Extending Levenshtein edit distance with phonological features

Abstract

The paper presents an automated task-based evaluation framework for an extension to the Levenshtein edit distance metric. This extension explicitly represents linguistic phonological features of compared characters so the metric can use information about characters’ internal structure rather than treat them as elementary atomic units of comparison. The proposed framework allows us to test alternative configurations of phonological features, and automatically select feature arrangements, which on a range of evaluation parameters systematically show greater improvements over the baseline Levenshtein edit distance.

Introduction

This paper present a methodology for the development and automated evaluation of a linguistic features set that extends the traditional Levenshtein edit distance metric used for the task of cognate identification.

Cognate identification is important for a range of different applications; in our experimental settings we use it for assisting MT developers in creating cognate lexicon for hybrid MT between closely related languages, some of these languages are under-resourced (e.g., Ukrainian and Russian, Portuguese and Spanish, Dutch and German). For many of such language pairs there are no electronic dictionaries, and there are only small parallel corpora available with limited lexical coverage. Typically these parallel corpora can provide translations for frequently used general words, but miss the ‘long tail’ of less frequent, often topic-specific or terminological words. However, in closely related languages these words are often cognates, which creates a possibility to rapidly extend bilingual lexicons in semi-automated way using non-parallel, comparable corpora and automated cognate identification techniques.

In this task, cognate candidates are generated from word lists created from large monolingual comparable corpora in both languages The assumption is that the developers have good linguistic intuition of both languages and work through lists of cognate candidates, checking which pairs can be added to the bilingual dictionary. Their productivity depends on whether cognates are presented high up in the list of candidates, ideally at the top, or at least in the top N items, where N lines should at most fit on a single screen.

Levenshtein edit distance (Levenshtein, 1966) is typically used to compare word pairs from different languages and determine if they are cognate candidates. It is computed for every pair of words in the two word lists (their Cartesian product), the search space may be restricted to matching part-of-speech codes, if this annotation is available for the corpus. However, there are several problems with the traditional Levenshtein metric, one of which is that all characters are treated equally. As a result, words that are intuitively close may receive a large distance score, e.g.,

Ukrainian (Uk) “жовтий” (zhovtyi)

Russian (Ru) “жёлтый” (zheltyi) = ‘yellow’ (Lev distance = 3),

where, for historical reasons, similar sounds are represented by different characters: [o] by Uk ‘о’ and Ru ‘ё’, [y] by Uk ‘и’ and Ru ‘ы’. On the other hand, words that are not cognates and are intuitively far apart, still receive the same distance scores, such as:

Uk “жовтий” (zhovtyi) = ‘yellow’ and

Ru “жуткий” (zhutkiy) = ‘dismal’ (Lev = 3).

Some modifications and extensions of the Levenshtein metric introduce weightings for different character mapping, but these weights need to be set or empirically determined for each specific mapping: compared characters do not have internal structure, so there is no way to predict the weights in advance for any possible pair in a principled way.

In this paper we present an automated task-based evaluation framework for an extension of the Levenshtein edit distance metric, which explicitly represents linguistic phonological features of compared characters so the metric can use information about characters’ internal structure rather than treat them as elementary atomic units of comparison. These distinctive features have been proposed for modeling dialectological variation and historical changes in languages (Nerbonne & Heeringa, 1997), and may be also applied to a range of computational linguistic tasks, such as transliteration, morphological alternations and cognate identification. However, there are multiple ways of identifying, representing and structurally arranging these features, there is a need for a methodology for evaluating alternative feature configurations. In our previously published pilot experiment, a smaller-scale manual evaluation indicated the need to use hierarchical phonological feature structures for consonants rather than flat feature vectors used in dialectology (Anonymous, 2016). Therefore, there is a need for developing a systematic automated evaluation methodology for designing and calibrating feature structures.

