PermA and Balloon: Tools for string alignment and text processing

Uwe D. Reichel

Institute of Phonetics and Speech Processing, University of Munich, Germany

reichelu@phonetik.uni-muenchen.de

Abstract

Two online research tools are presented in this paper: *PermA*, a general-purpose string aligner which can for example be used for grapheme-to-phoneme and phoneme-to-phoneme alignment, and *Balloon*, a text processing toolkit for German and English providing components for part-of-speech tagging, morphological analyses, and grapheme-to-phoneme conversion including syllabification and word-stress assignment. The general architectures of these tools are introduced with a focus on recent improvements concerning the alignment cost function derivation and word stress assignment.

Index Terms: alignment, grapheme-to-phoneme conversion, part-of-speech tagging, morphology, word-stress assignment, tools

1. Introduction

One major aim of the currently ongoing CLARIN-D [1] project is to provide webservices to facilitate research in humanities. In this context two online tools for alignment and text processing made available for research purpose will be introduced in the following sections with respect to their underlying algorithms.

Alignment By alignment corresponding parts of parallel symbol sequences are mutually linked. Typical speech science scenarios are the alignment of grapheme and phoneme sequences stored in a pronunciation dictionary or to align parallel broad canonic and narrow spontaneous speech transcriptions. In speech technology this serves to provide training material for grapheme-to-phoneme (G2P) and for phoneme-to-phoneme (P2P) conversion the latter for mapping canonic pronunciation onto a more natural transcription accounting for connected speech processes. In fundamental linguistic research it can be used to infer phonological rules transforming canonic to spontaneous speech.

Text processing Some elementary text processing steps as part-of-speech (POS) tagging and G2P conversion are captured by the toolkit introduced in this paper. POS tagging serves as a starting point for syntactical analyses. G2P conversion can be used for phonological analyses for example to derive phoneme statistics and phonotactic models.

This paper shortly addresses the tool components already introduced in former publications and focuses on new developments, as are the alignment cost function derivation and the word stress assignment procedure.

2. *PermA* – A probabilistic aligner

2.1. General architecture

PermA is a probabilistic aligner allowing for a uniform modeling of all edit operations. It is a further development of the *CoocA* aligner introduced in [2]. As usually *PermA* considers the alignment of two sequences v and w as a task to transform v to w by a minimum sum of edit costs, which is known as the *Levenshtein distance*. This minimisation task is solved algorithmically by means of dynamic programming by a method proposed by [3]. Hereby the standard basic edit operations *substitution, insertion, and deletion* are used.

2.2. Cost function

All edit costs are defined in terms of conditional probabilities reflecting symbol co-occurrences.

Notation In the subsequent sections v and w are the sequences to be aligned by transforming v to w with minimum costs, |v| denoting the length of v. c is a cost function, $c(v_i, w_j)$ denotes the cost to substitute the i-th symbol v by the j-th symbol of w, $c(v_i, \_)$ is the cost to delete v_i, and $c(\_, w_j)$ is the cost to insert w_j.

2.2.1. Substitutions

Substitution costs are defined as follows:

$$c(v_i, w_j) = \begin{cases} 0 & : \text{ equal}(v_i, w_j) \wedge v_i, w_j \in D \\ 1 - P(w_j|v_i) & : \text{ else.} \end{cases} \quad (1)$$

Zero-substitutions are only supported if the elements of v and w come from the same shared vocabulary D, which is generally true for P2P alignment, but not for G2P alignment, since for example in German the grapheme <x> is not mapped to the phoneme /x/ but to /k s/. Substitution costs for unequal (v_i, w_j)–pairs are defined in terms of conditional probabilities, which are calculated by maximum likelihood estimation: $P(w_j|v_i) = \frac{\#(v_i,w_j)}{\#(v_i)}$. As shown in Figure 1, in order to derive the

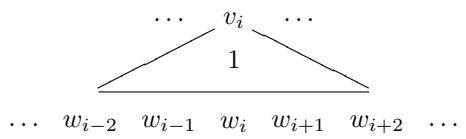

Figure 1: *Distribution of co-occurrence counts in a triangular window of area 1.*

s	c	h	u	l	e
_	_	S	u:	l	@
_	S	_	u:	l	@
S	_	_	u:	l	@
		$\cdots$			

Figure 2: *Permutation sample of empty symbols for uniform treatment of substitutions, insertions and deletions.*

co-occurrence statistics counts are incremented by centering a triangular window of a defined length (usually 5, but depending on the amount of the training material) and area 1 on v_i.

