Comparison of distance measures for historical spelling variants

S. Kempken, W. Luther, and T.Pilz

Institute of Computer Science and Interactive Systems
Universität Duisburg-Essen
D-47048 Duisburg, Lotharstr. 65, Germany
{kempken, luther, pilz}@informatik.uni-duisburg.de

Abstract. This paper describes the comparison of selected distance measures in their applicability for supporting retrieval of historical spelling variants (hsv). The interdisciplinary project Rule-based search in text databases with nonstandard orthography develops a fuzzy full-text search engine for historical text documents. This engine should provide easier text access for experts as well as interested amateurs. The FlexMetric framework enhances the distance measure algorithm found to be most efficient according to the results of the evaluation. This measure can be used for multiple applications, including searching, post-ranking, transformation and even reflection about one's own language.

1 Introduction

In recent years, many countries have started retro-digitization projects of precious originals. Events like the disastrous fire in the German Herzogin Anna Amalia Library, a World Heritage Site, in September 2004 show plainly the importance of such preservation, at least of the intellectual contents. Furthermore, these projects make accessible historical texts by building digital libraries that are of interest to scholars of all text-focused disciplines (philologists, historians, linguists, etc.) as well as interested amateurs. Right now, more than one hundred scientific initiatives are involved in the digitization of text collections, electronic editions, rare manuscripts, dictionaries, charters and illustrated books. Most of these initiatives provide digitized facsimiles, some offer additional full text. Hockey [11] provides a survey of important international projects.

The amount of time required to build a digital archive is not to be underestimated. Therefore, many retro-digitization projects focus on the constructional steps of the digitization process, which involve digitizing as well as tagging and aligning the text. Subsequent steps, like manual post processing or elaborate search functions, often need to be put at the bottom of the list. Compact Memory, a project for the digitization of historical Jewish periodicals, for example, combines a comely interface with a respectable archive and is well used. But, as it is a publicly funded project, the operator cannot devote his resources to manually revising optical character recognition (OCR) errors in the digitized texts or to offering advanced search capabilities. A reliable search engine, however, is the means that makes the data fully accessible.

Particularly historical but also regional texts often involve another important problem, apart from OCR errors: they contain spelling variants. German texts prior to 1901, when a major reform of German orthography took place, are not standardized. The result is a reduced recall ratio in those texts, due to queries that do not cover all possible spellings. The frequency of variant spelling increases significantly with the age of the text documents. Figure 1 shows the amount in percent of nonstandard tokens in 35 historical German texts from 1463 to 1876.

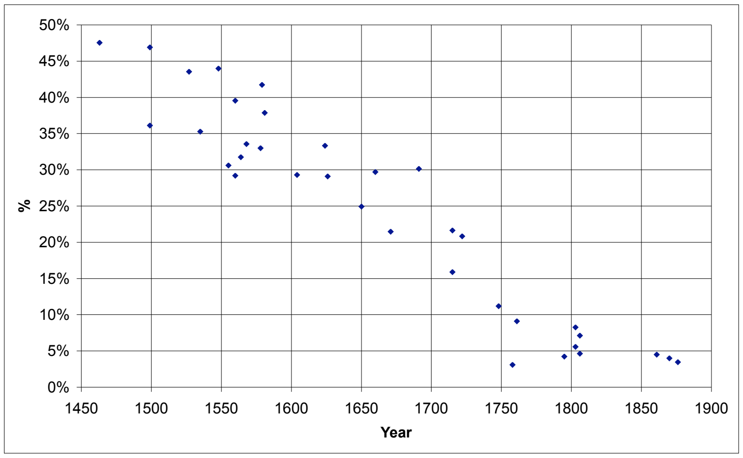

Fig. 1. Frequency of variant spellings in historical text documents

Historical spellings are by no means solely a German problem. Spelling variation is known to occur in English historical corpora also. An initiative by the University Centre for Computer Research on Language (UCREL) of Lancaster University and the University of Central Lancashire (UCLan) has developed a VARiant Detector (VARD) trained on 16th to 19th century data [18].

The interdisciplinary project Rule-based Search in Text Databases with Nonstandard Orthography (RSNSR) supported by the Deutsche Forschungsgemeinschaft (DFG [German Research Foundation]) is currently developing a fuzzy trainable full-text search engine for historical text documents [17]. Since our main focus is the time period from 1700 - 1900, regarding the results shown in Figure 1, 2 - 25% variant spellings are estimated for those texts. In the worst case, up to one quarter of a text will consist of nonstandard spellings.

