A Language-Independent Unsupervised Model for Morphological Segmentation

Vera Demberg
School of Informatics
University of Edinburgh
Edinburgh, EH8 9LW, GB
v.demberg@sms.ed.ac.uk

Abstract

Morphological segmentation has been shown to be beneficial to a range of NLP tasks such as machine translation, speech recognition, speech synthesis and information retrieval. Recently, a number of approaches to unsupervised morphological segmentation have been proposed. This paper describes an algorithm that draws from previous approaches and combines them into a simple model for morphological segmentation that outperforms other approaches on English and German, and also yields good results on agglutinative languages such as Finnish and Turkish. We also propose a method for detecting variation within stems in an unsupervised fashion. The segmentation quality reached with the new algorithm is good enough to improve grapheme-to-phoneme conversion.

1 Introduction

Morphological segmentation has been shown to be beneficial to a number of NLP tasks such as machine translation (Goldwater and McClosky, 2005), speech recognition (Kurimo et al., 2006), information retrieval (Monz and de Rijke, 2002) and question answering. Segmenting a word into meaning-bearing units is particularly interesting for morphologically complex languages where words can be composed of several morphemes through inflection, derivation and composition. Data sparseness for such languages can be significantly decreased when words are decomposed morphologically. There exist a number of rule-based morphological segmentation systems for a range of languages. However, expert knowledge and labour are expensive, and the analyzers must be updated on a regular basis in order to cope with language change (the emergence of new words and their inflections). One might argue that unsupervised algorithms are not an interesting option from the engineering point of view, because rule-based systems usually lead to better results. However, segmentations from an unsupervised algorithm that is language-independent are “cheap”, because the only resource needed is unannotated text. If such an unsupervised system reaches a performance level that is good enough to help another task, it can constitute an attractive additional component.

Recently, a number of approaches to unsupervised morphological segmentation have been proposed. These algorithms autonomously discover morpheme segmentations in unannotated text corpora. Here we describe a modification of one such unsupervised algorithm, RePortS (Keshava and Pitler, 2006). The RePortS algorithm performed best on English in a recent competition on unsupervised morphological segmentation (Kurimo et al., 2006), but had very low recall on morphologically more complex languages like German, Finnish or Turkish. We add a new step designed to achieve higher recall on morphologically complex languages and propose a method for identifying related stems that underwent regular non-concatenative morphological processes such as umlauting or ablauting, as well as morphological alternations along morpheme boundaries.

The paper is structured as follows: Section

Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, pages 920–927,
Prague, Czech Republic, June 2007.

2 discusses the relationship between language-dependency and the level of supervision of a learning algorithm. We then give an outline of the main steps of the RePortS algorithm in section 3 and explain the modifications to the original algorithm in section 4. Section 5 compares results for different languages, quantifies the gains from the modifications on the algorithm and evaluates the algorithm on a grapheme-to-phoneme conversion task. We finally summarize our results in section 6.

2 Previous Work

The world's languages can be classified according to their morphology into isolating languages (little or no morphology, e.g. Chinese), agglutinative languages (where a word can be decomposed into a large number of morphemes, e.g. Turkish) and inflectional languages (morphemes are fused together, e.g. Latin).

Phenomena that are difficult to cope with for many of the unsupervised algorithms are non-concatenative processes such as vowel harmonization, ablauting and umlauting, or modifications at the boundaries of morphemes, as well as infixation (e.g. in Tagalog: sulat 'write', s-um-ulat 'wrote', s-in-ulat 'was written'), circumfixation (e.g. in German: mach-en 'do', ge-mach-t 'done'), the Arabic broken plural or reduplications (e.g. in Pingelapese: mejr 'to sleep', mejmejr 'sleeping', mejmejmejr 'still sleeping'). For words that are subject to one of the above processes it is not trivial to automatically group related words and detect regular transformational patterns.