2.2.2. Deletions, insertions

Previous probabilistic cost function definitions did not allow for uniform handling of substitution, deletion, and insertion costs. Instead several heuristics have been exploited to model deletions and insertions. A review of these heuristics is given in [4]. *PermA* in contrast allows for a uniform modelling of all editing costs defining deletion and insertion costs as:

$$c(v_i, \_) = 1 - P(\_|v_i) \quad (2)$$

$$c(\_, w_j) = 1 - P(w_j|\_) \quad (3)$$

The counts for the maximum likelihood estimates $P(\_|v_i)$ and $P(w_j|\_)$ are derived the following way: Whenever v and w differ in length indicating deletions or insertions they are set to equal length by padding empty _symbols to the shorter sequence. Since the correct location of these _symbols is not known, a permutation of these symbols is carried out as illustrated in Figure 2. For each instance co-occurrence counts are incremented as explained in the previous section and the final increments are normalised by the number of permutation instances. To account for realistic alignment patterns and for low computational costs permutation can be constrained to a maximum allowed number of adjacent _–symbols, 2 would for example be a reasonable choice in German G2P alignment relating <sch> and /S/.

2.3. Evaluation

PermA evaluation on 450 word types with differing canonic and spontaneous transcriptions yielded a word error rate of 10.64% for grapheme-to-phoneme and of 1.31% for phoneme-to-phoneme alignment.

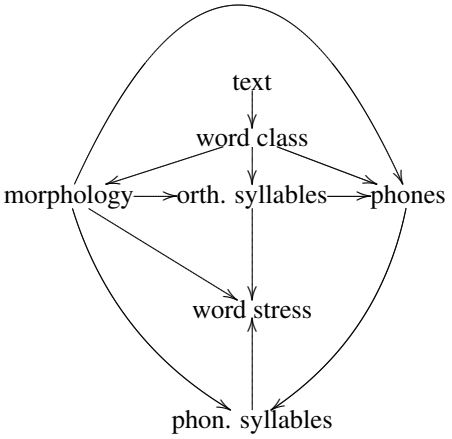

Figure 3: Balloon *information flow.*

3. *Balloon*: A text processing toolkit

The *Balloon* text processing toolkit contains the following functionalities: text normalisation, part-of-speech (POS) tagging, morphological analyses, and grapheme-to-phoneme conversion including syllabification and word stress assignment. The processing steps are illustrated in Figure 3.

3.1. Text normalisation

Several text normalisation steps are carried out by means of finite state techniques and heuristics [5]. Amongst others, normalisation includes the expansion of abbreviations and number-to-letter conversions, the latter complemented by a number disambiguation mechanism to account for year and phone number realisations. The expansion of German ordinal numbers includes congruent inflection.

3.2. Part-of-speech tagging

The challenges of part-of-speech (POS) tagging consist in ambiguous word-tag mappings and out-of-vocabulary cases. For *Balloon* a Markov tagger has been developed which has already been introduced in detail in [6, 7]. Similar to the *TnT* tagger [8] it additionally uses POS information stored in word suffixes. Whereas in the *TnT* approach these suffixes are simply defined by word-final substrings of certain lengths, the *Balloon* tagger tries to extract meaningful suffix units by means of an adaptation of the peak and plateau algorithm of [9].

3.3. Morphological analysis

The morphological analysis yields a flat segmentation of a word into morphemes and their morpheme classes. The concatenative approach underlying the analysis has

been introduced in [10] for German and in [5] for English. Generally, each input s is recursively divided into substring prefixes and suffixes from left to right. In the course of the recursion, a boundary dividing the current substring into the prefix a and the suffix b is accepted if (1) a is found in the lexicon, (2) there exists a permitted segmentation for b into $b_1 \dots b_n$, or (if not) b is found in the lexicon, (3) the sequence *class(a), class(b_1)* is not in conflict with language-dependent morphotactics, and 4) *class(b_n)* is compatible with the POS of the word.

3.4. Grapheme-to-phoneme conversion

G2P conversion is carried out by a C4.5 decision tree [11] as described in [12] for German and [13] for English. It is preceeded by a grapheme syllable segmentation which itself is also accomplished by a C4.5 tree [12]. The features used to map grapheme g_i onto phoneme p_i are: (1) the grapheme window $g_{i-2} \dots g_{i+2}$, (2) the position of g_i within the grapheme syllable to account for phenomena like the German terminal devoicing, (3) the class of the morpheme containing the grapheme, and recurrently (4) the left-adjacent G2P result. To cope with arbitrary *m-to-n*-mappings, graphemes are not only converted to single phonemes but also to phoneme clusters and empty phonemes arising from the alignment.

For English phonological syllabification is again carried out by a C4.5 tree [5]. For German syllable boundaries are placed in front of each sonority minimum, and their locations are subsequently adjusted in case German syllable phonotactics as specified by [14] is violated [12].