In contrast to capacious glossary projects like the *Deutsches Rechtswörterbuch (DRW)* of the *Heidelberger Akademie der Wissenschaften* or *Das Deutsche Wörterbuch von Jacob und Wilhelm Grimm auf CD-ROM und im Internet (DBW)* of the Universität Trier, RSNSR uses linguistic as well as statistical rules to represent highly varied spellings. These rules can be automatically derived from evidence data with the possibility of further expert adjustment. This allows a search engine to proceed successfully even for rare spellings without the need of extensive manual operation. A Java-based search engine with a phonetic rule set has already been built [16]. Future versions will be easily integrable into other projects. We are already cooperating with Deutsch Diachron Digital (DDD) [5], which contains texts from Old High German to Modern German. In 2006, after a prototyped solution has been achieved, we plan to integrate the fully functional search engine into the retro-digitization project Nietzsche-CD. In cooperation with UCREL and UCLan we are currently researching the possibilities for a rule-based search engine for Indo-European languages [1].

2 Requirements for hsv-distance measures

One of the main operational points in building a search engine for historical spelling variants is a reliable distance measure. Such a measure can be used in different stages of a query and therefore in more than one module of the engine:

- *Search.* Text retrieval on text in non-standard orthography is obviously more difficult than usual text retrieval. Most standard information retrieval systems build up an index of occurring terms, allowing the user to quickly find all documents containing the words he queried for. As mentioned above, an exact search may not yield good results for historical texts. An adequate distance measure operating on spelling variants provides arbitrary degrees of search fuzziness within a reasonable retrieval time. Standard fuzzy search, though, is of limited use as it does not take linguistic features into account. For example, if the user queries for the German term `urteil` (=judgment), the well known Levenshtein algorithm [14] does not differentiate between the existing variant `urtheil` and, for instance, `ubrteil` with respect to the string distance. A measure that takes heed of linguistic connections will be able to determine the actual variant from a list of candidates.
- *Ranking of Boolean results.* Retrieval in historical text documents is also possible starting from a given query term, using automatically or manually built rules that generate spelling variants. The variants produced are used for Boolean retrieval returning unclassified results. Afterwards, an hsv-distance measure is required to rank the results according to their distance to the term queried.

– *Transformation.* Historical spelling variants should be automatically transformed into their modern counterparts. The hsv-distance measure is used to identify the correct spelling in a dictionary.
– *Reflection.* The differences between a historical or regional spelling variant and its modern equivalent are often hard to evaluate, even for native speakers. An hsv-distance measure is a means of mapping linguistic distinctions on a single number. The visualization of word distances supports the reflection about language as being in a state of constant change.

The amount of support a distance measure can provide depends on its practicability in the particular context of historical spelling variants. Given the abundance of different distance measures and edit-distances available, a thorough evaluation is needed.

3 Comparative study of distance measures

In this section, we briefly describe the measures we compared regarding their retrieval effectiveness: the string edit distance, distances based on an evaluation of n-grams and the Editex algorithm by Zobel and Dart [21], a stochastic distance measure and the new hvs-distance measure computed with our FlexMetric algorithm.

The string edit distance is defined as the minimum number of edit operations needed to transform the one string into the other. These operations consist of character replacements, insertions and deletions. Levenshtein [14] presented a recursive algorithm for calculating the edit distance: Let the function $d(i, j)$ denote the costs needed to transform the first i characters of the string s into the first j characters of the string t. Then the following equations hold obviously:

$$d(0, 0) = 0, d(i, 0) = i, d(0, j) = j.$$

The complete edit distance for the two strings can then be calculated using the following recursive equation:

$$d(i + 1, j + 1) = min \begin{pmatrix} d(i + 1, j) + 1, \\ d(i, j + 1) + 1, \\ d(i, j) + cost(s_i, t_j) \end{pmatrix}, cost(a, b) = \begin{cases} 0 & \text{if a=b} \\ 1 & \text{otherwise} \end{cases}$$

A more efficient way is to use a dynamic programming approach, as described by Wagner and Fischer [20]. The string edit distance is widely used in a variety of applications as it can be determined efficiently and delivers good results.

Another type of string distance measure relies on the comparison of the n-grams derived from each of the strings. The term n-gram denotes a continuing sequence of n characters. Using padding tokens, $(l + n - 1)$ subsequences can be extracted from a particular string, where l denotes the length of the actual string. For instance, the string 'HISTORICAL' yields the following bigrams:

.H HI IS ST TO OR RI IC CA AL L.