A range of automated algorithms for morphological analysis cope with concatenative phenomena, and base their mechanics on statistics about hypothesized stems and affixes. These approaches can be further categorized into ones that use conditional entropy between letters to detect segment boundaries (Harris, 1955; Hafer and Weiss, 1974; Déjean, 1998; Monson et al., 2004; Bernhard, 2006; Keshava and Pitler, 2006; Bordag, 2006), approaches that use minimal description length and thereby minimize the size of the lexicon as measured in entries and links between the entries to constitute a word form (Goldsmith, 2001; Creutz and Lagus, 2006). These two types of approaches very closely tie the orthographic form of the word to the morphemes. They are thus not well-suited for coping with stem changes or modifications at the edges of morphemes. Only very few approaches have addressed word internal variations (Yarowski and Wicentowski, 2000; Neuvel and Fulop, 2002).

A popular and effective approach for detecting inflectional paradigms and filter affix lists is to cluster together affixes or regular transformational patterns that occur with the same stem (Monson et al., 2004; Goldsmith, 2001; Gaussier, 1999; Schone and Jurafsky, 2000; Yarowski and Wicentowski, 2000; Neuvel and Fulop, 2002; Jacquemin, 1997). We draw from this idea of clustering in order to detect orthographic variants of stems; see Section 4.3.

A few approaches also take into account syntactic and semantic information from the context the word occurs (Schone and Jurafsky, 2000; Bordag, 2006; Yarowski and Wicentowski, 2000; Jacquemin, 1997). Exploiting semantic and syntactic information is very attractive because it adds an additional dimension, but these approaches have to cope with more severe data sparseness issues than approaches that emphasize word-internal cues, and they can be computationally expensive, especially when they use LSA.

The original RePortS algorithm assumes morphology to be concatenative, and specializes on prefixation and suffixation, like most of the above approaches, which were developed and implemented for English (Goldsmith, 2001; Schone and Jurafsky, 2000; Neuvel and Fulop, 2002; Yarowski and Wicentowski, 2000; Gaussier, 1999). However, many languages are morphologically more complex. For example in German, an algorithm also needs to cope with compounding, and in Turkish words can be very long and complex. We therefore extended the original RePortS algorithm to be better adapted to complex morphology and suggest a method for coping with stem variation. These modifications render the algorithm more language-independent and thereby make it attractive for applying to other languages as well.

3 The RePortS Algorithm

On English, the RePortS algorithm clearly outperformed all other systems in Morpho Challenge

2005[1] (Kurimo et al., 2006), obtaining an F-measure of 76.8% (76.2% prec., 77.4% recall). The next best system obtained an F-score of 69%. However, the algorithm does not perform as well on other languages (Turkish, Finnish, German) due to low recall (see (Keshava and Pitler, 2006) and (Demberg, 2006), p. 47).

There are three main steps in the algorithm. First, the data is structured in two trees, which provide the basis for efficient calculation of transitional probabilities of a letter given its context. The second step is the affix acquisition step, during which a set of morphemes is identified from the corpus data. The third step uses these morphemes to segment words.

3.1 Data Structure

The data is stored in two trees, the forward tree and the backward tree. Branches correspond to letters, and nodes are annotated with the total corpus frequency of the letter sequence from the root of the tree up to the node. During the affix identification process, the forward tree is used for discovering suffixes by calculating the probability of seeing a certain letter given the previous letters of the word. The backward tree is used to determine the probability of a letter given the following letters of a word in order to find prefixes. If the transitional probability is high, the word should not be split, whereas low probability is a good indicator of a morpheme boundary. In such a tree, stems tend to stay together in long unary branches, while the branching factor is high in places where morpheme boundaries occur.

The underlying idea of exploiting "Letter Successor Variety" was first proposed in (Harris, 1955), and has since been used in a number of morphemic segmentation algorithms (Hafer and Weiss, 1974; Bernhard, 2006; Bordag, 2006).

