Journal of Universal Computer Science, vol. 18, no. 20 (2012), 2750-2770
submitted: 24/1/12, accepted: 18/9/12, appeared: 1/12/12 © J.UCS

Computational Analysis of Medieval Manuscripts: A New Tool for Analysis and Mapping of Medieval Documents to Modern Orthography

Mushtaq Ahmad
(Dept. of Computer Science, University of Pretoria, Pretoria, South Africa
Dept. of Computing Services, Athabasca University, Athabasca, Canada
mushtaqa@athabascau.ca)

Stefan Gruner
(Dept. of Computer Science, University of Pretoria, Pretoria, South Africa
sgruner@cs.up.ac.za)

Muhammad Tanvir Afzal
(Dept. of Computer Science, Mohammad Ali Jinnah University, Islamabad
Pakistan
mafzal@jinnah.edu.pk)

Abstract: Medieval manuscripts or other written documents from that period contain valuable information about people, religion, and politics of the medieval period, making the study of medieval documents a necessary pre-requisite to gaining in-depth knowledge of medieval history. Although tool-less study of such documents is possible and has been ongoing for centuries, much subtle information remains locked such manuscripts unless it gets revealed by effective means of computational analysis. Automatic analysis of medieval manuscripts is a non-trivial task mainly due to non-conforming styles, spelling peculiarities, or lack of relational structures (hyper-links), which could be used to answer meaningful queries. Natural Language Processing (NLP) tools and algorithms are used to carry out computational analysis of text data. However due to high percentage of spelling variations in medieval manuscripts, NLP tools and algorithms cannot be applied directly for computational analysis. If the spelling variations are mapped to standard dictionary words, then application of standard NLP tools and algorithms becomes possible. In this paper we describe a web-based software tool CAMM (Computational Analysis of Medieval Manuscripts) that maps medieval spelling variations to a modern German dictionary. Here we describe the steps taken to acquire, reformat, and analyze data, produce putative mappings as well as the steps taken to evaluate the findings. At the time of the writing of this paper, CAMM provides access to 11275 manuscripts organized into 54 collections containing a total of 242446 distinctly spelled words. CAMM accurately corrects spelling of 55% percent of the verifiable words. CAMM is freely available at http://researchworks.cs.athabascau.ca/

Key Words: MPEG spelling variations, mapping, phonetic algorithms
Category: I.7.1, I.7.2, I.7.m, J.5 umanities

1 Introduction

Medieval manuscripts form the majority of the remaining and preserved documents available to us today from that period. The primary use of manuscripts was to preserve ideas, knowledge, and facts, which the writers or their superiors believed were worth preserving. There are also letters through which people communicated, including contracts, bills, or other deeds of legal relevance. Those documents provide a glimpse of the people, society, political and religious beliefs and affairs of their era. When studied collectively, they offer the possibility of discovering interesting larger 'patterns' on the subjects discussed in the individual manuscripts.

However, historical documents are difficult to analyze. They were handwritten on skin (parchment) or paper. Over centuries, portions of those documents deteriorated physically which makes their writing difficult to decipher, even with help of advanced scanner and optical character recognition software. The writers of those documents are typically not known by name, and the context of their writing is often unclear from our contemporary point of view.

European orthography was not standardized until about two centuries ago. In medieval Europe, manuscripts were often written to be read out loud rather than being studied quietly; the general public (including even some members of the political leadership) was mostly illiterate. Writing was restricted to important matters only, since the writing material was expensive to obtain. Moreover, the medieval world was to a large extent regionally confined, with only little trans-regional mobility, such that many regional customs, habits and dialects were preserved. These factors also contributed to the delay of standardization in orthography.

In Germany it was only in 1880 when the first documented comprehensive effort to standardize the German orthography was released as "Vollständiges Orthographisches Wörterbuch der deutschen Sprache" (complete orthographic dictionary of the German language). This is now commonly known as "the Duden". Modern German orthography is nowadays regulated by the "Rat für deutsche Rechtschreibung", RdR (council for German orthography).