Usually, sets of bigrams or trigrams are compared. There are several possible ways of deriving a non-negative number that represents a distance [6] derived from comparison of the n-gram sets. In our experiments, we used formula 1. In contrast to the other algorithms, it does not denote a distance but a similarity measure for the two strings x and y, where B_x denotes the set of bigrams derived from string x and B_y of string y, respectively:

$$sim(x,y) = 2\frac{|B_x \cap B_y|}{|B_x| + |B_y|} \tag{1}$$

Zobel and Dart [21] presented the Editex algorithm as a new phonetic matching technique. It combines the properties of string edit distances with letter-grouping strategies used in well-known phonetic indexing algorithms like Soundex [13] or Phonix [9]. By doing so, they achieved superior results for tasks of phonetic matching. Basically, it defines an enhancement to the simple string edit distance by introducing a more complex cost function that takes the actual characters being modified into account. Additionally, a double occurrence of characters is implicitly reduced to a single one.

Ristad and Yianilos [19] suggest a stochastic interpretation of string distances. They model them according to the probability of individual operations needed to transform one string into the other. These operations are equivalent to the character replacements, insertions and deletions used to define the string edit distance. Additionally, the probability of identity operations (e.g. `a` to `a`) is taken into account. The actual probabilities are learned from a training set of string pairs using an expectation-maximization algorithm. The authors suggest two different distance measures: the so-called Viterbi distance, which takes into account only the most likely path when transforming the start into the end string, and the stochastic edit distance, which considers all possible paths and also was the one used in our experiments.

The FlexMetric framework developed by one of the authors [12] combines the simplicity of a dynamic programming algorithm with the flexibility of defining arbitrary costs for each possible character transformation.

The basic idea is very similar to the concept behind the string edit distance. The only difference is that, rather than the number of transformations, the costs for the individual operations are taken into account. The costs for the least expensive sequence of operations required to transform the one string into the other define the distance between the two strings. The cheapest sequence can be calculated using a dynamic programming algorithm resembling the one used for evaluating the string edit distance.

As the edit operations correspond with the transformations regarded in the stochastic evaluation previously described, it is possible to derive the actual costs from the probability distribution learned using the expectation-maximization algorithm according to the following principle: The more likely a particular transformation is, the lower the costs that should be assigned to it. This way, character deviations between modern and historical spellings that occur frequently in the training set lead to cheaper corresponding transformations. Thus, the resulting distance value will also be smaller. The best results are achieved using a logarithmic transformation, as shown in [12].

4 Evaluation methodology

As there are several use cases for a hsv-distance measure and therefore several methods of evaluation, we first describe the assumptions and constraints that lead to solid quality criteria for the particular algorithms. As we concentrate on the effectiveness and not the efficiency of the algorithms, aspects like memory consumption and time needed are not taken into account.

The main problem in judging the quality of string distance measures lies in comparing their applicability for different tasks. It is obvious that a distance measure that has been specifically trained to detect certain linguistic deviations can no longer yield objective results when used to quantify a relation between spellings as it necessarily valuates the trained deviation with lower costs, leading to a shorter distance. Thus, if, for instance, the measure is used to build up a genealogical tree of spelling variants of the same term, it inherently prefers relations it was specifically trained for. This effect leads to unusable results.

In order to avoid this conflict, we have to concentrate on evaluating the potential of the various algorithms for the following text retrieval task: the user queries for the modern spelling, and all documents containing the query term or a historical variant should be returned as results.

Hence, a synthetic information retrieval system (IRS) has to be constructed consisting of a document collection, a retrieval function, and a set of queries along with relevance judgments. This allows the evaluation of the effectiveness of the algorithms with standard methods in Information Retrievalrecall and precision [2].

As we want to concentrate on the algorithms' ability to cognize connections between a query term and its historical spellings, we do not regard a collection of complete texts, but rather a list of words. This way, further factors influencing retrieval results (such as term frequency in the documents) are ignored.

We assembled a list of 3,156 unique pairs of strings, each consisting of a historical deviant spelling and the modern standard spelling. These were manually compiled from 40 historical German documents written from 1350 to 1876. Thus, a number of queries (modern spellings) and relevant answers (historical spellings) for the IRS are found.

The string edit, Editex and FlexMetric distances, can be turned into a normalized similarity function for two strings a, b according to equation 2. The stochastic distance is normalized according to equation 3. These functions yield values between 0 (no similarity / maximum distance) and 1 (identity / no distance). Thus, they can be used to classify the term collection according to the computed similarity to the query in the IRS.

$$sim(a,b) = 1 - \frac{dist(a,b)}{\max\{|a|,|b|\}} \tag{2}$$

$$sim(a,b) = \frac{\min_{c \in Testset} dist(c,b)}{dist(a,b)} \tag{3}$$

To build a collection of searchable terms and spelling variants, we use a manually maintained dictionary of 217,000 contemporary German words derived from the free spelling-correction tool Excalibur. The historical word forms found by the IRS are added to the dictionary, whereas the corresponding modern terms are removed. In this way, it is ensured that no other relevant documents (spelling variants) are in the collection. Hence, we are able to exactly determine the medium recall level after retrieving the first one to five most similar terms and the medium precision level at 100% recall as quality indicators.