3.2 Finding Affixes

The second step is concerned with finding good affixes. The procedure is quite simple and can be divided into two subtasks. (1) generating all possible affixes and (2) validating them. The validation step is necessary to exclude bad affix candidates (e.g. letter sequences that occur together frequently such as *sch*, *spr* or *ch* in German or *sh*, *th*, *qu* in English). An affix is validated if all three criteria are satisfied for at least 5% of its occurrences:

1. The substring that remains after peeling off an affix is also a word in the lexicon.
2. The transitional probability between the second-last and the last stem letter is ≈ 1.
3. The transitional probability of the affix letter next to the stem is <1 (tolerance 0.02).

Finally, all affixes that are concatenations of two or more other suffixes (e.g., *-ungen* can be split up in *-ung* and *-en* in German) are removed. This step returns two lists of morphological segments. The prefix list contains prefixes as well as stems that usually occur at the beginning of words, while the suffix list contains suffixes and stems that occur at the end of words. In the remainder of the paper, we will refer to the content of these lists as "prefixes" and "suffixes", although they also include stems. There are several assumptions encoded in this procedure that are specific to English, and cause recall to be low for other languages: 1) all stems are valid words in the lexicon; 2) affixes occur at the beginning or end of words only; and 3) affixation does not change stems. In section 4, we propose ways of relaxing these assumptions to make this step less language-specific.

3.3 Segmenting Words

The final step is the complete segmentation of words given the list of affixes acquired in the previous step. The original RePortS algorithm uses a very simple method that peels off the most probable suffix that has a transitional probability smaller than 1, until no more affixes match or until less than half of the word remains. This last condition is problematic since it does not scale up well to languages with complex morphology. The same peeling-off process is executed for prefixes.

Although this method is better than using a heuristic such as 'always peel off the longest possible affix', because it takes into account probable sites of fractures in words, it is not sensitive to the affix context or the morphotactics of the language. Typical mistakes that arise from this condition are that inflectional suffixes, which can only occur word-finally, might be split off in the middle of a word after previously having peeled off a number of other suffixes.

[1] www.cis.hut.fi/morphochallenge2005/

4 Modifications and Extensions

4.1 Morpheme Acquisition

When we ran the original algorithm on a German data set, no suffixes were validated but reasonable prefix lists were found. The algorithm works fine for English suffixes – why does it fail on German? The algorithm's failure to detect German suffixes is caused by the invalid assumption that a stem must be a word in the corpus. German verb stems do not occur on their own (except for certain imperative forms). After stripping off the suffix of the verb *abholst* 'fetch', the remaining string *abhol* cannot be found in the lexicon. However, words like *abholen*, *abholt*, *abhole* or *Abholung* are part of the corpus. The same problem also occurs for German nouns.

Therefore, this first condition of the affix acquisition step needs to be replaced. We therefore introduced an additional step for building an intermediate stem candidate list into the affix acquisition process. The first condition is replaced by a condition that checks whether a stem is in the stem candidate list. This new stem candidate acquisition procedure comprises three steps:

Step 1: Creation of stem candidate list

All substrings that satisfy conditions 2 and 3 but not condition 1, are stored together with the set of affixes they occur with. This process is similar to the idea of registering signatures (Goldsmith, 2001; Neuvel and Fulop, 2002). For example, let us assume our corpus contains the words *Aufführender*, *Aufführung*, *aufführt* and *Aufführlaune* but not the stem itself, since *aufführ* 'act' is not a valid German word. Conditions 2 and 3 are met, because the transitional probability between *aufführ* and the next letter is low (there are a lot of different possible continuations) and the transitional probability $P(r|\textit{aufführ}) \approx 1$. The stem candidate *aufführ* is then stored together with the suffix candidates $\{$*ender*, *ung*, *en*, *t*, *laune*$\}$.