A wide range of orthography and linguistic phenomena can be found in medieval documents. They result from individual and regional orthographical habits (as mentioned above), and also from local variation in the language itself. Study of medieval documents thus requires dealing with spelling variations [PE+08], along with linguistic variations. Linguistic variations can be phonetic, morphologic, lexical, grammatical or semantic.

In the absence of standardized orthography, writers relied much on the sounds of pronunciation of the words to spell them out in writing. It is thus conceivable that the spelling of writers was influenced by their local dialects. Dialects differ in intonation, pause, and stress, (sometimes even in dictionary and grammar), but the general phonetic sequence usually remains the same (or at least sufficiently similar). For example, when stretched, the word “haus” can be pronounced as: *haaoos*, or: *huus*, but in both cases the phonetic key remains the same, namely: ‘HS’. In this context the goal of our work is to normalize historic spelling variations in historic texts to contemporary orthography, mainly for these two reasons:

– to make digital representations of historic texts better search-able, such that contemporary search-words can be used to find their historic counter parts, and

– to present historic texts more legibly to lay readers who are interested only in the contents however not in the spelling variations of those documents.

Because it is likely that a typical user will not be able to discern all possible spelling variations, for example: ‘haus’ or ‘hus’, it is necessary that a mean is devised wherein a user can receive a set of suggested spelling variations for a particular word that is retrieved from the ample database store of medieval documents.

Since a phonetic key can help to identify records across different dialects, phonetic key mapping could aid in the mapping of variant grapheme sets to standardized dictionaries. To discover spelling variations or phonetic keys generally text analysis algorithms are applied to modern printed text whereby the assumption is that the text is grammatically and lexically correct with respect to contemporary standards and has been proofread for standard-conformance before publication. For this reason most text analysis algorithms produce aberrant results when applied to documents that are rich in spelling variations and regional peculiarities such as medieval manuscripts.

Orthography does not have to provide a unique phonemic description of the words, and various different graphemes could be used to represent the same phoneme. For example, the graphemes ‘kapitel’, ‘kapittel’, and ‘capitel’ all represent the phoneme: KPTL. However only the grapheme sequence ‘kapitel’ can be found in nowadays German dictionary. Thus the graphemes defined in a modern dictionary are only a subset of all graphemes that could possibly represent a phoneme. Thus, a German dictionary word is simply a grapheme sequence, which has been designated to be the correct spelling by the RdR. Based on this observation, it could be postulated that ‘correcting’ a spelling variation simply requires mapping the alternative grapheme sequence (variation) to the normalized grapheme sequences (word defined in a dictionary or used in a corpus).

A comprehensive modern *dictionary* provides a list of words together with their definitions, etymology, phonetics, pronunciation, and lexical information. However, a dictionary does not provide all possible morphological forms of these words. A larger set of different morphological forms of words can be found in a large *corpus* such as “Europarl” Corpus [Koe05]. Thus a reference system based on a dictionary and corpus is more effective than dictionary or corpus alone.

As mentioned above, the purpose of our work is motivated by the desire to make valuable historic documents better accessible to a wider audience of modern readers, who cannot cope with the peculiarities of medieval syntax, by transforming such documents to modern orthographic forms, while still preserving and presenting the original manuscript for users or viewers who take advantage of the search function. In the work carried out for this paper, word extraction from 11275 manuscripts resulted in 242446 unique grapheme sequences. Mapping so many grapheme sequences manually is not feasible and calls for the application of tool-supported mapping algorithms. For a meaningful mapping, some common property shared by two graphemes in question is required. This common property, functioning like a meta-model, could be a phoneme, character distance, neighborhood profile, grapheme profile or some other statistical feature linking the two.