If two or more terms are equidistant to the term queried, but just one of them is considered relevant, the worst case is assumed: the sequence of answers is arranged in such a way that the relevant term comes last.

A special problem arises in the case of the stochastic distance measure and the FlexMetric approach as these algorithms require a decent training set of string pairs. In order to maximize the utilization of the manually compiled list, we used cross-validation. The list is randomly split into ten parts of preferably equal length. Nine of them are used to train the distance measures. The newly trained measure is evaluated on the remaining records. This is done ten times, once for each part. The individual results are averaged afterwards.

5 Results and interpretation

Measure	Pr.	R 1	R 2	R 3	R 4	R 5
Bigram evaluation	37.9 %	24.5 %	35.6 %	42.6 %	48.2 %	54.4 %
Editex	56.1 %	43.3 %	55.2 %	63.4 %	69.2 %	72.6 %
Levenshtein	38.9 %	22.9 %	36.6 %	47.1 %	53.4 %	58.9 %
FlexMetric	55.0 %	38.6 %	58.2 %	65.7 %	70.8 %	75.0 %
Stochastic measure	62.4 %	46.7 %	65.3 %	74.7 %	79.6 %	83.1 %

Table 1. Evaluation results

The actual experimental results shown in table 1 can be summarized as follows:

– The string edit distance and n-gram algorithms yield comparable results. This was to be expected as both of them evaluate a deviation regardless of its context or the affected characters respectively.

- The Editex algorithm delivers superior results. It takes into account linguistic aspects due to its letter-grouping strategy. For example, the replacement of a vowel sound with another is in terms of a cost measure cheaper than the replacement of a vowel with a consonant sound. Also, phonetically similar letters are grouped. As our results clearly show, this strategy better reflects linguistic developments than the algorithms that process simple character transformations.
- The results yielded by the stochastic distance and the FlexMetric approach are also above those produced by the basic string edit distance and n-gram algorithms. As they both rely on the same learned probability distribution, this is not surprising. The main difference lies in their conceptual complexity: whereas the stochastic distance measure needs an extensive evaluation of the probability distribution for each term pair, the FlexMetric uses a derived cost measure in a simple dynamic programming algorithm. Hence, it allows intuitive optimizations like re-using previously calculated values for $1 : n$ comparisons. Furthermore, and most important to our field of application, the derived cost measure is more likely to be understood and optimized by a human user, for example, for linguistic analysis.
- In [12], the stochastic and FlexMetric distance delivered precision values of 73.7% and 69.0% respectively. We explain this gain in performance with the nature of the tested set. The evidences evaluated in [12] were compiled from a set of documents originating in a smaller time interval. The advantage of the trainable measures is their ability to adapt to specific features of the training set. Hence, this advantage is lost if the set of documents used for evaluation is compiled from a too broad range of origins and thus contains too many different spelling variants (cf. figure 1).

6 Conclusion

From the results shown above, we draw the following conclusions:

- The better adapted an algorithm is to specific phenomena in the domain of historical spellings, the better the retrieval results that can be expected from it.
- The paramount results of a trained distance measure can be transferred to a simpler evaluation algorithm without significant loss in quality.

In this sense, we have created a simple, easy to handle string distance measure by using a decent training set of string pairs. As an result of our evaluation, this distance measure is capable of correctly identifying unknown historical spelling variants of a given query term with an accuracy of more than 50% and is thus superior to common fuzzy search algorithms like Levenshtein string edit distance or n-gram-based comparisons. We expect a further improvement of the retrieval quality from the usage of a set of trained distance measures: By evaluating a document's metadata, that measure that has been trained on spelling variants from about the same time interval and location can be used for retrieval. The verification of this assumption is part of our current research.

7 Further work and outlook

The FlexMetric distance measure reflects properties of the spellings it was trained on. Thus, it may be used to detect the occurrence of certain deviations. The fact of their occurrence is, in turn, an indicator of the place and date of the origin of the text. Hence, the FlexMetric can be used to classify texts of unknown origin. Several measures can be trained on text evidence from different times and places. The measure that yields the best results on an unclassified text is assumedly trained on spellings occurring in a text from the same period and location.

Currently, we are developing a collection of trained measures for three time periods between 1350 and 1900 and three German language areas. The evaluation of this approach is part of our research.