Step 2: Ranking candidate stems

There are two types of affix candidates: type-1 affix candidates are words that are contained in the data base as full words (those are due to compounding); type-2 affix candidates are inflectional and derivational suffixes. When ranking the stem candidates, we take into account the number of type-1 affix candidates and the average frequency of tpye-2 affix candidates.

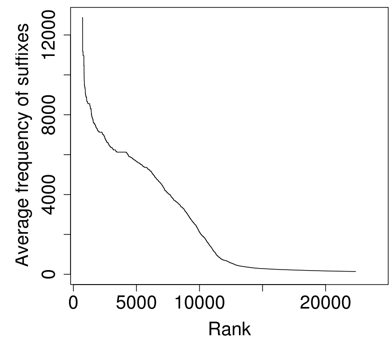

Figure 1: Determining the threshold for validating the best candidates from the stem candidate list.

The first condition has very good precision, similar to the original method. The morphemes found with this method are predominantly stem forms that occur in compounding or derivation (Kompositionsstämme and Derivationsstämme). The second condition enables us to differentiate between stems that occur with common suffixes (and therefore have high average frequencies), and pseudostems such as *runtersch* whose affix list contains many non-morphemes (e.g. *lucken*, *iebt*, *aute*). These non-morphemes are very rare since they are not generated by a regular process.

Step 3: Pruning

All stem candidates that occur less than three times are removed from the list. The remaining stem candidates are ordered according to the average frequency of their non-word suffixes. This criterion puts the high quality stem candidates (that occur with very common suffixes) to the top of the list. In order to obtain a high-precision stem list, it is necessary to cut the list of candidates at some point. The threshold for this is determined by the data: we choose the point at which the function of list-rank vs. score changes steepness (see Figure 1). This visual change of steepness corresponds to the point where potential stems found get more noisy because the strings with which they occur are not common affixes. We found the performance of the resulting morphological system to be quite stable ($\pm 1\%$ f-score) for any cutting point on the slope between 20% and 50% of the list (for the German data set ranks 4000 and 12000), but importantly before the function tails off. The threshold was also robust across the other languages and data sets.

4.2 Morphological Segmentation

As discussed in section 3.3, the original implementation of the algorithm iteratively chops off the most probable affixes at both edges of the word without taking into account the context of the affix. In morphologically complex languages, this context-blind approach often leads to suboptimal results, and also allows segmentations that are morphotactically impossible, such as inflectional suffixes in the middle of words. Another risk is that the letter sequence that is left after removing potential prefixes and suffixes from both ends is not a proper stem itself but just a single letter or vowel-less letter-sequence.

These problems can be solved by using a bi-gram language model to capture the morphotactic properties of a particular language. Instead of simply peeling off the most probable affixes from both ends of the word, all possible segmentations of the word are generated and ranked using the language model. The probabilities for the language model are learnt from a set of words that were segmented with the original simple approach. This bootstrapping allows us to ensure that the approach remains fully unsupervised. At the beginning and end of each word, an edge marker '#' is attached to the word. The model can then also acquire probabilities about which affixes occur most often at the edges of words.

Table 2 shows that filtering the segmentation results with the n-gram language model caused a significant improvement on the overall F-score for most languages, and led to significant changes in precision and recall. Whereas the original segmentation yielded balanced precision and recall (both 68%), the new filtering boosts precision to over 73%, with 64% recall. Which method is preferable (i.e. whether precision or recall is more important) is task-dependent.

In future work, we plan to draw on (Creutz and Lagus, 2006), who use a HMM with morphemic categories to impose morphotactic constraints. In such an approach, each element from the affix list is assigned with a certain probability to the underlying categories of "stem", "prefix" or "suffix", depending on the left and right perplexity of morphemes, as well as morpheme length and frequency. The transitional probabilities from one category to the next model the morphotactic rules of a language, which can thus be learnt automatically.

