Dialect Pronunciation Comparison and Spoken Word Recognition

Martijn Wieling
Alfa-Informatica
University of Groningen
wieling@gmail.com

John Nerbonne
Alfa-Informatica
University of Groningen
j.nerbonne@rug.nl

Abstract

Two adaptations of the regular Levenshtein distance algorithm are proposed based on psycholinguistic work on spoken word recognition. The first adaptation is inspired by the Cohort model which assumes that the word-initial part is more important for word recognition than the word-final part. The second adaptation is based on the notion that stressed syllables contain more information and are more important for word recognition than unstressed syllables. The adapted algorithms are evaluated on a large contemporary collection of Dutch dialect material, the Goeman-Taeldeman-Van Reenen-Project (GTRP, collected 1980–1995) and a relatively small Norwegian dataset for which dialect speakers judgments of proximity is available.

Keywords

Dialect, Levenshtein algorithm, Spoken Word Recognition, Cohort theory, Dutch, Goeman-Taeldeman-Van Reenen-Project.

1 Introduction

The Levenshtein distance algorithm is a popular sequence-based method used to measure the perceptual distances between dialects (See Heeringa [12] for detailed descriptions of the Levenshtein algorithm as applied to dialectology as well comparison to alternative approaches). In the Levenshtein algorithm every edit operation is assigned a certain cost (in our case all operations have the same cost, 1). The location of the edit operations is not relevant in determining the cost; a substitution at the first position of both strings has the same cost as a substitution at the final position of both strings. While this is a sensible notion, there are some theories of spoken word recognition which suggest another approach.

Although there is research on perceived similarity which emphasizes that vowels and codas are more important than onsets [11, 22, 27], which would suggest that we weigh *later* segments more heavily than initial ones, we focus on the (opposing) conjecture that initial elements are more important. Our results in general, but also some specific experiments make it unlikely that later elements play a proportionately greater role, as we shall see (Section 5, Discussion).

Although it is natural to examine psycholinguistic theories of word recognition as a source of ideas about which parts of words might be most important to dialect perception, we should also be aware that word recognition and dialect perception are different. The task of spoken word recognition is to determine which word was said while the purpose of dialect variation is to signal the speaker's provenance. Thus aspects of the speech signal that support word recognition may not support inferences about the speaker's (geographic) identity. This is related to the semiotic division between the relation of signs to denotations (or meanings) on the one hand and the relation of signs to senders or interpreters on the other [3]. From the point of view of communication (or word recognition), dialect variation only adds noise to a signal. So we shall not pretend to criticize theories of word recognition, even in case it turns out that they contribute little to dialect perception. But it is equally plausible that the mechanisms that make some parts of the speech signal more important for recognition and perception would also be important dialectologically.

The Cohort model was the first very influential theory on spoken word recognition. The Cohort theory [19, 20] proposes that word recognition occurs by activating words in memory based on the sequential (left-to-right) processing of the input sound. The first phoneme of a word activates all words which start with that sound, the *word-initial cohort*. Additional phonemes narrow the cohort by ruling out members which do not match the heard sound sequence. For instance after hearing the first phoneme of the word 'elephant', the words 'elephant', 'elevator' and 'enemy' are activated. After hearing the second phoneme the cohort is reduced to the words 'elephant' and 'elevator'. Subsequent phonemes will reduce the number of items in the cohort until only the word 'elephant' remains and is recognized. Hence, the start of the word is more important than later parts of the word [21]. In the context of developing a computational model for detecting confusable drug names, Kondrak and Dorr [15] report that 74.2% of confusable drug names in their data set have identical initial letters, as opposed to only 6.5% of randomly selected pairs. Even though the Cohort model has a number of drawbacks (e.g., correct recognition of a word is not possible when the start of a word is misheard) and other theories of word recognition have been proposed which do not rely on left-to-right activation [14, 17], the start of a word is nevertheless important in word recognition [4, 28].

There is also evidence for the importance of stressed syllables in word recognition. First, stressed syllables are more easily identified out of their original context than unstressed syllables [6]. And second, stressed syllables have been found to be more informative than unstressed syllables [1].

The Levenshtein algorithm can be easily extended to incorporate these theories. Cohort theory can be modelled by weighting differences in the beginning of both strings more heavily than differences at the end. The importance of stressed syllables can be modelled by weighting differences in stressed syllables more strongly than differences in unstressed syllables.

2 Material

In this study we use two different dialect data sources. The first data set consists of data from the most recent Dutch dialect data source, the Goeman-Taeldeman-Van Reenen-Project (GTRP) [9, 25]. The GTRP consists of digital transcriptions for 613 dialect varieties in the Netherlands (424 varieties) and Belgium (189 varieties), gathered during the period 1980–1995. The geographic distribution of these varieties is shown in Figure 1. For every variety, a maximum of 1876 items was narrowly transcribed according to the International Phonetic Alphabet. The items consist of separate words and word groups, including pronominals, adjectives and nouns. A more detailed overview of the data collection is given in [24].