The RSNSR project will provide an online search engine that can be used for literature studies by both experts and amateurs. Following the cognitions of a developed prototype, a simplistic interface will be set up. Among its functions is already a visualization of the rules used. An automatic text categorization that estimates the time and location of origin will follow soon.

This engine will then be integrated into different projects in the context of digitizing historical texts. One of these projects will be DDD. The development of our search engine is accompanied by other projects that also provide modules for successful retro-digitization and literature research. Two of these are also held at the Universität Duisburg-Essen: the development of partial text recognition software for German Fraktur fonts [15] and a web-based system for assisted literature research [3]. With these and RSNSR, a framework for the retro-digitization of historical documents is taking shape.

8 Acknowledgements

The presented research was carried out in a recent project Rule-based search in text databases with nonstandard orthography and funded by the German Research Council.

References

1. D. Archer, A. Ernst-Gerlach, S. Kempken S, T. Pilz, and P. Rayson, *The identification of spelling variants in English and German historical texts: manual or automatic?*, proposed for Digital Humanities 2006, July 4 - 9, Paris, France.
2. R. Baeza-Yates, B. Ribeiro-Neto, *Modern Information Retrieval*, Addison-Wesley, 2000.

3. D. Biella, W. Luther, T. Pilz, *A web-based system for assisted literature research*, Proceedings of the 3rd European Conference on e-Learning 2004, Nov. 25 - 26, Paris, France.
4. Bibliotheca Augustana, FH Augsburg; http://www.fh-augsburg.de/~harsch/augustana.html (accessed 05 Jan. 2006)
5. Deutsch Diachron Digital; http://www.deutschdiachrondigital.designato.de (accessed 05 Jan. 2006)
6. K. Erikson, *Approximate Swedish Name Matching - Survey and Test of Different Algorithms*, Nada report TRITA-NA-E9721, 1997.
7. Excalibur; http://www.eg.bucknell.edu/~excalibr/excalibur.html (accessed 05 Jan. 2006)
8. documentArchiv.de; http://www.documentarchiv.de (accessed 05 Jan. 2006)
9. T. Gadd, PHONIX: The Algorithm, *Program: Automated Library and Information Systems* **24**(4): pp. 363 - 366 (1990).
10. Hessisches Staatsarchiv Darmstadt; http://www.stad.hessen.de/DigitalesArchiv/anfang.html (accessed 05 Jan. 2005)
11. S. Hockey, Living with Google: Perspectives on Humanities Computing and Digital Libraries, *Literary and Linguistic Computing*, **20**: pp. 7 - 24 (2004).
12. S. Kempken, *Bewertung von historischen und regionalen Schreibvarianten mit Hilfe von Abstandsmaßen*, Thesis, Universität Duisburg-Essen (2005).
13. D. Knuth, *The Art of Computer Programming, Vol. 3: Searching and Sorting*, Addison-Wesley, pp. 391 - 392 (1973).
14. V. Levenshtein, Binary codes capable of correcting deletions, insertions and reversals, *Soviet Physics Doklady*, **10** (8): pp. 707 - 710 (1966).
15. L. Mischke, W. Luther, *Document Image De-Warping Based on Detection of Distorted Text Lines*, in: Fabio Roli, Sergio Vitulano (eds.), *Image Analysis and Processing - ICIAP 2005 proceedings*, Cagliari, Italy, September 2005, LNCS 3617, Springer, pp. 1068 - 1075.
16. T. Pilz, *Unscharfe Suche in Textdatenbanken mit nichtstandardisierter Rechtschreibung am Beispiel von Frakturtexten zur Nietzsche-Rezeption*, Thesis (civil service examination), Universität Duisburg-Essen (2003).
17. T. Pilz, W. Luther, N. Fuhr, U. Ammon, *Rule-based search in text databases with nonstandard orthography*, Proceedings ACH/ALLC 2005, June 15 - 18 Victoria, Canada.
18. P. Rayson, D. Archer, N. Smith, *VARD versus Word: A comparison of the UCREL variant detector and modern spell checkers on English historical corpora*, Proceedings of the Corpus Linguistics 2005 conference, July 14 - 17, Birmingham, UK.
19. E. Ristad, P. Yianilos, Learning String Edit Distance, *IEEE Transactions on Pattern Recognition and Machine Intelligence* **20** (5), pp. 522 - 532 (1998).
20. R. Wagner, J. Fischer, The String-to-String Correction Problem, *Journal of the ACM* **21** (1), pp. 168 - 173 (1974).
21. J. Zobel, P. Dart, *Phonetic String Matching: Lessons from Information Retrieval*, Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 166 - 172 (1996).