Information Retrieval of Word Form Variants in Spoken Language Corpora Using Generalized Edit Distance

Siim Orasmaa[1], Reina Käärik[1], Jaak Vilo[1], Tiit Hennoste[2]
[1]Institute of Computer Science, University of Tartu, Estonia
[2]Institute of Estonian and General Linguistics, University of Tartu, Estonia
J. Liivi Str 2, 50409 Tartu, Estonia
E-mail: {siim.orasmaa; reinak; jaak.vilo; tiit.hennoste}@ut.ee

Abstract

An important feature of spoken language corpora is existence of different spelling variants of words in transcription. So there is an important problem for linguist who works with large spoken corpora: how to find all variants of the word without annotating them manually? Our work describes a search engine that enables finding different spelling variants (true positives) from corpus of spoken language, and reduces efficiently the amount of false positives returned during the search. Our search engine uses a generalized variant of the edit distance algorithm that allows defining text-specific string to string transformations in addition to the default edit operations defined in edit distance. We have extended our algorithm with capability to block transformations in specific substrings of search words. User can mark certain regions (blocked regions) of the search word where edit operations are not allowed. Our material comes from the Corpus of Spoken Estonian of the University of Tartu which consists of about 2000 dialogues and texts, about 1.4 million running text units in total.

1. Problem

An important feature of spoken language is existence of several different pronunciation variants of words. The variants are transcribed with different spelling variants in the corpus. So there is an important problem for linguist who works with large spoken corpora: how to find all variants of the word from transcribed corpus?

Many search engines will return only exact matches for the input search word. From linguist's point of view, such engine is not feasible, because it would demand from user to predict all the possible spelling variants and enter them one by one.

Alternatively, one could use approximate search algorithms such as Levenshtein distance to allow variations from the given input word. However, when using the Levenshtein distance, one must frequently browse through large amount of false positive results in order to find the few relevant variants.

We have developed a search engine which solves the problem using a generalized variant of the edit distance algorithm that allows defining text-specific string to string transformations in addition to the default edit operations defined in edit distance. We have extended the algorithm with blocking capabilities: user can mark certain regions of the search word where edit operations are not allowed.

A technique similar to our generalized edit distance has been used by Lindén (Lindén, 2006) for finding cross-lingual spelling variants of technical terms. A concept of blocking changes in certain regions of the search string has been used for example in Unix tool *agrep* (Wu & Manber, 1992).

However, to our best knowledge, there has not been made a tool that combines these two techniques, and there is no tool for searching word variants of spoken language where variability is very complicated and frequently unpredictable.

2. Corpus of Spoken Estonian of the University of Tartu

Our material comes from the Corpus of Spoken Estonian of the University of Tartu (Hennoste et al, 2008). The corpus includes mainly audio recordings and consists of about 2000 dialogues and texts which makes about 1.4 million running text units (words, fillers, pauses) in total. The corpus is divided by five dimensions that influence the language use:

- social and dialectical background of interactants;
- dialogue *vs* monologue;
- the degree of spontaneity of speech;
- the closeness of contact between participants (face-to-face, telephone, mass-media);
- the degree of institutionality of communicative situation (private/institutional).

The corpus is divided as follows at the moment:

- telephone conversations (63% of texts): private and institutional calls (directory inquiries, travel agency information requests, services etc.), telesales conversations, shopping information, taxi calls etc;
- face-to-face conversations (29%): everyday and institutional dialogues or monologues: shop dialogues, service dialogues (post office, library, shoemaker, etc.), conversations between strangers on the streets, doctor-patient encounters, interviews, travel agency dialogues, classroom interactions, meetings, conference presentations, lectures etc;
- media broadcasts (8%).

The transcription system of the conversational analysis (CA) is used which means that the categories crucial from the interactional point of view are important in the transcription (Hennoste, 2000b).

The words are transcribed in accordance with pronunciation but the characters of standard Estonian orthography are used (e.g. *sis* (standard spelling *siis*

'then'), *halloo* (*hallo* 'hello'), *onju* (*on ju* 'isn't it'), *kaeksada* (*kaheksasada* 'eight hundred') There are special spelling rules for dialogue particles which are used in written texts only in special cases (e.g. *ahah* (English 'oh'), *ee, õõ* (pause fillers).

Estonian has a complicated system of quantity patterns. What is important here is the fact that there are three stresses or 'quantity degrees' in Estonian. Short vowels and consonants are transcribed with one vowel or consonant, long and overlong ones with two vowels or consonants here. The exception is transcription of stops. Short stops are written with *g, b, d*, long ones with *k, p, t*, and overlong stops with *kk, pp, tt*.