3.5. Word stress assignment

Balloon assigns word stress again by means of a C4.5 tree that predicts for each syllable whether it is stressed or not. The feature vector contains the word class, the syllable weight, the compound part index and the class of the morpheme containing the nucleus of the syllable for which the word stress decision is to be made.

The shortcomings of this syllable-based approach are, (1) that the stress assignment decision is made locally in isolation for each syllable not accounting for the overall word pattern, and (2) that it does not serve to locate the main stress in compounds which is highly relevant for the very productive compounding mechanism in German.

For this reason an alternative stress assignment algorithm has been developed which incorporates principles of metrical phonology [15] to locate main stress within compounds and of instance-based learning to account for global word patterns influencing stress assignment. The algorithm consists of three steps: (1) compound decomposition of the morphological segmentation, (2) metrical tree induction to locate the stressed compound part, and (3) instance-based classification to locate the stressed syllable within this compound part.

3.5.1. Compound decomposition

Compound decomposition operates on the output of the flat morphological segmentation introduced in section 3.3. It simply places compound boundaries between adjacent morphemes m_i and m_{i+1} if the following set memberships hold: $m_i \in \{$lexical morph, inflection ending, suffix, ordinal marker, gap morpheme$\}$, $m_{i+1} \in \{$lexical morph, prefix, adverb, verbal particle$\}$.

For the compound *Wasserstandsanzeiger (water level tracer)* which can be morphologically split into *wasser+stand+s+an+zeig+er* the compound decomposition places a boundary between the gap morpheme and the verb particle thus yielding [*wasser+stand+s*]+[*an+zeig+er*].

3.5.2. Metrical tree induction

To locate word stress within compounds [15] provides the recursive *compound stress rule (CSR)* saying: given constituent [AB], B is strong s if only B is further divisible, else A. The stressed compound part is finally identified by following the s branches from the tree root to the leafs.

Coherence Trees In the current approach, metrical trees are inferred from *coherence trees* reflecting the coherence values of adjacent compound parts x, y. Following [16] we measure coherence in terms of the likelihood ratio $\frac{H0}{H1}$ of the two hypotheses H0: *x and y are independent* and H1: *x and y are mutually dependent*. The likelihood ratio is derived from the likelihoods L_{H0} and L_{H1} of the observed co-occurrence frequencies of x and y. The ratio is then transformed to a χ^2 value by $\chi^2 = -2ln\frac{L_{H0}}{L_{H1}}$ resulting in a gradual interpretation of coherence: the higher χ^2, the higher the coherence between x and y. A coherence tree is then constructed recursively, branching at local coherence minima as is illustrated for the compound [*braun*][*kohle*][*berg*][*bau*][*skandal*][*nudel*] *(brown coal mining sleazebag)* in Figure 4.

From the resulting coherence tree a metrical tree is derived by adding s (strong) and w (weak) labels to the branches according to the CSR as shown in Figure 4.

3.5.3. Instance-based learning

To locate the stressed syllable within the compound part x at the end of the s-path in the metrical tree, it is first checked whether x contains one or more stressable syllables. Only one stressable syllable is given for example in the presence of stress-attracting affixes as in German *Reg*+**ent** *(regent)*, or if after affix stripping a one-syllable word stem remains as in *Ver*+**bann**+*ung (banishment)*.

Within multi-syllable word stems stress is located by k-nearest-neighbor-classification. The stem in question is compared with a database of word stems stored together with their stress position relative to the final syllable. The comparison is carried out with respect to six features that are vowel quantity (long, short, reduced) and

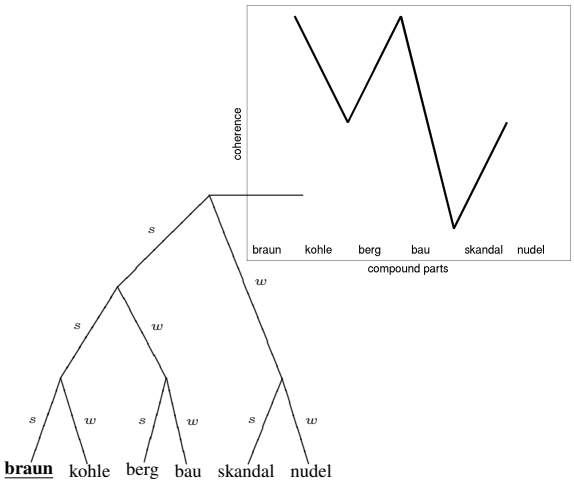

Figure 4: Metrical tree inference: recursive branching at local compound part coherence minima.

syllable coda type (open, closed) of the last 3 syllables. The distance D between the feature vectors a and b is measured by means of the weighted Hamming distance: $D(a,b) = \sum_{a_i \neq b_i} w_i$, i indexing the features. Here the weight w of a feature X is defined as the *mutual information* $MI(X;Y)$ between X and the word stress position Y. $MI(X;Y)$ is based on the entropy H of Y and its conditional entropy given that the value of feature X is known: $MI(X;Y) = H(Y) - H(Y|X)$. It turned out that vowel quantity, especially in the ult and penult syllable contains more information about the word stress position than syllable coda type.