Because the GTRP was compiled with a view to documenting both phonological and morphological variation [7] and our purpose here is the analysis of variation in pronunciation, many items of the GTRP are ignored. We use the same 562 item subset as used in [30] and introduced and discussed in depth in [29]. In short, the 1876 item word list was filtered by selecting only single word items, plural nouns (the singular form was preceded by an article and therefore not included), base forms of adjectives instead of comparative forms and the first-person plural verb instead of other forms. We omit words whose variation is primarily morphological as we wish to focus on pronunciation. In all varieties the same lexeme was used for a single item.

Because the GTRP transcriptions of Belgian varieties are fundamentally different from transcriptions of Netherlandic varieties [29], we will analyze the two data sets separately. Furthermore, note that we will not look at diacritics, but only at the segmental phonetic symbols (82 for the Netherlands and 50 for Belgium). The average length of items (without diacritics) in the GTRP is 4.7 tokens.

The second data set is a Norwegian dataset for which dialect speakers' judgments of proximity are available [13]. The Norwegian dataset consists of 15 places for which 58 different words of the fable 'The North Wind and the Sun' were phonetically transcribed. The average length of items in this dataset is 3.5 tokens. The perceptual distances were obtained by similarity judgments of groups of high school pupils from all 15 places; the pupils judged all dialects on a scale from 1 (most similar to native dialect) to 10 (least similar to native dialect). Note that these perceptual distances are not necessarily symmetrical; an inhabitant from region A may rate dialect B more different than an inhabitant from region B rates dialect A.

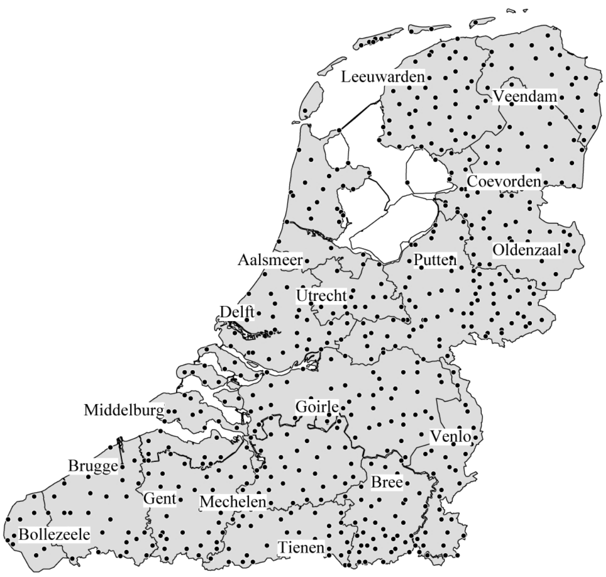

Fig. 1: *Distribution of GTRP localities*

3 Adapted Levenshtein distance algorithms

It is straightforward to adapt the regular Levenshtein algorithm to allow for custom weighting based on the positions `i` and `j` in both strings. The adapted algorithm shown in pseudocode below uses a cost function `CF(i,j)` to calculate the weight of an edit operation at positions `i` and `j` in both strings. Note that the regular Levenshtein distance can be calculated by setting `CF(i,j)` to 1 for every `i` and `j`.

```
LEVEN_TABLE(0,0) = 0

FOR i := 1 TO LENGTH(string1)
  LEVEN_TABLE(i,0) := LEVENTABLE(i-1, 0) + CF(i,0)
END

FOR j := 0 TO LENGTH(string2)
  LEVEN_TABLE(0,j) := LEVENTABLE(0, j-1) + CF(0,j)
END

FOR i := 1 TO LENGTH(string1) DO
  FOR j := 1 TO LENGTH(string2) DO
    LEVEN_TABLE(i,j) :=
      MIN(
           LEVEN_TABLE(i-1, j) + INS_COST * CF(i,j),
           LEVEN_TABLE(i, j-1) + DEL_COST * CF(i,j),
           IF finalchar1 = finalchar2 THEN
             LEVEN_TABLE(i-1, j-1) // no cost
           ELSE
             LEVEN_TABLE(i-1, j-1) + SUBST_COST * CF(i,j)
           END
         )
  END
END

RESULT :=

      LEVEN_TABLE( LENGTH(string1),LENGTH(string2) )
                           LENGTH(string2) )
```

We use a slightly adapted version of the Levenshtein algorithm displayed above. The modified Levenshtein algorithm enforces a linguistic syllabicity constraint: only vowels may match with vowels, and consonants with consonants. The specific details of this modification are described in more detail in [29].

3.1 Cohort inspired algorithms

In the Cohort model the importance of a sound segment is maximal at the onset of a word and decreases from there until the end. This can be modeled by setting the cost of an edit operation highest at the start of both strings, while gradually decreasing the cost when traversing to the end of both strings.[1]

We experimented with several weighting schemes to model the Cohort theory: a linear decay, an exponential decay, a square root decay and a (natural) logarithmic decay of the cost. The respective cost functions are specified in pseudocode below. Note that the optimal parameters for the exponential and linear decay functions were defined by experimentation.

```
// Exponential decay cost function
CF(i,j) := POW( 1.1, ( LENGTH(string1) - i +
                                   LENGTH(string2) - j ) )

// Linear decay cost function
CF(i,j) := 0.2 * ( LENGTH(string1) - i +
                                 LENGTH(string2) - j ) + 1

// Square root decay cost function
CF(i,j) := SQRT( LENGTH(string1) - i +
                               LENGTH(string2) - j + 1 )

// Logarithmic decay cost function
CF(i,j) := LOG( LENGTH(string1) - i +
                              LENGTH(string2) - j + EXP(1) )
```

Figure 2 visualizes the cost functions for two strings which have an added length of 10 tokens. For every method the cost of an edit operation is highest at the start (left side of the graph) and lowest at the end of the strings (right side of the graph). The final edit operation in every cost function always has cost 1. The cost of earlier operations depends on the position in both strings. For example, the cost of a different-token substitution of the second character in string A with the first character in string B can be found by looking at value 3 on the x-axis.