3. Spelling Variants in the Corpus of Spoken Estonian

In the corpus we can find at least 5% of running words which have different types of variants dividable into four groups (Hennoste, 2000a).

1. Missing of some sound or syllable in different positions of the word (*mõtlesin* > *mõtsin* 'I thought'). Special case here is missing of laryngeal *h* at the beginning of the first or second syllable of the word (*kaheksa* > *kaeksa* 'eight', *helistama* > *elistama* 'to call').
2. Changing diphthong with single vowel (*päev* > *pääv* 'day') or consonant cluster with long or short single consonant (*seljast* > *sellast* 'from back').
3. Changing a vowel with some acoustically higher or lower vowel (*üheksa* > *öheksa* 'nine').
4. Replacing of a (over)long vowel or consonant with short one (*kuule* > *kule* 'hear!') or a shorter consonant with longer one (*allkirja* > *alkkirja* 'signature').

As variation takes place in different vowels and consonants, the number of different variants in corpus is very high and adding new words to the corpus can bring new variants in.

4. The Generalized Edit Distance

There are two criteria for the search engine: it must find automatically all variants of an input word (true positives) and it must give as little as possible wrong words (false positives).

The regular edit distance (Levenshtein distance) satisfies the first criterion as all the relevant word variants can be reached with combining the three edit operations (replacing, deleting or adding an arbitrary letter). However, arbitrary transformations often produce a large number of results and time-consuming work is still required to find the few relevant variants amongst them.

The generalized edit distance allows addressing problems with the second criterion. It allows user to define, which differences or variations from given search string are important and frequent and hence more permissible in the matching. When defining these variations (string to string transformations), we can use our knowledge about known variation of sounds, which was previously acquired through the analysis of text and conversation corpora.

The permissible transformations can be composed manually or even generated automatically in some cases. In manual design we can use our prior knowledge about types and frequency of different spelling variants. The weights of transformations can be assigned by search engine designer, for example using some default values. In the case of transparent and open user interface, users can be allowed to change the predefined transformations or their weights, making some experimental queries on the corpus

Consonants		**Vowels**
t > d (0.02)	j > jj (0.04)	oo > o (0.02)
d > t (0.02)	jj > j (0.04)	aa > a (0.02)
t > tt (0.04)	l > ll (0.02)	ee > e (0.02)
tt > t (0.04)	ll > l (0.02)	ii > i (0.02)
b > p (0.02)	m > mm (0.02)	uu > u (0.02)
p > b (0.02)	mm > m (0.04)	üü > ü (0.02)
p > pp (0.02)	h > hh (0.02)	õõ > õ (0.02)
pp > p (0.04)	h > (0.10)	öö > ö (0.02)
k > g (0.02)	nn > n (0.04)	
g > k (0.04)	n > nn (0.04)	
k > kk (0.02)	rr > r (0.04)	
kk > k (0.02)	r > rr (0.04)	
s > ss (0.02)	v > vv (0.04)	
ss > s (0.04)	vv > v (0.04)	

Table 1: List of 37 transformations currently used in the search engine.

In current implementation we have manually compiled a set of transformations and their weights using the following guidelines.

1. By default, the operation of replacing, deleting or adding an arbitrary letter in the search string has predefined cost of 1.0 (as it is normally defined in the edit distance). If we want that our transformation gets used during the search, its weight must be lower than the sum of weights of the default edit operations, applied to perform the same string to string transformation. If generalized edit distance between the search string and a match is greater than 1.0, it should indicate that at least one of the regular edit operations was used during the search (so, the match contains at least one error).
2. The transformations are applied in parallel, that is to say, when a substring has already been changed by a transformation, it cannot be changed once again. For example, the transformation aa→a cannot be recursively applied as a substitute for the transformation aaaa→a.

Our tests on the Corpus of Spoken Estonian have shown that the best results are obtained if we only use transformations of long and overlong sounds to short ones and shorter consonants to longer ones. The only exception is transformations of h. Overall we have currently defined 37 transformations, which have predefined weights in range 0.02 to 0.1.