4.3 Learning Stem Variation

Stem variation through ablauting and umlauting (an English example is run–ran) is an interesting problem that cannot be captured by the algorithm outlined above, as variations take place within the morphemes. Stem variations can be context-dependent and do not constitute a morpheme in themselves. German umlauting and ablauting leads to data sparseness problems in morphological segmentation and affix acquisition. One problem is that affixes which usually cause ablauting or umlauting are very difficult to find. Typically, ablauted or umlauted stems are only seen with a very small number of different affixes, which means that the affix sets of such stems are divided into several unrelated subsets, causing the stem to be pruned from the stem candidate list. Secondly, ablauting and umlauting lead to low transitional probabilities at the positions in stems where these phenomena occur. Consider for example the affix set for the stem candidate *bockspr*, which contains the pseudoaffixes *ung*, *ünge* and *ingen*. The morphemes *sprung*, *sprüng* and *springen* are derived from the root *spring* 'to jump'. In the segmentation step this low transitional probability thus leads to oversegmentation.

We therefore investigated whether we can learn these regular stem variations automatically. A simple way to acquire the stem variations is to look at the suffix clusters which are calculated during the stem-acquisition step. When looking at the sets of substrings that are clustered together by having the same prefix, we found that they are often inflections of one another, because lexicalized compounds are used frequently in different inflectional variants. For example, we find *Trainingssprung* as well as *Trainingssprünge* in the corpus. The affix list of the stem candidate *trainings* thus contains the words *sprung* and *sprünge*. Edit distance can then be used to find differences between all words in a certain affix list. Pairs with small edit distances are stored and ranked by frequency. Regular transformation rules (e.g. ablauting and umlauting, $u \rightarrow ü..e$) occur at the top of the list and are automatically accepted as rules (see Table 1). This method allows us to not only find the relation between two words in the lexicon (*Sprung* and *Sprünge*) but also to automatically learn rules that can be applied to unknown words to check whether their variant is a word in the lexicon.

freq.	diff.	examples
1682	a ä..e	sack-säcke, brach-bräche, stark-stärke
344	a ä	sahen-sähen, garten-gärten
321	u ü..e	flug-flüge, bund-bünde
289	ä a..s	verträge-vertrages, pässe-passes
189	o ö..e	chor-chöre, strom-ströme, ?röhre-rohr
175	t en	setzt-setzen, bringt-bringen
168	a u	laden-luden, *damm-dumm
160	ß ss	läßt-lässt, mißbrauch-missbrauch
[. . .]		
136	a en	firma-firmen, thema-themen
[. . .]		
2	ß g	*fließen-fliegen, *laßt-lagt
2	um o	*studiums-studios

Table 1: Excerpts from the stem variation detection algorithm results. Morphologically unrelated word pairs are marked with an asterisk.

We integrated information about stem variation from the regular stem transformation rules (those with the highest frequencies) into the segmentation step by creating equivalence sets of letters. For example, the rule $u \rightarrow$ *ü..e* generates an equivalence set {ü, u}. These two letters then count as the same letter when calculating transitional probabilities. We evaluated the benefit of integrating stem variation information for German on the German CELEX data set, and achieved an improvement of 2% in recall, without any loss in precision (F-measure: 69.4%, Precision: 68.1%, Recall: 70.8%; values for RePortS-stems). For better comparability to other systems and languages, results reported in the next section refer to the system version that does not incorporate stem variation.

5 Evaluation

For evaluating the different versions of the algorithm on English, Turkish and Finnish, we used the training and test sets from MorphoChallenge to enable comparison with other systems. Performance of the algorithm on German was evaluated on 244k manually annotated words from CELEX because German was not included in the MorphoChallenge data.

