon and
          morphotactics_ok(class(prfx), class(sfx)) and
          compatible(class(sfx), pos(word)))
            morphs := [prfx, sfx, morphs]
            return 1
      endif
    endif
  endfor
  return 0
```

Figure 1: Algorithm for morphological segmentation (morphotactics_ok is applied for all combinations of possible classes of *prfx* and *sfx*)

according to morphotactics the preposition *nach* cannot follow the noun *Samstag*. Reapplying 'segmentation' to the segments in isolation leads to the desired splitting in [*Samstag*]+[*nach+mit+tag*] and reflects the different connection strengths between the parts of the compound.

Discovery of new morphemes If a complete segmentation of type *w* fails, it is tested whether *w* is partially segmentable, that is if the function 'segmentation' can be applied successfully to substrings of *w*. If just one substring *s* of *w* remains unsegmentable and if *s* is found in several different segmentable (or lexical) environments it is treated as a (not classified) morpheme, and affected tokens are segmented accordingly. This method has turned out to be a good way to get along with data sparseness. In our data for example *Chef* (*boss*), did not occur as a simplex, but has been found in five different segmentable (or lexical) contexts, among them *Chefpilot* (*chief pilot*) and *Regierungschef* (*head of government*) and could therefore be considered as a morpheme.

4.3. Evaluation by hand

The model's performance was manually evaluated for 1400 types. The types were chosen randomly from those, that are potentially segmentable due to their POS. Omissions and false insertions of segment boundaries were counted, a boundary displacement was punished by adding one omission and one insertion. The model yields 87 % recall and 98 % precision, indicating a rather restrained placement of boundaries. For 80 % of the types the analysis was completely correct. As the corpus used here is rather small recall should be improved by adding more input data containing more simplex forms. For example, *Flugzeug* (*aero plane*) could not be segmented, because *Zeug* was not part of the corpus.

5. Measures for automated evaluation

Several algorithms for unsupervised morphological segmentation are based on the MDL principle (Rissanen, 1989). According to this principle initially used in coding theory, the quality of an explanatory model for a data set depends (i) on the degree of data compression, that can be interpreted as vocabulary size reduction, and (ii) on the model's compactness. We adopted this framework in order to evaluate morphological segmentations.

In order to model the data size aspect of the MDL principle we tested vocabulary size and entropy measures for the evaluation of segmentations of different quality. The model size aspect is considered here by comparing the size of reduced deterministic finite state automata resulting from the respective segmentations. Both aspects are finally integrated by linear combination.

Evaluated models The following segmentations have been compared by the measures described above: (i) the complete segmentation found by our algorithm as described in section 4., (ii) partial outputs of this segmentation, (iii) segmentation by chance into the mean number of morphs per type (as found in (i)), and (iv) letter by letter segmentation. Partial segmentations were obtained by merging the last n morphs of each type, n ranging from 1 to 9, which was the highest number of morphs per type found in the results of the complete segmentation.[5] If n is equal or greater than the number of morphs for type w, then w will not be segmented. Such, $n = 9$ reflects the case that the data has not been segmented at all.

5.1. Data size

Morphological segmentation reduces vocabulary size. This fact is also reflected in the entropy measure, which is defined for the data set X as follows:

$$H(X) = -\sum_{x \in X} P(x) \log_2 P(x) [\text{bit}] \quad (1)$$

It gives the amount of bits needed to encode the data. $P(x)$ is the probability of the item $x \in X$, where X was the set of unigrams and the set of bigrams. Entropy gets low if the probabilities assigned to the x's get high, which can be achieved by the reduction of vocabulary size due to morphological segmentation. In general it can be stated: the better the segmentation, the higher the reduction, the lower the entropy.

Results Figures 2 and 3 show the relation between model quality and data size represented as vocabulary size and n-gram entropies. The 11 models are lined up on the x-axis from left to right as follows: letter by letter segmentation, segmentation by chance, no segmentation, and segmentations with an increasing number of segments allowed for each type. It can be seen that vocabulary size and unigram entropy correspond better to the model quality than bigram entropy, for which the small amount of data might be responsible.

[5] 9 morphs have been found in *Un+ab+häng+ig+keit+s+er+klär+ung* (*declaration of independence*).

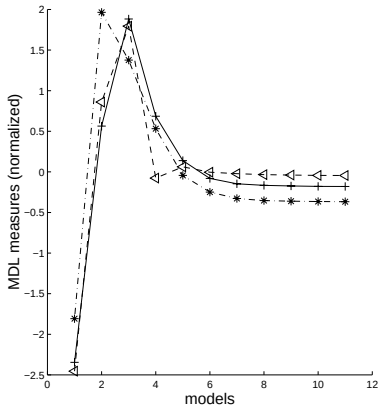

Figure 2: Entropies (normalized to mean=0 and standard deviation=1) for segmentation models increasing in quality from left to right; + unigram, ◁ bigram, ∗ vocabulary size

Including weights To cope with the fact that, due to its small vocabulary size, letter by letter segmentations lead to the best entropy values, we tested the incorporation of weights in the evaluation measure in order to penalize multiple occurrences of a morpheme within the same word. The contribution of each morpheme m is weighted by $c^{\frac{n}{o}}$, where n is the number of types that m is part of, o is the number of types in which m occurs just once, and c is a language dependent constant (here: 16, chosen heuristically). This penalty reflects the assumption that in languages like German (but not universally of course) morpheme repetitions are uncommon. This is also reflected in our data, where just 0.8% of the types showed such a repetition. As can be seen in Figure 3, this weighting has no influence on the ranking of segmentations of different quality except for the letter by letter case, which is degraded, showing a high amount of repetitions within one type.