In the following we will refer to the Cohort-inspired Levenshtein algorithms as leven-cohort algorithms.

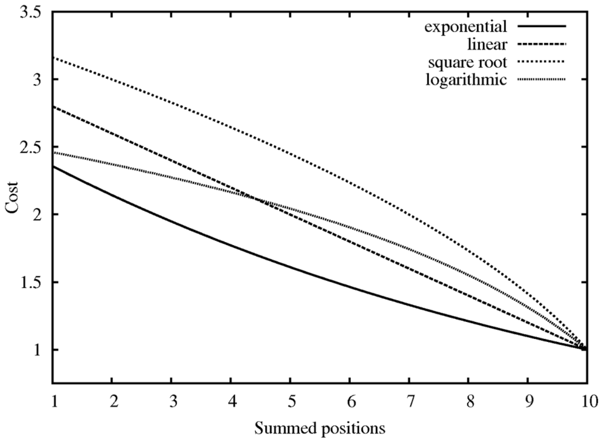

Fig. 2: *Cost functions for leven-cohort algorithms (see the text in Section 3.1 for a detailed explanation)*

3.2 Stress-based algorithm

To model the idea that stressed syllables are more important than unstressed syllables, the adapted Levenshtein algorithm ideally should assign a higher cost to edit operations which occur in stressed syllables rather than in unstressed syllables. Because it was not possible to identify the stressed syllable in every dataset and stress was placed predominantly on the first syllable in all datasets[2], the stress-based method was approximated by assigning a larger cost to edit operations occurring within the first three positions of both words.[3] The resulting cost function is shown in the pseudocode below.

```
// Stress-based cost function
HIGH_COST := 2
LOW_COST := 1

IF (i <= 3) AND (j <= 3) THEN
  CF(i,j) := HIGH_COST
ELSE
  CF(i,j) := LOW_COST
END
```

We will refer to the adapted Levenshtein distance algorithm as the leven-stress algorithm.

3.3 Length normalization

It is obvious that pairs of longer strings will on average have a larger Levenshtein distance than pairs of shorter strings. This bias is even greater for the leven-cohort algorithms because the average costs for edit operations are higher for longer strings than for shorter strings (i.e. an initial edit operation will have a higher cost for longer strings than shorter strings, while the final edit operation always has cost 1).

Because it is likely that dialect perception is word-based [12], it makes sense to normalize the Levenshtein distance so that it is length independent. For the regular Levenshtein distance there are several normalization methods. Heeringa et al. [13] pointed out that normalization by alignment length was the most natural procedure because then similarity and difference were each others' inverses (i.e. sum to 1). Other methods include normalizing by the length of the longest string, normalizing by the length of the shortest string and normalizing by the average string length [8].

Unfortunately these methods are not suitable to normalize the distances obtained with the leven-cohort (or leven-stress) algorithms. To see this, consider two different strings of length 1. The regular Levenshtein distance of these strings is exactly 1 (a single substitution). Because in this case both strings have the same length and the only edit-operation involved is the substitution, all normalization methods mentioned above yield the same normalized value (in this case 1). It is easy to see that for two strings of length 2 which do not have a character in common, the raw Levenshtein distance is 2 (two substitutions), while the normalized Levenshtein distance equals 1. Thus, in both cases

[1] Kondrak and Dorr [15] obtained good results in modeling name confusability using their PREFIX measure, which effectively favors the initial phonemes of the words. We learned of this too late too include it in comparisions.

[2] The Norwegian data at `http://www.ling.hf.ntnu.no/nos` appears to contain no non-initial stress; the GTRP datasets have < 2% non-initial stress.

[3] Using the first two positions of both words gave somewhat lower performance, but was still better than using the regular Levenshtein algorithm

the normalized Levenshtein distance is the same. This makes sense because in both situations the two strings are maximally different.

When considering the leven-cohort algorithm for two different strings of length 1, the raw and normalized adapted Levenshtein distance again equal 1 (because the final edit operation in every leven-cohort algorithm has cost 1). However when we increase the string length for both strings with one character, the raw distance will increase with a value larger than 1 (see Figure 2) and thus normalizing by string length will insufficiently counterbalance the positional weighting and yield a higher relative distance for longer strings. An example of this relative distance increase is shown in the table below. The first line of numbers shows the raw distances, while the bottom line shows the normalized distances for the linear leven-cohort algorithm.

r	r o	r o o	r o o d
g	g e	g e e	g e e l
1	2.4	4.2	6.4
1	**1.2**	**1.4**	**1.6**

Fortunately it is possible to construct a better normalization method which can be applied to the leven-stress and leven-cohort algorithms. In the following section we will adapt the method of normalization by alignment length while also preserving the desired feature that similarity and difference are each others' inverses.