2 Related Work

Work in the wider context of our paper, with the aim of providing any software and computational support methods to the faculty of history, is called *History-Informatics* [BVG08].

As far as the particular topic of spelling variations is concerned, [EGF06] describes a probabilistic approach to search terms to generate possible historical spelling variants and produce a list of transformation rules. Spelling variants are matched against a dictionary whereby tokens are excluded. All remaining tokens are manually processed, and a list of transformation rules is produced. In [PL+06] we can find an engine for “Rule-based search in text databases with nonstandard orthography” (RSNSR). The rules used to find spelling variants are derived manually and statistically. In [PE+08] the automatic versus manual detection of spelling variations in English and German historical texts is discussed. Although the problem of normalizing spelling variations in historical documents is different from spelling correction in modern orthography, many classical approaches to spelling correction, such as the use of phonetic keys or the well-known Levenshtein distance [Lev66], can be applied in our context, too.

The problem of spelling variations in old German is explained in detail by [HH+07] from a linguistic perspective. There we can also find a wider survey of research in this field. The spelling variation problem has been classified into eight categories: new word form, Latin words, variations in word splitting, partial new word form, variation in prefixes or suffixes, typesetting variations, graphemic-phonetic variations, and new characters. From the list of those eight problems, our CAMM tool (as described in the subsequent section) attempts to solve two, namely graphemic-phonetic variations and new characters.

Nearly every phoneme can be represented by different grapheme combinations. At the same time a particular phoneme has its unique phonetic key. Based on this premise, many of the spelling correction algorithms use phonetic keys in one or more steps. By far the most widely used phonetic key generation algorithm is *Metaphone* [Phi90]. Several variants of Metaphone, such as double Metaphone or Cologne Metaphone based on Postel's algorithm [Pos69], have been published over the years for various application purposes.

Computational spelling correction methods are based on either distance-based or similarity-based methods. Similarity-based methods compute dictionary word hash keys to compute a similarity score used to evaluate similarity. *Soundex* [Knu73] and *Speedcop* [PZa84] are similarity-based phonetic algorithms for spelling correction. Distance-based methods, on the other hand, compute distance between dictionary words and misspelled words. *Correct* [Kes04] and GNU *Aspell* [Atk11] are distance-based methods. Those algorithms have largely been used to correct the spelling of words in modern languages and are also applied in the context of automated speech recognition, as further explained in the following:

– Soundex encodes consonants and vowels if a vowel is the first letter in the word. The encoded sounds are used to search for correctly spelled words. Although Soundex is designed for English, it can be adapted to be used on other languages. Daitch-Mokotoff Soundex [Mok06] is a refinement of Soundex to make it more suitable for German and Slavic words. The Klner Phonetik [Pos69] is particularly suitable for German words.

– Speedcop computes a key for every word in the dictionary by taking the first letter followed by every consonant in the order it is written, followed by the vowels in the order they appear. Each letter can appear only once in the key. A key is generated for the word in question and the keys are compared with the keys in the dictionary. The key can be varied moving forward and backward, to find suitable candidates.

– Correct is based on a model of sound-spelling correspondences in the English orthography. It ranks misspellings by the Levenshtein distance from potentially correct words, combined with the frequency of sound-spelling correspondences. The ranking is then used to compute the most probable correct spelling.

– Aspell is the standard GNU spelling corrector. It uses the Metaphone algorithm to generate phonetic keys and compares those keys against the phonetic keys of a given dictionary. Then it computes the number of changes required to change the string to a dictionary string. The string with the lowest number of required modifications is returned as the most probable correct spelling.