5. Example of Using Generalized Edit Distance

Consider finding all the variants of the word *läheb* ('he/she goes') from the corpus (Table 2). There are 6 variants of the verb currently in corpus: *läheb, lähep, läeb, lähäp, lääb, läb.* When we are using the Levenshtein distance-based search, we must allow at least 2 edit operations to get all 6 variants, which gives us overall 62 words as a result. When using the generalized edit distance with transformations given Table 1, we can lower the maximum distance to 1.5, which gives us 18 words as a result, including all the relevant variants of the verb. Our example shows that using generalized edit distance gives those relevant answers with fewer false positives than if using standard edit distance (12 vs 56 false positives respectively).

Distance limitation	**Edit distance**	**Generalized edit distance**
$d < 1.0$	**0.00 läheb** (Relevant: 1/1)	**0.00 läheb** **0.02 lähep** **0.10 läeb** (Relevant: 3/3)
$d <= 1.0$	1.00 lähe '(not) going' 1.00 lähen 'I am going' 1.00 lähed 'you are going' **1.00 läeb** **1.00 lähep** **1.00 lähäb** 1.00 täheb 'means' 1.00 Lähe '(not) going' 1.00 lähem 'nearer' (Relevant: 4/10)	1.00 lähe '(not) going' 1.00 lähen 'I am going' 1.00 lähed 'you are going' **1.00 lähäb** 1.00 täheb 'means' 1.00 Lähe '(not) going' 1.00 lähem 'nearer' (Relevant: 4/10)
$d <= 1.5$		1.10 loeb 'he/she reads' **1.10 läb** 1.10 näeb 'he/she sees' **1.10 lääb** 1.10 läen 'I am going' 1.10 läe '(not) going' 1.10 läet 'you are going' 1.12 läp 'laptop' (Relevant: 6/18)
$d <= 2.0$	2.00 lähme 'we are going' 2.00 vähe 'less' 2.00 lähete 'you are going' 2.00 loeb 'you are reading' 2.00 vähem 'lesser' **2.00 läb** 2.00 pähe 'to head' 2.00 läheks 'would go' 2.00 näeb 'is seeing' 2.00 lõpeb 'is ending'	

	2.00 lahe ‘cool’ 2.00 lehe ‘newspaper's’ 2.00 lehes ‘in newspaper’ 2.00 ähib ‘is puffing’ 2.00 lehed ‘leaves’ 2.00 laseb ‘is allowing’ 2.00 tähed ‘letters’ 2.00 lähebki ‘is going’ 2.00 läheme ‘we are going’ 2.00 läme ‘we are going’ 2.00 lühem ‘shorter’ 2.00 lähegi ‘(is not) going’ **2.00 lääb** 2.00 läen ‘I am going’ 2.00 läh ‘go’ beginning 2.00 lähema ‘of nearest’ 2.00 lähim ‘nearest’ 2.00 lähte ‘?’ 2.00 mähet ‘diaper’ 2.00 lähäd ‘you are going’ 2.00 lõhub ‘is breaking’ 2.00 nähe ‘phenomenon’ 2.00 tähe ‘meaning’ beginning 2.00 vähemb ‘less’ 2.00 lahee ‘?’ 2.00 lahel ‘at the gulf’ 2.00 lehel ‘on the paper’ 2.00 lohe ‘dragon’ 2.00 lohed ‘dragons’ 2.00 läbib ‘is going through’ 2.00 läe ‘(not) going’ 2.00 läet ‘you are going’ 2.00 lähegu ‘let (him/her) go’ 2.00 läheneb ‘is approaching’ 2.00 lähm ‘we are going’ beginning 2.00 lähtu ‘originating’ 2.00 lähva ‘they are going’ 2.00 lähä ‘is (not) going’ 2.00 läte ‘origin’ 2.00 lõhe ‘salmon’ 2.00 mähe ‘diaper’ 2.00 nähes ‘when seeing’ 2.00 tähen ‘meaning’ beginning (Relevant: 6/62)	

Table 2: Comparison of regular edit distance and generalized edit distance on searching variants of the word *läheb*. The relevant results are marked with bold. The decimals preceding words in table show the distance between a word and the search string.

6. Blocked Regions in Search Word

When the maximum edit distance is increased, the number of results that needs to be examined also typically increases. Allowing the regular edit operations to occur at arbitrary locations in search string introduces many transformations that are not probable variations in real language (e.g. in Table 2 only 6 of 62 word variants returned by edit distance were relevant).

In order to address this problem, we extend our algorithm with capability to block transformations in specific substrings of search string/word (we call these substrings *blocked regions*).

Decisions about which substrings to block must be based on previous knowledge in which positions of the word the sounds do not vary and in which positions they vary in a certain predictable mode.

The analysis of our corpus has shown that there are two positions where the variation is well predictable and the transformations could be blocked.