Normalization by alignment cost

Instead of normalizing by alignment length, we normalize by the cost of the alignment. The cost of a specific alignment can be found by assuming that all aligned identical symbols are replaced by (different-symbol) substitutions. The distance of the new alignment (with the same length as the original alignment) is used for normalization. Note that this approach equals normalizing by alignment length when the costs of all edit operation equal 1, because in that situation the alignment length is equal to the cost of the alignment.

To make this approach clear, consider a possible alignment (including the costs) using the regular Levenshtein algorithm for two Dutch dialectal variants of the word 'milk', [mɔəlkə] and [mɛlək]:[4]

m	ɔ	ə	l		k	ə
m	ɛ		l	ə	k	
0	1	1	0	1	0	1

The total Levenshtein distance of these two words is 4. The cost of this alignment can be calculated by replacing all identical symbol pairs with different-symbol substitutions (additional costs are marked in boldface):

m	ɔ	ə	l		k	ə
m̂	ɛ		l̂	ə	k̂	
1	1	1	**1**	1	**1**	1

The cost of this alignment is 7 and so the normalized Levenshtein distance is $\frac{4}{7}$. The total similarity is equal to the additional costs (in boldface) introduced by replacing all identical symbol pairs with different-symbol substitutions, in this case 3. Because the normalized similarity is $\frac{3}{7}$, similarity and difference are each others' inverses. As pointed out earlier, these normalized values are equal to the values which are obtained by normalizing by alignment length.

To see that this normalization approach can also be used when position-dependent costs are used, consider the alignment (and corresponding edit operation costs) for the same strings using the linear leven-cohort algorithm.

m	ɔ	ə	l		k	ə
m	ɛ		l	ə	k	
0	2.4	2.2	0	1.6	0	1

The total leven-cohort distance of these two words is 7.2. The cost of this alignment equals 13 and is calculated as follows:

m	ɔ	ə	l		k	ə
m̂	ɛ		l̂	ə	k̂	
2.8	2.4	2.2	**1.8**	1.6	**1.2**	1

In this case the normalized leven-cohort distance equals $\frac{7.2}{13}$, while the normalized similarity equals $\frac{5.8}{13}$.

The normalization method introduced above will always yield normalized distance and similarity values in the range $[0, 1]$. Because the cost of an alignment is equal to the sum of the similarity and the distance of that alignment, the normalized values will always sum to 1 and thus are each others' inverses when normalized. For instance, two identical aligned strings of any length will have a normalized similarity of 1 (and distance of 0), while two completely different strings of any length will have a normalized distance of 1 (and similarity of 0).

In the previous examples, the cost of a substitution did not depend on the symbols involved. For instance, substituting an /a/ with an /e/ did not differ from substituting an /a/ with an /o/ (if the positions were the same). However, when substitution costs vary, it is not immediately clear which substitution cost should be used to calculate the similarity of two identical symbols (indicated in boldface in the examples above). In that case we suggest using the highest substitution cost involving that symbol.

Heeringa et al. [13] reported that the results using the raw Levenshtein distances were a better approximation of dialect differences as perceived by dialect speakers than results based on normalized Levenshtein distances. Because our normalization method is comparable to the method used in [13], we will examine if this is also the case in this study.

4 Results

First we assessed the reliability of the distance measurements using Cronbach's α. For the Norwegian distance measurements Cronbach's α ranged between

[4] The example involves a diphthong which might be analysed a single segment rather than as two. Heeringa [12], p.190, compares the two approaches and shows that there is little difference in overall quality of the dialect difference measure.

0.86 and 0.87, while it was equal to 0.99 for the Belgian and Netherlandic distance measurements. Because these values are much higher than the accepted threshold in social science (where $\alpha > 0.70$ is regarded as acceptable) we conclude that our distance measurements are highly consistent.

To evaluate the quality of the results, we used the local incoherence measurement which measures the degree to which geographically close varieties also represent linguistically similar varieties [23]. See Heeringa et al. [13] for a brief sketch of the local incoherence measure. In calculating the local incoherence values, the geographic distances for the Netherlands and Belgium were measured "as the crow flies", while we used travel time for Norway due to its rugged landscape [10]. Although the local incoherence cannot be used as a "gold standard", it can be used as an indicative heuristic for quality in dialectological measurements. Additionally, the quality of the Norwegian distances was assessed by correlating them with the perceptual distances.

We calculated the local incoherence values of the dialect distances obtained using the leven-stress and leven-cohort algorithms on the Norwegian data and the Netherlandic and Belgian GTRP data. Table 1 shows these values for both the normalized data (given between parentheses) as well as the unnormalized data. Because the local incoherence values are based on geographical distance (or travel time), the values in Table 1 can only be compared within a single column (i.e. data set), but not between the three separate columns.