Although the authors of [HH+07] have explained the problem in detail and propose conceptual solutions, a software tool was hitherto not provided. POM, the Phonetic Orthography Mapper, was our own first software tool that attempted to solve the problem of normalizing spellings in medieval historical documents [ARG11]. POM uses phonetic keys and computes the likelihood of a word being spelled with a certain grapheme sequence on the basis of Hidden Markov Model (HMM) profiles to map the spelling variations. Our CAMM tool, as described in the remainder of this article, builds on POM but it also provides a word-by-word lexical and statistical analysis. Moreover it also provides a user-friendly interface to a computational analysis tool for medieval manuscripts. The aim of CAMM, in comparison to POM [ARG11], is not only to normalize medieval spelling variations, but also to enable historians to study the 'computationally enhanced' historical documents via a set of computational methods, lexical, and statistical data provided by the software tool.

3 CAMM: Computational Analysis of Medieval Documents

As mentioned above the purpose of the CAMM tool is not only provide computational analysis of medieval German manuscripts, but also to allow users to search, investigate, and annotate the manuscripts. However, the normalization of spelling variations was our main concern for this paper.

3.1 Data Source and Data Processing

Data in the Monasterium project [Kra09] [Hei10] are stored in the XML format defined by the Charter Encoding Initiative (CEI) [BVG08]. Currently the archive contains approximately 200000 digitalized historical manuscripts. This data source was chosen because of the contents of the data, data accessibility, relevance to our research project, along with the suitable format the data are stored in. Monasterium's XML archive contained 198502 XML documents at the time of our most recent access. Those documents were transferred into MySQL database storage. For our experiments 11275 manuscripts, written in medieval German, were chosen from that database. 5815163 words were extracted from the manuscripts and their frequencies were recorded. Overall there were 242486 uniquely spelled graphemes forming 88579 phonemes. On that data basis, the following six steps, further explained in the subsequent sub-sections, had to be carried out to make the CAMM tool operational:
(1) 'Shredding' XML documents to SQL
(2) Annotating paragraphs, sentences, phrases, and words from manuscripts
(3) Finding a German dictionary and converting it to a suitable SQL format
(4) Extracting paragraphs, sentences, phrases and words from Europarl corpus

(5) Devising a scoring system to rank the graphemes
(6) Creating a web interface to show the findings

3.2 Shredding XML documents to SQL

XML is useful for storing data with annotations and enabling communication between otherwise incompatible systems or data archives. Until recently fetching and manipulating data in XML was slow, thus making it unsuitable for our computationally intensive research works. However, well-known newer XML database systems such as *eXist*, *Sedna*, or *BaseX* have overcome that shortcoming by using high performance indexers such as *Lucene*. Thus the conventional argument that XML is too slow no longer holds true. Relational databases on the other hand have extensive established libraries, support, documentation, and they are flexible and easier to manage and administer locally. Our data analysis requires strong relational algebra, thus relational databases are the best choice for powerful relational algebra features. For this reason *MySQL*, an open source relational database management system, was chosen for our project.

A *shredder* is software that distributes an XML document to SQL tables. Different groups have created numerous shredders over the years such as *XLight* [ZHS10], *XPEV* [QZ+05], *XParent* [JL+02], *XRel* [YA+01], *XTRON* [MLC08], or *INode* [LNg04]. However, none of those could handle the tri-layer complex XML structure of Monasterium data which for every manuscript is stored across three different XML files in different directories. To overcome this hurdle, a new shredder called "Document, Path, Egde, and Value" (DPEV) was programmed to shred XML to SQL. Another program was written to convert the DPEV SQL tables to a normalized data model.

Figure 1 depicts the data model designed for CAMM. The German dictionary and the Europarl corpus have been used along with three algorithms (Metaphone, double Metaphone, and Cologne Phonetic) and the results have been stored in the mom_word table after processing the manuscript data.