Table 2 shows that the introduction of the stem candidate acquisition step led to much higher recall on German, Finnish and Turkish, but caused some losses in precision. For English, adding both components did not have a large effect on either precision or recall. This means that this component is well behaved, i.e. it improves performance on languages where the intermediate stem-acquisition step is needed, but does not impair results on other languages. Recall for Finnish is still very low. It can be improved (at the expense of precision) by selecting the analysis with the largest number of segments in the segmentation step. The results for this heuristic was only evaluated on a smaller test set (ca. 700 wds), hence marked with an asterisk in Table 2.

Lang.	alg.version	F-Meas.	Prec.	Recall
Eng[1]	original	76.8%	76.2%	77.4%
	stems	67.6%	62.9%	73.1%
	n-gram seg.	75.1%	74.4%	75.9%
Ger[2]	original	59.2%	71.1%	50.7%
	stems	68.4%	68.1%	68.6%
	n-gram seg.	68.9%	73.7%	64.6%
Tur[1]	original	54.2%	72.9%	43.1%
	stems	61.8%	65.9%	58.2%
	n-gram seg.	64.2%	65.2%	63.3%
Fin[1]	original	47.1%	84.5%	32.6%
	stems	56.6%	74.1%	45.8%
	n-gram seg.	58.9%	76.1%	48.1%
	max-split*	61.3%	66.3%	56.9%

Table 2: Performance of the algorithm with the modifications on different languages.
[1]MorphoChallenge Data, [2]German CELEX

The algorithm is very efficient: When trained on the 240m tokens of the German TAZ corpus, it takes up less than 1 GB of memory. The training phase takes approx. 5 min on a 2.4GHz machine, and the segmentation of the 250k test words takes 3 min for the version that does the simple segmentation and about 8 min for the version that generates all possible segmentations and uses the language model.

5.1 Comparison to other systems

This modified version of the algorithm performs second best for English (after original RePortS) and ranks third for Turkish (after Bernhards algorithm with 65.3% F-measure and Morfessor-Categories-MAP with 70.7%). On German, our method significantly outperformed the other unsupervised algorithms, see Table 3. While most of the systems compared here were developed for languages other than German, (Bordag, 2006) describes a system initially built for German. When trained on the "Projekt Deutscher Wortschatz" corpus which comprises 24 million sentences, it achieves an F-score of 61% (precision 60%, recall 62%[2]) when evaluated on the full CELEX corpus.

[2]Data from personal communication.

morphology	F-Meas.	Prec.	Recall
SMOR-disamb2	83.6%	87.1%	80.4%
ETI	79.5%	75.4%	84.1%
SMOR-disamb1	71.8%	95.4%	57.6%
RePortS-lm	68.8%	73.7%	64.6%
RePortS-stems	68.4%	68.1%	68.6%
best Bernhard	63.5%	64.9%	62.1%
Bordag	61.4%	60.6%	62.3%
orig. RePortS	59.2%	71.1%	50.7%
best Morfessor 1.0	52.6%	70.9%	41.8%

Table 3: Evaluating rule-based and data-based systems for morphological segmentation with respect to CELEX manual morphological annotation.

Rule-based systems are currently the most common approach to morphological decomposition and perform better at segmenting words than state-of-the-art unsupervised algorithms (see Table 3 for performance of state-of-the-art rule-based systems evaluated on the same data). Both the ETI[3] and the SMOR (Schmid et al., 2004) systems rely on a large lexicon and a set of rules. The SMOR system returns a set of analyses that can be disambiguated in different ways. For details refer to pp. 29–33 in (Demberg, 2006).

5.2 Evaluation on Grapheme-to-Phoneme Conversion

Morphological segmentation is not of value in itself – the question is whether it can help improve results on an application. Performance improvements due to morphological information have been reported for example in MT, information retrieval, and speech recognition. For the latter task, morphological segmentations from the unsupervised systems presented here have been shown to improve accuracy (Kurimo et al., 2006).