	NL	BEL	NOR
exp.	1.93 (1.92)	0.76 (0.73)	0.44 (0.48)
lin.	1.92 (1.91)	0.76 (0.73)	0.44 (0.49)
sqrt.	1.91 (1.90)	0.76 (0.73)	0.43 (0.50)
log.	1.91 (1.90)	0.76 (0.73)	0.43 (0.49)
stress	1.89 (1.89)	0.75 (0.75)	0.45 (0.49)
regular	1.94 (1.94)	0.80 (0.79)	0.37 (0.45)

Table 1: *Local incoherence values of the calculated distances for all data sets using the algorithms described in Section 3. The values between parentheses are based on the normalized distances, while the other values are based on the unnormalized distances. Lower values within a column indicate better results.*

For the Belgian and Netherlandic dialectal data we can observe slightly improved results (lower local incoherence) using both the leven-stress and the leven-cohort algorithms as compared to the regular Levenshtein algorithm. The leven-stress algorithm yields the best performance, while the exponential leven-cohort algorithm performs worst. In contrast, for the Norwegian data a much better performance in terms of local incoherence is obtained using the regular Levenshtein algorithm.

When dividing the Belgian and Netherlandic data in a group consisting of all (387) polysyllabic words having an average length of 5 tokens and a group of all (175) monosyllabic words having an average length of 3.8 tokens, we found that the leven-stress and leven-cohort algorithms gave improved results over the regular Levenshtein algorithm only in the polysyllabic group. In the monosyllabic group, the results were similar.

When inspecting the correlations of the Norwegian dialect distances with the perceptual distances in Table 2 we also observe that the best performance results from the regular Levenshtein algorithm. However, Heeringa et al. [13] mentioned that, since dialect distances satisfy the triangle inequality (i.e. $\forall x, y, z : d(x, y) \leq d(x, z) + d(z, y)$), the dialect distances cannot be seen as independent observations. We analyze the relationship between the computed dialect distances and the perceptual distances by calculating the correlation coefficient, but its statistical significance cannot be assayed in the usual way, e.g., via a table in a statistics test or via a software package such as SPSS or R. To solve this problem the Mantel test [2] can be used, which determines the significance of the correlation by repeatedly permuting the matrix rows and columns and recalculating the correlation coefficient. By using this method, Heeringa et al. [13] found that the correlation coefficients needed to differ by more than 0.1 to indicate statistical significance (when applied to the Norwegian data). Hence, the different algorithms all yield similar performance on the Norwegian data set.

	Correlation r
exponential	0.63 (0.64)
linear	0.63 (0.64)
square root	0.63 (0.64)
logarithmic	0.64 (0.64)
stress	0.66 (0.64)
regular	0.66 (0.66)

Table 2: *Correlations of the calculated distances using the using the algorithms described in Section 3 with the Norwegian perceptual data. The values between parentheses are based on the normalized distances, while the other values are based on the unnormalized distances. Higher values indicate better results.*

As mentioned earlier, Heeringa et al. [13] indicated that normalizing the Norwegian dialect distances reduced performance. However as can be seen in Table 1 this is not the case for the Netherlandic and Belgian distances. Normalizing the Netherlandic distances improves results slightly, while normalizing the Belgian distances improves results more clearly. Furthermore, Table 2 also shows no reduced performance for the normalized Norwegian dialect distances when correlating them with the perceptual distances as compared to the unnormalized distances. Because the average length of the Norwegian data is only 3.5 tokens, our position-sensitive weightings have relatively little opportunity to distinguish themselves. It is therefore not surprising that the leven-cohort and leven-stress approaches perform roughly the same as the regular Levenshtein algorithm.

Even though the algorithms introduced in Section 3 calculate dialect distances using different approaches, the results are very similar. The Norwegian dialect distances calculated with the leven-stress and leven-cohort algorithms correlated highly with the regular Levenshtein distances ($r > 0.97$). This was also the case for Belgium ($r > 0.97$) and the Netherlands ($r > 0.95$). Because of these high correlations, the dialectal maps based on the adapted algorithms resemble the maps obtained using the regular Levenshtein distance which were discussed in [29] a great deal.

To give an example of the high level of similarity between the results of the regular and the adapted Levenshtein distance algorithms, Figure 3 shows the dialectal maps for the results obtained using the regular Levenshtein algorithm (top) and the logarithmic leven-cohort algorithm (bottom).

The maps on the left show a clustering in ten groups based on UPGMA (Unweighted Pair Group Method with Arithmetic mean; see [12] for a detailed explanation). In these maps phonetically close dialectal varieties are marked with the same symbol. However note that the symbols can only be compared within a map, not between the two maps (e.g., a dialectal variety indicated by a square in the top map does not need to have a relationship with a dialectal variety indicated by a square in the bottom map). Because clustering is unstable, in that small differences in input data can lead to large differences in the classifications derived, we repeatedly added random small amounts of noise to the data and iteratively generated the cluster borders based on the noisy input data. Only borders which showed up during most of the 100 iterations are shown in the map. The maps in the middle show the most robust cluster borders; darker lines indicate more robust borders. Finally, the maps on the right show a vector at each locality pointing in the direction of the region it is phonetically most similar to.