3.3 Annotation of Paragraphs, Sentences, Phrases, and Words

Human Language is repetitive (redundant). For this reason, frequency analysis is important and fruitful in automated text analysis. A deterministic 'sliding window' algorithm was implemented to annotate paragraphs sentences and phrases. Words are extracted and their frequency and location is annotated such that each word can be mapped back to every sentence and paragraph of manuscript that it occurred in. In addition, the neighborhood of each word is also recorded. 'Neighborhood' refers to information such as words that appear to the right and left of the word, how often the word is the first or last word in a sentence, or its proximity to syntactic symbols such as a comma or question mark. Large volume

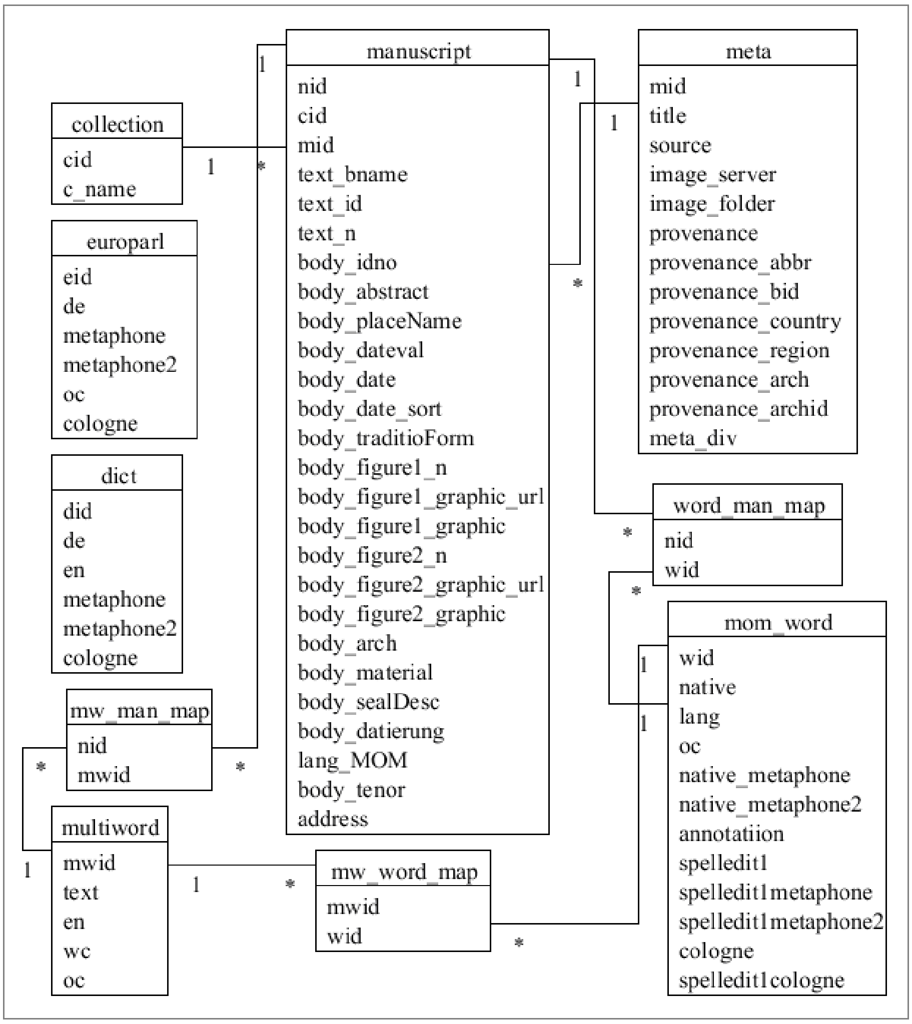

Figure 1: *Data Schema of CAMM*

of data (13.2 Gigabytes) of descriptive data is collected at this stage, which is used in later phases. An exhaustive explanation of every type of data is beyond the scope of this paper. However, this information is available online in the help and documentation files.