Another motivation for evaluating the system on a task rather than on manually annotated data is that linguistically motivated morphological segmentation is not necessarily the best possible segmentation for a certain task. Evaluation against a manually annotated corpus prefers segmentations that are closest to linguistically motivated analyses. Furthermore, it might be important for a certain task to find a particular type of morpheme boundaries (e.g. boundaries between stems), but for another task it might be very important to find boundaries between stems and suffixes. The standard evaluation procedure does not differentiate between the types of mistakes made. Finally, only evaluation on a task can provide information as to whether high precision or high recall is more important, therefore, the decision as to which version of the algorithm should be chosen can only be taken given a specific task.

[3] Eloquent Technology, Inc. (ETI) TTS system. www.mindspring.com/~ssshp/ssshp_cd/ss_eloq.htm

morphology	F-Meas. (CELEX)	PER (dt)
CELEX	100%	2.64%
ETI	79.5%	2.78%
SMOR-disamb2	83.0%	3.00%
SMOR-disamb1	71.8%	3.28%
RePortS-lm	68.8%	3.45%
no morphology		3.63%
orig. RePortS	59.2%	3.83%
Bernhard	63.5%	3.88%
RePortS-stem	68.4%	3.98%
Morfessor 1.0	52.6%	4.10%
Bordag	64.1%	4.38%

Table 4: F-measure for evaluation on manually annotated CELEX and phoneme error rate (PER) from g2p conversion using a decision tree (dt).

For these reasons we decided to evaluate the segmentation from the new versions of the RePortS algorithm on a German grapheme-to-phoneme (g2p) conversion task. The evaluation on this task is motivated by the fact that (Demberg, 2007) showed that good-quality morphological preprocessing can improve g2p conversion results. We here compare the effect of using our system's segmentations to a range of different morphological segmentations from other systems. We ran each of the rule-based systems (ETI, SMOR-disamb1, SMOR-disamb2) and the unsupervised algorithms (original RePortS, Bernhard, Morfessor 1.0, Bordag) on the CELEX data set and retrained our decision tree (an implementation based on (Lucassen and Mercer, 1984)) on the different morphological segmentations.

Table 4 shows the F-score of the different systems when evaluated on the manually annotated CELEX data (full data set) and the phoneme error rate (PER) for the g2p conversion algorithm when annotated with morphological boundaries (smaller test set, since the decision tree is a supervised method and needs training data). As we can see from the results, the distribution of precision and recall (see Table 3) has an important impact on the conversion quality: the RePortS version with higher precision significantly outperforms the other version on the task, although their F-measures are almost identical. Remarkably, the RePortS version that uses the filtering step is the only unsupervised system that beats the no-morphology baseline ($p < 0.0001$). While all other unsupervised systems tested here make the system perform worse than it would without morphological information, this new version improves accuracy on g2p conversion.

6 Conclusions

A significant improvement in F-score was achieved by three simple modifications to the RePortS algorithm: generating an intermediary high-precision stem candidate list, using a language model to disambiguate between alternative segmentations, and learning patterns for regular stem variation, which can then also be exploited for segmentation. These modifications improved results on four different languages considered: English, German, Turkish and Finnish, and achieved the best results reported so far for an unsupervised system for morphological segmentation on German. We showed that the new version of the algorithm is the only unsupervised system among the systems evaluated here that achieves sufficient quality to improve transcription performance on a grapheme-to-phoneme conversion task.

Acknowledgments

I would like to thank Emily Pitler and Samarth Keshava for making available the code of the RePortS algorithm, and Stefan Bordag and Delphine Bernhard for running their algorithms on the German data. Many thanks also to Matti Varjokallio for evaluating the data on the MorphoChallenge test sets for Finnish, Turkish and English. Furthermore, I am very grateful to Christoph Zwirello and Gregor Möhler for training the decision tree on the new morphological segmentation. I also want to thank Frank Keller and the ACL reviewers for valuable and insightful comments.

References

Delphine Bernhard. 2006. Unsupervised morphological segmentation based on segment predictability and word segments alignment. In *Proceedings of 2nd Pascal Challenges Workshop*, pages 19–24, Venice, Italy.