5 Discussion

In this study we have developed a number of algorithms to calculate dialect distances based on theories of spoken word recognition. Unfortunately these algorithms did not show consistent results across all datasets. While improved results were found on the GTRP datasets using the adapted algorithms, this was not the case for the Norwegian dataset. We emphasize that our results do *not* reflect on the theories of word recognition we employed, as word recognition and the recognition of signals of geographical or social identity may be very different.

There are also some differences between the Norwegian dataset and the GTRP datasets which are worth mentioning. First, the Norwegian dataset is very small (less than 1000 items in total) compared to the size of the GTRP datasets (both consist of more than 100,000 items). Due to the small size and the fact that dialect distances are not statistically independent, it is almost impossible to find significant differences between the results of the different algorithms using the Norwegian perceptual data [13]. Second, there is a substantial difference between the average word length for the GTRP data and the Norwegian data. The average word length in the GTRP data is 4.7 tokens, while it is only 3.5 tokens for the Norwegian data. Because our algorithms employ a cost function based on position and word length, this likely influences the results. For example, consider the leven-stress algorithm which weighs differences in the first three tokens more heavily. Because in the Norwegian dataset the average word consists of only slightly more than three tokens, the leven-stress approach will almost be equal to the regular Levenshtein algorithm. Additionally, we found that the improvement over the regular Levenshtein algorithm in the GTRP datasets was not present in the group of monosyllabic words, emphasizing the importance of word length using the leven-cohort and leven-stress algorithms.

Because Heeringa [13] also found differing results between the Norwegian dataset and another larger (German) dataset, we are not entirely sure that the Norwegian dataset due to its limited size and small word length is optimal for evaluating dialect distance algorithms. To verify the quality of the Norwegian dataset we think it would be useful to obtain perceptual dialect distances of another bigger dataset having a larger average word length and use these for comparison. If it turns out that the Norwegian dataset is indeed unrepresentative, then we would be encouraged by the consistently superior results obtained when we weighted initial segments more heavily in the GTRP datasets.

All the leven-cohort algorithms perform similarly. It would be interesting to experiment with more diverse methods which emphasize the beginnings of the words. For instance by placing additional weight only on the first phoneme of the word or on the part before the recognition point (i.e. the point after which subsequent phones do not change the recognized word in the Cohort model).

The leven-stress algorithm described in Section 3.2 uses an approximation of the position and length of the stressed syllable. It would be interesting to evaluate the performance of this algorithm when the exact position and length of the stressed syllable can be used instead (perhaps by using automatic syllabification). Furthermore, it would be very appealing to compare the performance of the leven-stress algorithm to the performance of the leven-cohort algorithm on a dataset where stress is predominantly placed on the final syllable (and/or on a dataset where stress is variable). In the one case the leven-stress algorithm weighs differences at the end of the words more strongly (and in the other it weighs variably depending on stress), while the leven-cohort algorithm weighs differences at the start of the words more strongly.

The fact that we obtained different results on the Dutch data and the Norwegian data might suggest that we are overlooking a confounding variable. The literature on similarity indicates that rhymes (vowels plus codas) are more important than onsets (initial material) [11, 22, 27]. Although the results seem to contradict this directly, we were concerned that syllable structure might need to be controlled for, so we also examined the result of weighting later elements more heavily, restricting our attention to the monosyllabic subsets, effectively inverting the weighting used by the leven-cohort model. But using increasing weights on word-final segments did not improve results.

Besides applying position-dependent weighting, another sensible approach could be to weight edit operations based on the type of the sound segments involved. For instance, there is evidence that consonants and vowels are not equally important in word recognition. Several studies found that correcting a non-word