Until the universal usage of standardized orthography, graphemes tended to evolve faster than phonemes. This makes phoneme computation, annotation, and mapping critical for spelling variation mapping. As mentioned above the Metaphone algorithm, in addition to the Double Metaphone and Cologne Metaphone, were used to calculate and store phonemes for all words in the freedict dictionary, the Europarl corpus and the manuscripts. The algorithms map grapheme sequences to phoneme sequences. For example, ‘p’ maps phoneme P unless ‘p’ is followed by ‘h’ such that it would map to F. Metaphone is specific to English and thus produces aberrant results when used with other languages. For example, French ‘ch’ sounds like English ‘sh’, and German ‘ch’ (which has two different context-dependent pronunciations) can sound like Russian ‘kh’. Several Metaphone variations have been proposed over the years such as double Metaphone, triple Metaphone, and Cologne Metaphone. They all attempt to extend Metaphone by including sound representation of languages other than English. CAMM allows users to use Metaphone, double Metaphone, or Cologne Metaphone, and to compare results from each of them.

3.4 Dictionary

Since spelling correction in our work is based on the premise that words from manuscripts shall be mapped to dictionary words, a suitable dictionary is essential. The dictionary must be in a format that can be converted to SQL since the mapping comparison needs to be done in SQL. *Wörterbuchnetz* [Bra07] would provide the best dictionary for this purpose, however it is copyrighted and we were unable to gain permission to use it. Currently CAMM uses an open-source dictionary, *KTranslator* [Fer07] that provides its content in a tab-delimited format. An auxiliary program was written to convert tab-delimited text to SQL. 81542 distinct grapheme sequences were extracted from that dictionary. Single and double Metaphone results were computed for each word. A dictionary provides a fairly complete set of words, but it does not provide all morphological forms of the words. The already mentioned Europarl corpus [Koe05] contains almost 40 Million words in 348936 distinct grapheme sequences, thus providing a better coverage of the different morphological forms.

3.5 Mapping of Spelling Variations

Grapheme sequences (words) from manuscripts are mapped to grapheme sequences in the dictionary based on their score. The score is based on a number of factors. Phonemes are used to filter possibilities. It is not computationally feasible to compare 242000 words with 82000 dictionary words and to perform complex statistical operations on every combination. Therefore the comparisons are limited to phoneme identities of the graphemes. Grapheme frequency, neighborhood analysis, tri-word frequency, and profile scores are used to score the grapheme combinations. If the score of the best result is not better by at least one order of magnitudes the top-scoring graphemes are returned. Suppose there is an alternate grapheme sequence, then the following steps are carried out to map it to a grapheme from a dictionary:
(1) Select all words from the dictionary with the same phoneme sequence,

(2) Compute the string similarities for each grapheme sequence,
(3) Select the grapheme sequence with maximum similarity,
(4) If several sequences are selected compute the smallest Levenshtein distance,
(5) Add the word and neighborhood profile,
(6) Compute the score,
(7) Iterate steps 2→6 until every grapheme sequence is accounted for,
(8) Print all results until there is a score difference of one order of magnitude.

3.6 Module Structure

CAMM is a computational analysis tool-kit with a convenient user interface. The interface allows users to select data sets from a repository of data sources and then to apply different algorithms to the chosen data set. The results of the investigation are stored in a database. The results of the finding are then rendered by a Web API and made available online. Figure 2 illustrates the top-level layout of the software system. CAMM can handle XML and symbol-separated files such as tab-delimited or CSV files. Results of the findings are not presented in a static format to the users. Users can interact with CAMM to select a dictionary or corpus, a Metaphone algorithm, and manuscript texts. For each set of choices users are provided with detailed lexical and statistical analyses. CAMM provides two different functionalities from software design perspective, namely: computation and presentation. The one is to analyze, the other one is to store the results and to make them available to the research community for further information retrieval. For a computational analysis, CAMM creates new tables for every experiment. The parameters must be provided in the command line by the user conducting an experiment. Once a computational experiment is complete and the generated data is stored in the database, the user can choose to export the data to existing tables used by the modules shown in Figure 3. This operation makes the data available for querying and online viewing.