Analysis has shown that there is no variation in the consonants at the beginning of word (except h which pronunciation varies very much). So it is possible to block the first consonant transformations.

The second possible position for blocking is the end of the word. There are three different possibilities.

First, there are some hardly predictable end transformations. The last sounds of the particles could disappear or could be changed in different ways (e.g. *millal>milla* 'when', *kui>ku* 'if, how') and the sounds at the end of the nominative case of the nouns could disappear (*päev>päe* 'day').

Second, there are quite well predictable variations. The stops at the end of the word could have short and long variants (b/p, d/t, g/k) (*läheb/lähep* 'goes'; *tändab/tändap* 'it means, it is') and the ending of present participle have four different variants: *nud/nd/d/n* (*teinud/teind/tein/teid* 'have done').

Third, there are endings where variability does not exist, at least in our corpus. Estonian is a language where the syntactical information is concentrated at the end of the word (person, mood, negation of the verb, the case endings of the nouns which show the role and meaning of the word in the clause. E.g. *mina istun, sina istud, tema istub* 'I am sitting, you are sitting, he/she is sitting', *toit:toidu:toidus* 'food NOMINATIVE: food GENITIVE: food INESSIVE 'in food'.

So it is clear that using blocked regions requires some previous knowledge about which parts of a search word are least likely to vary.

7. Types of Blocked Regions

Currently, we are allowing two kinds of blocked regions:

a) regions that block the regular edit distance only;

b) regions that block both the regular edit distance and the generalized edit distance.

Special metacharacters are used to mark down the regions where transformations are not permissible.

The region of type a) is defined by surrounding a substring of a search string with parentheses. For example, if the search string *läheb* is written as *(l)ähe(b)*, the regular edit operations are not allowed on the first and on the last letter of the string. The effect is that neither of the letters could be deleted or replaced by some arbitrary letter during the search. However, the transformations of the generalized edit distance (*l* > *ll* and *b* > *p*) and arbitrary additions are still permitted.

Arbitrary additions of the regular edit distance are blocked in two ways. Firstly, no arbitrary addition is allowed between the two blocked letters, e.g. *(l)(ä)heb*[1] and *(lä)heb* both successfully block random addition between *l* and *ä*. Secondly, blocked regions at the beginning (or at the end) of search string can be extended so that no addition is permitted before the first (or after the last) letter of the string. Such extensions are marked by double parentheses, for example, *((lä)heb* blocks random additions to the beginning of the string and *lähe(b))* disables random additions after the last letter *b*.

The region of type b) is surrounded by characters < and >. This type of blocking does not allow any modification on the blocked region, so if the search word *läheb* is written as *<lä>heb*, all the results contain substring *lä*.

8. Example of Using Blocked Regions

With blocked regions, we can further narrow down the number of results returned by both the regular edit distance and the generalized edit distance. Consider finding variants of the word *läheb* when the regular edit distance changes on the first and on the last letter of the string are blocked.

Table 3 compares edit distance and generalized edit distance in search with the blocked regions. The number of overall search results with edit distance is reduced from 62 (see Table 2) to 12. However, one of the relevant variants (*lähep*) cannot be reached with given query, because the variant contains a change in blocked regions (*b* > *p*). In order to get all the relevant results with regular edit distance, we must remove the blocked region from the end of the search string, which gives us 48 results at distance d <= 2.0 (not in Table 3).

With generalized edit distance, the transformation *b* > *p* is still allowed in the last blocked region, so no loss occurs and number of variants is reduced to 8. All removed variants are false positives.

9. The Implementation

Our approximate search tool runs currently only on LINUX platform.

Our implementation of the search engine has a web-based user interface, which allows user to change transformations and their weights, thus allowing the experimental approach on composing the transformations.

The time complexity of a search engine using generalized edit distance depends on the number and size of used transformations. The efficiency of the search is achieved by using Aho-Corasick multiple pattern matching algorithm (Aho & Corasick, 1975) applied to generalized edit distance calculations (Käärik, 2006).

In addition to above described full matching, the partial matching can be performed; it means that we can match the search string with the prefix, suffix or infix of a word.

In addition, we have implemented the support for recursive queries, which allows selecting a sub-corpus (defined by words selected from results of previous queries), entering a new search word and performing a search upon this sub-corpus, in order to find co-occurrences of the words.

[1] In some cases one might also want to block changes on two consecutive letters, but allow insertions between them, like the marking *(l)(ä)heb* suggests. This is currently not supported by our engine; however we plan to implement this in the future.