Stefan Bordag. 2006. Two-step approach to unsupervised morpheme segmentation. In *Proceedings of 2nd Pascal Challenges Workshop*, pages 25–29, Venice, Italy.

Mathias Creutz and Krista Lagus. 2006. Unsupervised models for morpheme segmentation and morphology learning. In *ACM Transaction on Speech and Language Processing*.

H. Déjean. 1998. Morphemes as necessary concepts for structures: Discovery from untagged corpora. In *Workshop on paradigms and Grounding in Natural Language Learning*, pages 295–299, Adelaide, Australia.

Vera Demberg. 2006. Letter-to-phoneme conversion for a German TTS-System. Master's thesis. *IMS, Univ. of Stuttgart*.

Vera Demberg. 2007. Phonological constraints and morphological preprocessing for grapheme-to-phoneme conversion. In *Proc. of ACL-2007*.

Eric Gaussier. 1999. Unsupervised learning of derivational morphology from inflectional lexicons. In *ACL '99 Workshop Proceedings*, University of Maryland.

CELEX German Linguistic User Guide, 1995. *Center for Lexical Information*. Max-Planck-Institut for Psycholinguistics, Nijmegen.

John Goldsmith. 2001. Unsupervised learning of the morphology of a natural language. *computational Linguistics*, 27(2):153–198, June.

S. Goldwater and D. McClosky. 2005. Improving statistical mt through morphological analysis. In *Proc. of EMNLP*.

Margaret A. Hafer and Stephen F. Weiss. 1974. Word segmentation by letter successor varieties. *Information Storage and Retrieval 10*, pages 371–385.

Zellig Harris. 1955. From phoneme to morpheme. *Language 31*, pages 190–222.

Christian Jacquemin. 1997. Guessing morphology from terms and corpora. In *Research and Development in Information Retrieval*, pages 156–165.

S. Keshava and E. Pitler. 2006. A simpler, intuitive approach to morpheme induction. In *Proceedings of 2nd Pascal Challenges Workshop*, pages 31–35, Venice, Italy.

M. Kurimo, M. Creutz, M. Varjokallio, E. Arisoy, and M. Saraclar. 2006. Unsupervsied segmentation of words into morphemes – Challenge 2005: An introduction and evaluation report. In *Proc. of 2nd Pascal Challenges Workshop*, Italy.

J. Lucassen and R. Mercer. 1984. An information theoretic approach to the automatic determination of phonemic baseforms. In *ICASSP 9*.

C. Monson, A. Lavie, J. Carbonell, and L. Levin. 2004. Unsupervised induction of natural language morphology inflection classes. In *Proceedings of the Seventh Meeting of ACL-SIGPHON*, pages 52–61, Barcelona, Spain.

C. Monz and M. de Rijke. 2002. Shallow morphological analysis in monolingual information retrieval for Dutch, German, and Italian. In *Proceedings CLEF 2001, LNCS 2406*.

Sylvain Neuvel and Sean Fulop. 2002. Unsupervised learning of morphology without morphemes. In *Proc. of the Wshp on Morphological and Phonological Learning, ACL Pub.*

Helmut Schmid, Arne Fitschen, and Ulrich Heid. 2004. SMOR: A German computational morphology covering derivation, composition and inflection. In *Proc. of LREC*.

Patrick Schone and Daniel Jurafsky. 2000. Knowledge-free induction of morphology using latent semantic analysis. In *Proc. of CoNLL-2000 and LLL-2000*, Lisbon, Portugal.

Tageszeitung (TAZ) Corpus. Contrapress Media GmbH. `https://www.taz.de/pt/.etc/nf/dvd`.

David Yarowski and Richard Wicentowski. 2000. Minimally supervised morphological analysis by multimodal alignment. In *Proceedings of ACL 2000*, Hong Kong.