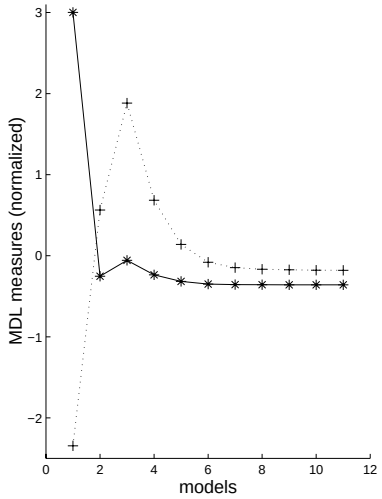

Figure 3: Unigram Entropy (normalized to mean=0 and standard deviation=1) – with (∗) and without (+) weights – for segmentation models increasing in quality from left to right

5.2. Model size

Reduced deterministic finite state automata (DFSA) derived directly from the segmented data can be considered as abstract representations of the segmentation model, because regarding their output they are equivalent. Model comparison is facilitated by transforming beforehand each model in such a DFSA in order to get an uniform representation. Model size as being a quality feature of the MDL principle can be compared this way by just counting the automata's states.

From the segmented data we constructed reduced DFSAs accepting exactly the given segmentations. The construction was accomplished the way tries are constructed (cf. Figure 4). A trie is a tree storing strings in which there is one node for every common prefix. The trie's root was appointed as the automaton's initial state, the leaves as the final states. The automaton was then minimized by a common method of iteratively grouping indistinguishable states in finer grained partitions.

```
highest_state := 0
s := 0   %initial state
Q := [s]   %set of states
F := [ ]   %set of final states
foreach word of wordlist
    state := s
    foreach morph of word
        if not defined δ(state,morph)    %transition
            highest_state := highest_state+1
            δ(state,morph) := highest_state
            Q := [Q, highest_state]
        state := δ(state,morph)
    end
    F := [F, state]
end
```

Figure 4: Algorithm for DFSA construction (as a trie)

Results As expected, Figure 5 shows that model size gets large for letter by letter segmentation (model 1), and after a minimum at the no-segmentation model (model 3), it increases again. The first observation is simply explained by the fact that letters can be combined much more freely than morphemes, resulting in much larger models. Secondly, a DFSA only distinguishing between types just needs two states, and finally, with increasing segmentation capability the number of needed states also increases.

5.3. Linear Combination

We tried to integrate data and model size aspects to a single quality measure by linear combination. The weights were estimated by least square fit for a decreasing linear function of slope -1. It turned out that – decreasing monotonically – the combination of DFSA size and vocabulary size reflects segmentation quality more appropriately than the combination of DFSA size and entropy. The resulting measure is thus:

$$1.55 \cdot \text{vocabulary size} + 1.26 \cdot \text{DFSA size} \qquad (2)$$

As can be seen in Figure 5 the models are ranked according to their quality. With increasing quality the curve gets flatter due to the decreasing amount of improvement (much lesser improvement is achieved for example, if the number of allowed morphemes per type is incremented from 7 to 8 than from 2 to 3, because much lesser types are concerned).

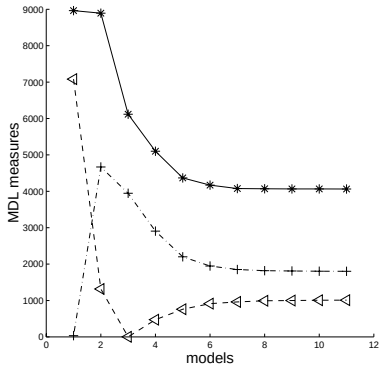

Figure 5: vocabulary (+) and model size ($\triangleleft$) and their linear combination ($*$) for segmentation models increasing in quality from left to right

6. Conclusion

In this paper we have introduced a new segmentation method that needs just very little manually prepared linguistic knowledge. It has to be tested if its performance increases when being confronted with a larger corpus. On one hand it has access to more simplex forms, what should increase recall of segment boundaries. On the other hand it is like all rule based systems quite vulnerable to noise, that arises for example from POS tagging errors. Heuristics would have to be added to cope with this problem.

The linear combination of model size, represented as the number of states of an equivalent reduced DFSA, and vocabulary size given after the application of this model has proved to be an adequate measure to evaluate the segmentation models used in this study, and a more general usability should be tested in future studies.

7. References

Goldsmith, J., 2001. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27.

Hausser, R. (ed.), 1996. *Linguistische Verifikation. Dokumentation zur ersten Morpholympics*. Tübingen: Niemeyer.

Klosa, A., W. Scholze-Stubenrecht, and M. Wermke (eds.), 1998. *Duden, Die Grammatik*. Mannheim: Dudenverlag, 6th edition.

Koskenniemi, K., 1983. Two-level morphology: A general computational model of word-form recognition and production. Technical Report 11, Dep. of General Linguistics, Helsinki.

Rissanen, J., 1989. *Stochastic Complexity in Statistical Inquiry*. Singapore: World Scientific Publishing Co.

Rumelhart, D. E. and J. L. McClelland, 1986. On learning the past tenses of english verbs. In J. L. McClelland, D. E. Rumelhart, and the PDP Research Group (eds.), *Parallel distributed processing: Explorations in the microstructure of cognition*. Cambridge: MIT Press.