Distance limitation	Using edit distance with search word (*l*)*ähe(b)*	Using generalized edit distance with search word (*l*)*ähe(b)*
d < 1.0	**0.00 läheb** (Relevant: 1/1)	**0.00 läheb** **0.02 lähep** **0.10 läeb** (Relevant: 3/3)
d <= 1.0	**1.00 läeb** **1.00 lähäb** (Relevant: 3/3)	**1.00 lähäb** (Relevant: 4/4)
d <= 1.5		1.10 loeb 'he/she reads' **1.10 läb** **1.10 lääb** 1.12 läp 'laptop' (Relevant: 6/8)
d <= 2.0	2.00 loeb 'you are reading **2.00 läb** 2.00 lõpeb 'is ending' 2.00 laseb 'is allowing' 2.00 lähebki 'is going' **2.00 lääb** 2.00 lõhub 'is breaking' 2.00 läbib 'is going through' 2.00 läheneb 'is approaching' (Relevant: 5/12)	

Table 3: Comparison of edit distance and generalized edit distance when using (*l*)*ähe(b*) as search string. Parentheses in search string mark the blocked regions where edit distance operations are not allowed. The relevant results are marked with bold. The decimals preceding words in table show the distance between a word and the search string.

10. Conclusion

In this work, we have introduced a search engine which allows retrieval of word variants from a spoken language corpus, and reduces efficiently the amount of false positives returned during the search. We have developed a generalized variant of the edit distance algorithm that allows defining text-specific string to string transformations in addition to the default edit operations defined in edit distance. The algorithm is also extended with blocking capabilities: user can mark certain regions of the search word where edit operations are not allowed.

Our tests on the Corpus of Spoken Estonian of the University of Tartu have shown that the best results are obtained if we only use transformations of long and overlong sounds to short ones and shorter consonants to longer ones. The only exception is transformations of *h*.

Blocking applied at the first and at the last letter of a search word has also shown promising results, however, this approach still requires further research.

11. Perspective and Future Work

Our future research will focus on extending the set of generalized edit distance transformations, for example, allowing transformations of raising or lowering of vowels. The method of combining the generalized edit distance with blocked regions also needs further analysis in order to find more general cases when blocking can be safely applied.

Our search engine can be adapted to the other languages and to different varieties of a language (e.g. old standards which have typically different spelling, old rural dialects etc). In addition, it can be used to perform searches from dictionaries by presumed pronunciation rather than exact spelling. In that case the transcription of the pronunciation or pronunciation-spelling information can be used. For other alphabets, e.g. Cyrillic, this can be of great help allowing efficient searches using flexible spellings in Latin alphabets, for example.

We have also experimented with automatic methods for generating text transformations and weights with a rather good success.

Acknowledgements

This research was supported by the European Regional Development Fund through the Estonian Center of Excellence in Computer Science, EXCS; Estonian National Program for Language Technology grants EKKTT09-61 and EKKTT06-12; Estonian Science Foundation grant no 7503; European Social Fund's Doctoral Studies and Internationalisation Programme DoRa.

References

Aho, A., Corasick M.J. (1975). Efficient string matching: An aid to bibliographic search. *Communications of the ACM*, 18 (6), pp. 333–340. doi:10.1145/360825.360855.

Hennoste, T. (2000a). Introduction to Spoken Estonian III. *Akadeemia*, 7, pp.1553--1582. (in Estonian)

Hennoste, T. (2000b). Studying Spoken Estonian: transcription, background, and corpus. *Keel ja Kirjandus*, 2, pp. 91--106. (in Estonian)

Hennoste, T., Gerassimenko, O., Kasterpalu, R., Koit, M., Rääbis, A., Strandson, K. (2008). From Human Communication to Intelligent User Interfaces: Corpora of Spoken Estonian. *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, 2008, European Language Resources Association (ELRA), Marrakech, Morocco, May, 28-30. http://www.lrec-conf.org/proceedings/lrec2008/

Käärik, R. (2006). *Generalized edit distance*. BSc thesis. University of Tartu. (in Estonian with English summary) http://www.egeeninc.com/u/vilo/edu/Students/Reina_Kaarik/Bakalaureuset88.pdf

Lindén, K. (2006). Multilingual modeling of cross-lingual spelling variants, *Information Retrieval*, 9(3), pp. 295--310.

Wu, S., Manber, U. (1992). Fast Text Searching Allowing Errors. *Communications of the ACM*, 35 (10), pp. 83--91.