3.5.4. Evaluation

For a sample of 700 compounds containing more than two parts the accuracy of the hierarchical compound analysis based on the morphological segmentation and the coherence tree induction amounted 83%. Compound stress assignment was successful in 95% of all cases. The adequacy of the compound stress rule expressed in the conditional probability $P(\text{stress correct} \mid \text{compound analysis correct})$ is 0.96, indicating that for the used data this rule is appropriate.

In a 10-fold cross validation task on 1300 multi-syllable word stems the k-nearest-neighbor-classification for k=15 successfully predicted the stress location in 84% of all cases. This value is to be seen as a lower performance bound, since one-syllable simplex and stressed affix cases have not been included in the test set and are trivially classified correctly.

4. Conclusion

In this paper the underlying algorithms of an alignment and a text processing online tool have been addressed. These tools can currently be found following this link:

http://www.phonetik.uni-muenchen.de/∼reichelu/webservices.html.

As regards webservice chaining *Balloon* is already integrated in the MAUS webservice for phonetic segmentation [17] currently to be found here:

https://webapp.phonetik.uni-muenchen.de/BASWebServices.

5. Acknowledgments

The work of the author has been carried out within the CLARIN-D project [1] (BMBF-funded).

6. References

[1] "http://eu.clarin-d.de/index.php/en/," Clarin-D web page.

[2] U. Reichel and R. Winkelmann, "Phoneme-to-phoneme alignment and conversion," in *Elektronische Sprachverarbeitung 2010*, ser. Studientexte zur Sprachkommunikation, R. Hoffmann, Ed. Dresden: TUDpress, 2010, pp. 126–133.

[3] R. Wagner and M. Fischer, "The string to string correction problem," *Journal of the Association for Computing Machinery*, vol. 21, no. 1, 1974.

[4] G. Kondrak, "Algorithms for Language Reconstruction," Ph.D. dissertation, University of Toronto, 2002.

[5] U. Reichel and H. Pfitzinger, "Text Preprocessing for Speech Synthesis," in *Proc. TC-Star Speech to Speech Translation Workshop*, Barcelona, Spain, 2006, pp. 207–212.

[6] U. Reichel, "Improving Data Driven Part-of-Speech Tagging by Morphologic Knowledge Induction," in *Proc. AST Workshop*, Maribor, 2005.

[7] U. Reichel and L. Bucar Shigemori, "Automatic correction of part-of-speech corpora," *Speech and Language Technology*, vol. 11, pp. 167–174, 2008.

[8] T. Brants, "TnT – a statistical part-of-speech tagger," in *Proc. ANLP-2000*, Seattle, WA, 2000, pp. 224–231.

[9] M. Nascimento and A. da Cunha, "An experiment stemming non-traditional text," in *Proc. SPIRE'98*, Santa Cruz de La Sierra, Bolivia, 1998, pp. 74–80.

[10] U. Reichel and K. Weilhammer, "Automated Morphological Segmentation and Evaluation," in *Proc. 4th Language Resources & Evaluation Conference*, Lisbon, Portugal, 2004, pp. 503–506.

[11] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Mateo: Morgan Kaufmann, 1993.

[12] U. Reichel and F. Schiel, "Using Morphology and Phoneme History to improve Grapheme-to-Phoneme Conversion," in *Proc. Eurospeech*, Lisboa, 2005, pp. 1937–1940.

[13] U. Reichel, H. Pfitzinger, and H. Hain, "English grapheme-to-phoneme conversion and evaluation," *Speech and Language Technology*, vol. 11, pp. 159–166, 2008.

[14] K. Kohler, *Einführung in die Phonetik des Deutschen*. Berlin: Erich Schmidt Verlag, 1995.

[15] M. Liberman and A. Prince, "On Stress and Linguistic Rhythm," *Linguistic Inquiry*, vol. 8, pp. 249–336, 1977.

[16] T. Dunning, "Accurate methods for the statistics of surprise and coincidence," *Computational Linguistics*, vol. 19, pp. 61–74, 1993.

[17] F. Schiel, "Automatic Phonetic Transcription of Non-Prompted Speech," in *Proc. ICPhS*, San Francisco, 1999, pp. 607–610.