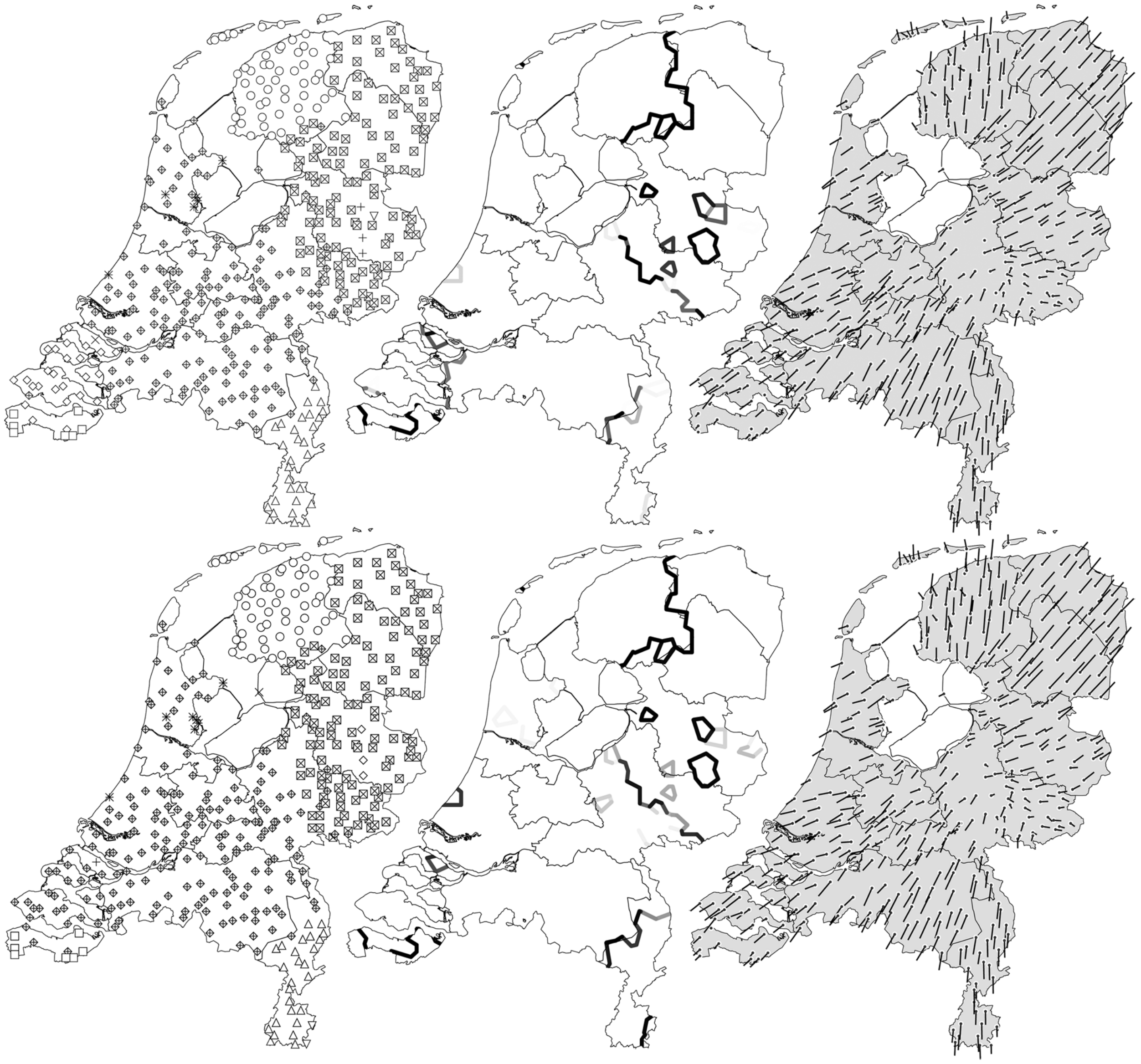

Fig. 3: *Dialect distances for regular Levenshtein method (top) and logarithmic leven-cohort method (bottom). The maps on the left show the ten main clusters for both methods, indicated by distinct symbols. Note that the shape of these symbols can only be compared within a map, not between the top and bottom maps. The maps in the middle show robust cluster borders (darker lines indicate more robust cluster borders) obtained by repeated clustering using random small amounts of noise. The maps on the right show for each locality a vector towards the region which is phonetically most similar. See Section 4 for further explanation.*

into an intelligible word is easier to do when there is a vowel mismatch than a consonant mismatch [5, 18, 26], e.g. *teeble* → *table* versus *teeble* → *feeble*. It would be interesting to adapt the Levenshtein distance algorithm to incorporate this assumption, for instance by assigning lower costs to vowel-vowel substitutions than for consonant-consonant substitutions.

Together with the adapted Levenshtein algorithms, we also introduced a normalization method for the new algorithms which respects the constraint that similarity and distance be each other's inverses. In contrast to Heeringa et al. [13] we do not find support for preferring unnormalized distances over normalized distances. However this does not contradict their results. In our algorithms a stronger bias towards longer words is present than in their study, hence normalization is more important.

Even though there are differences in performance on the GTRP datasets and the Norwegian dataset, we found that the dialect distances calculated using the adapted algorithms for a single dataset were highly similar to the results obtained with the regular Levenshtein algorithm. A possible cause for this similarity is the aggregate level of analysis; we are looking at the language level instead of the word level. As a better indicator of the performance of the adapted Levenshtein algorithms, it would be very useful to examine the performance on the word level. For instance by evaluating the algorithms on the task of recognizing cognates [16].

Acknowledgments

We thank Prof. Veronika Ehrich, University of Tübingen, for an insightful question which led to this study. We thank the Meertens Instituut for making the Goeman-Taeldeman-Van Reenen-Project data available for research and especially Boudewijn van den Berg for answering our questions regarding this data. We would also like to thank Peter Kleiweg for providing support and the software we used to create the maps. Last but not least we are very thankful for the useful comments of the anonymous reviewers.

References

[1] G. Altman and D. Carter. Lexical stress and lexical discriminability: Stressed syllables are more informative, but why? *Computer Speech and Language*, 3:265–275, 1989.

[2] E. Bonnet and Y. van de Peer. zt: a software tool for simple and partial Mantel tests. *Journal of Statistical Software*, 7(10):1–12, 2002.

[3] K. Bühler. *Sprachtheorie. Die Darstellungsfunktion der Sprache.* Gustav Fischer, Jena, 1934.

[4] R. Cole. Listening for mispronunciations: A measure of what we hear during speech. *Perception and Psychophysics*, 13:153–156, 1973.