Here we present a framework for evaluating different arrangements of phonological features using the task of automated cognate identification. Apart from practical applications mentioned above, this task also has standardized evaluation metrics, e.g., cognate in top-N candidates, which can be applied automatically and used in the feature-engineering task for selection of optimal feature arrangements. Our evaluation results show greater improvement for hierarchical feature structures, on a number of automatically computed parameters, compared to our previous pilot experiment and also rule out unproductive modifications of the metric.

that flat feature vectors, which are traditionally used for modeling dialectological variation, are not sufficient, and there is a need for hierarchical feature structures.

These results agree with our earlier pilot experiment that used manual evaluation on a smaller scale but give more detailed range of evaluation parameters and systematically show greater improvements over the baseline Levenshtein edit distance.

that achieve best performance

often uses Levenshtein edit distance, where phonological feature structures can be embedded. It also

confirm our that flat phonological

In our earlier work manual evaluation methodology

calibrating the metric.

for feature engineering

for there is a need to

Phonological distinctive features and their application for cognate identification

The theory of phonological distinctive features, first proposed by Roman Jacobson (Anderson, 1995: 116), associates each phoneme (an elementary segmental unit of speech which can distinguish meanings) with a unique set of values for categories, which may apply to larger classes of sounds. For example, phoneme [t] has the following values for the categories:

‘type’: consonant

‘voice’: unvoiced

‘maner of articulation’: plosive;

‘active articulation organ’: front of the tongue

‘passive articulation organ’: alveolar

Phoneme [d] has the same articulation, but is pronounced with the use of vocal cords, so it differs only in the value of one distinctive feature,

‘voice’: voiced

with all other categories and values remaining the same.

In historical development of languages and in morphological variation within a language the sound changes more often apply only to values of certain distinctive features within characters, but not to the whole category-value system, e.g.: Ge “Tag” = Nl “dag” (‘day); Ge: “machen” = Nl “maken” (‘make’). Therefore, in languages where the writing system is at least partially motivated by pronunciation, it would be useful to represent the phonological distinctive features, in order to differentiate between varying degrees of closeness for different classes of characters, e.g., vowels, sonants and consonants, or sounds with identical or similar articulation. Greater closeness between characters in terms of their phonological features has important linguistic and technical applications, such as modeling dialectal variation, historical change, morphological and derivational changes in words, such as stem alternations in inflected forms.

Phonological distinctive features have been integrated into the Levenshtein distance metric in the following way (c.f. Anonymous, 2016: 123):

1. Substitution cost for two characters is calculated as 1 [minus] F-measure between Precision and Recall of their feature overlap; this allows calculating the cost for character with different numbers of features, and the metric remains symmetric. Intuitively, to substitute [t] with [d] in “Tag” 🡪 “dag” we need to re-write only one feature out of 5, so the cost is 0.2 rather than 1.
2. The order of matching the distinctive features was found to be important. Sections 3 and 4 describe an experiment on comparing two different arrangements of features: as flat feature vectors and as feature hierarchies, where matching lower level features is a pre-condition for attempting to match lower level features. Hierarchical organization achieved better performance compared to traditional flat feature vectors. Intuitively this means that not all feature categories should be treated equally, some of them have higher priority and license comparison of lower level features
3. Insertion and deletion costs have been calibrated for the range between 0.2 and 1 using the proposed evaluation framework, described in this paper in Section 3. Optimal performance on cognate identification was achieved for cost of insertion = deletion = 1.

For the task of cognate identification, the introduction of these features distinguishes different types of character substitutions and gives more accurate prediction of the degree of closeness between compared characters and words, e.g., for the word pairs discussed above, where Levenshtein distance =3 in both:

Ukrainian (Uk) “*жовтий*” (zhovtyi) = ‘yellow’

ж

'type:consonant',

'voice:ff-voiced',

'maner:ff-fricative', '

active:ff-fronttongue',

'passive:ff-palatal'

о

'type:vowel',

'backness:back',

'height:mid',

'roundedness:rounded',

'palate:nonpalatalizing'