[5] A. Cutler, N. Sebastian-Galles, O. Soler-Vilageliu, and B. van Ooijen. Constraints of vowels and consonants on lexical selection: cross-linguistic comparisons. *Memory & Cognition*, 28(5):746–55, 2000.

[6] A. Cutler, D. Dahan, and W. van Donselaar. Prosody in the comprehension of spoken language: A literature review. *Language and Speech*, 40(2):141–201, 1997.

[7] G. de Schutter, B. van den Berg, T. Goeman, and T. de Jong. *Morfologische Atlas van de Nederlandse Dialecten (MAND) Deel 1.* Amsterdam University Press, Meertens Instituut - KNAW, Koninklijke Academie voor Nederlandse Taal- en Letterkunde, Amsterdam, 2005.

[8] M. Ellison and S. Kirby. Measuring language divergence by intra-lexical comparison. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the ACL*, pages 273–280, Morristown, NJ, 2006. Association for Computational Linguistics.

[9] T. Goeman and J. Taeldeman. Fonologie en morfologie van de Nederlandse dialecten. een nieuwe materiaalverzameling en twee nieuwe atlasprojecten. *Taal en Tongval*, 48:38–59, 1996.

[10] C. Gooskens. Traveling time as a predictor of linguistic distance. *Dialectologia et Geolinguistica*, 13:38–62, 2005.

[11] U. Hahn and T. M. Bailey. What Makes Words Sound Similar? *Cognition*, 97(3):227–267, 2005.

[12] W. Heeringa. *Measuring Dialect Pronunciation Differences using Levenshtein Distance.* PhD thesis, Rijksuniversiteit Groningen, 2004.

[13] W. Heeringa, P. Kleiweg, C. Gooskens, and J. Nerbonne. Evaluation of string distance algorithms for dialectology. In J. Nerbonne and E. Hinrichs, editors, *Linguistic Distances*, pages 51–62, Shroudsburg, PA, 2006. ACL.

[14] P. W. Jusczyk and P. A. Luce. Speech perception and spoken word recognition: Past and present. *Ear and Hearing*, 23(1):2–40, 2002.

[15] G. Kondrak and B. Dorr. Automatic identification of confusable drug names. *Artificial Intelligence in Medicine*, 36(1):29–42, 2006.

[16] G. Kondrak and T. Sherif. Evaluation of several phonetic similarity algorithms on the task of cognate identification. *Proceedings of the Workshop on Linguistic Distances*, pages 43–50, 2006.

[17] P. A. Luce and C. T. McLennan. Spoken word recognition: The challenge of variation. In D. Pisoni and R. Remez, editors, *The Handbook of Speech Perception*, pages 591–609, Oxford, 2005. Blackwell Publishing.

[18] E. A. Marks, D. R. Moates, Z. S. Bond, and V. Stockmal. Word reconstruction and consonant features in English and Spanish. *Linguistics*, 40(2):421–438, 2002.

[19] W. D. Marslen-Wilson. Functional parallelism in spoken word-recognition. *Cognition*, 25(1-2):71–102, 1987.

[20] W. D. Marslen-Wilson and A. Welsh. Processing interactions and lexical access during word recognition in continuous speech. *Cognitive Psychology*, 10:29–63, 1978.

[21] W. D. Marslen-Wilson and P. Zwitserlood. Accessing spoken words: The importance of word onsets. *Journal of Experimental Psychology. Human Perception and Performance*, 15(3):576–585, 1989.

[22] D. L. Nelson and L. D.Nelson. Rated acoustic (articulatory) similarity for word pairs varying in number and ordinal position of common letters. *Psychonomic Science*, 19:81–82, 1970.

[23] J. Nerbonne and P. Kleiweg. Toward a dialectological yardstick. *Journal of Quantitative Linguistics*, 14(2):148–167, 2007.

[24] J. Taeldeman and G. Verleyen. De FAND: een kind van zijn tijd. *Taal en Tongval*, 51:217–240, 1999.

[25] B. L. van den Berg. *Phonology & Morphology of Dutch & Frisian Dialects in 1.1 million transcriptions.* Goeman-Taeldeman-Van Reenen project 1980-1995, Meertens Instituut Electronic Publications in Linguistics 3. Meertens Instituut (CD-ROM), Amsterdam, 2003.

[26] B. van Ooijen. Vowel mutability and lexical selection in English: evidence from a word reconstruction task. *Memory & Cognition*, 24(5):573–83, 1996.

[27] P. Vitz and B. S. Winkler. Predicting the judged 'similarity of sound' of English words. *Journal of Verbal Learning and Verbal Behavior*, 12:373–388, 1973.

[28] A. Walley. Spoken word recognition by young children and adults. *Cognitive Development*, 3:137–165, 1988.

[29] M. Wieling, W. Heeringa, and J. Nerbonne. An aggregate analysis of pronunciation in the Goeman-Taeldeman-Van Reenen-Project data. *Taal en Tongval*, 59(1), 2007.

[30] M. Wieling, T. Leinonen, and J. Nerbonne. Inducing sound segment differences using Pair Hidden Markov Models. In J. Nerbonne, T. Ellison, and G. Kondrak, editors, *Computing and Historical Phonology: 9th ACL Special Interest Group for Morphology and Phonology*, pages 48–56, 2007.