в

'type:consonant',

'voice:fl-voiced',

'maner:fl-fricative',

'active:fl-labial',

'passive:fl-bilabial'

т

'type:consonant',

'voice:pf-unvoiced',

'maner:pf-plosive',

'active:pf-fronttongue',

'passive:pf-alveolar']

и

'type:vowel',

'backness:front',

'height:closemid',

'roundedness:unrounded',

'palate:nonpalatalizing']

й

'type:consonant',

'voice:am-sonorant',

'maner:am-approximant',

'active:am-midtongue',

'passive:am-palatal'

Russian (Ru) “*жёлтый*” (zheltyi) = ‘yellow’ (non-matching characters)

ё // highlighted matches with Uk “о”

**'type:vowel',**

**'backness:back',**

**'height:mid', '**

**roundedness:rounded',**

'palate:palatalizing'

л // highlighted matches with Uk “в”

**'type:consonant',**

'voice:lf-sonorant', '

maner:lf-lateral',

'active:lf-fronttongue',

'passive:lf-alveolar'

ы // highlighted matches with

**'type:vowel',**

'backness:central',

**'height:closemid',**

**'roundedness:unrounded',**

**'palate:nonpalatalizing'**

Calculation of the GLev metric:

***0.0*** *1.0 2.0 3.0 4.0 5.0 6.0*

*1.0* ***0.0*** *1.0 2.0 3.0 4.0 5.0*

*2.0 1.0* ***0.2*** *1.2 2.2 3.2 4.2*

*3.0 2.0 1.2* ***1.0*** *2.0 3.0 4.0*

*4.0 3.0 2.2 2.0* ***1.0*** *2.0 3.0*

*5.0 4.0 3.2 3.0 2.0* ***1.2*** *2.2*

*6.0 5.0 4.2 4.0 3.0 2.2* ***1.2***

cf.: Metric calculated with Ru “*жуткий*” (zhutkiy) = ‘dismal’ , where GLev = 2.0 > 1.2

**0.0** 1.0 2.0 3.0 4.0 5.0 6.0

1.0 **0.0** 1.0 2.0 3.0 4.0 5.0

2.0 1.0 **0.2** 1.2 2.2 3.2 4.2

3.0 2.0 1.2 **1.0** 1.2 2.2 3.2

4.0 3.0 2.2 2.0 **1.8** 2.2 3.0

5.0 4.0 3.2 3.0 2.8 **2.0** 3.0

6.0 5.0 4.2 4.0 3.8 3.0 **2.0**

/// we should not treat all the features equvally – some features are more equal than others

This relies on the property that.

changes often affect individual values rather than more complex systems of categories

change more frequently than , while more likely to be affected than the

not to the whole set

in variations apply to certain properti change only certain distinctive features rather than entire characters

variation often applies to

is intuitively appealing, but it also

between c which more often alternate in speech in different forms of the same word, in

associated with different characters is the basis for differentiating between varying degrees of closeness between different classes of ch

finer-grained and intuitively appealing

, which represents

justifies an intuitive notion of different degree of closeness between different types of characters.

and to differentiate between varying degrees of closeness between the

which gives them internal structure and shows

rather than using characters as elementary units.

,

allows the metric to rewrite

distinctive phonological features for characters

an extension to Levenshtein edit distance metric with linguistic features which are based on

and an automated task-based

automated task-based evaluation framework for

a task-based evaluation framework for an extension to features

The paper presents an extension

a feature engineering task for

Evaluation framework

The paper presents the development of a task-based evaluation framework

The need to extend --

results of an experiment on

References

Anderson, S. R. (1985). Phonology in the twentieth century: Theories of rules and theories of representations. University of Chicago Press

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. Soviet Physics Doklady 10 (8): 707–710.

Nerbonne, J., & Heeringa, W. (1997). Measuring dialect distance phonetically. In Proceedings of the Third Meeting of the ACL Special Interest Group in Computational Phonology (SIGPHON-97).