4 User Interface of CAMM

Figure 4 shows the home page of CAMM. Manuscripts are organized into collections. Only those collections, in which text versions of the content (a.k.a. 'tenor') are available, are displayed in CAMM. This page provides easy access to the collections. As mentioned above, CAMM allows users to try out different computational analyses and view statistical and lexical data on the analysis and the manuscripts. Users are required to choose a manuscript, a Metaphone algorithm, manuscript text, and a dictionary or corpus. Default selections are provided as shown in Figure 5. The same figure also shows different options for user to start the process of annotation. The user can select any of the algorithms such as: metaphone, double metaphone and cologne. The user can select types of

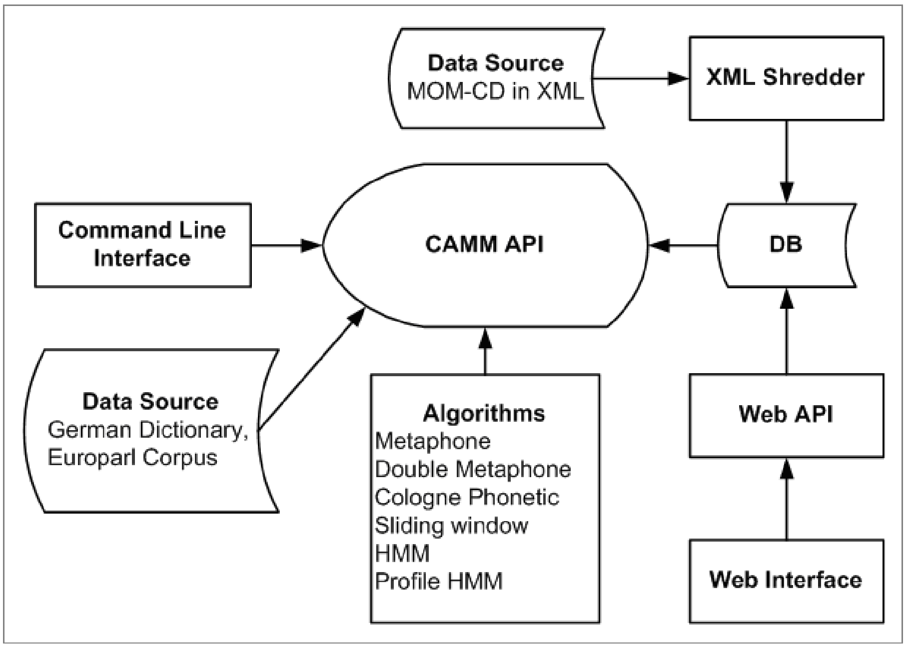

Figure 2: *Module Structure of CAMM*

texts from the options: original and corrected with diphtong. Furthermore, the user can select a dictionary from options: German dictionary and Europarl dictionary. The tool has the default selection of these options that is metaphone, original text, and Europarl corpus, and user has the choice to select different combinations to see the results produced by the tool.

Once a selection has been made, the user is provided with the title, abstract, manuscript text, corrected manuscript text, annotation form and some statistics about the analysis, as depicted in Figures 6-12. Figure 6 displays the selected manuscript in which each word has a link. The user can click on any linked word to see the statistical analysis of the word. Figure 7 shows a detailed statistical analysis of a selected word, 'haben'. The occurrence, the metaphone, and the double metaphone of this word are listed. Other listed dictionary candidates that also include Europarl corpus candidates help the user to find the very close spellings of the chosen word for annotation. In some circumstances, when CAMM is not able to reliably map spellings or to find any suitable mapping candidate, the unmapped words are colored red as shown in Figure 8.

On the screen shown in Figure 9 the user has an option to annotate the words after entering a master key. The user can annotate a whole manuscript or can make partial annotations, word by word. In Figure 9 the user has option to annotate the whole manuscript, whereas in Figure 10 the user has option

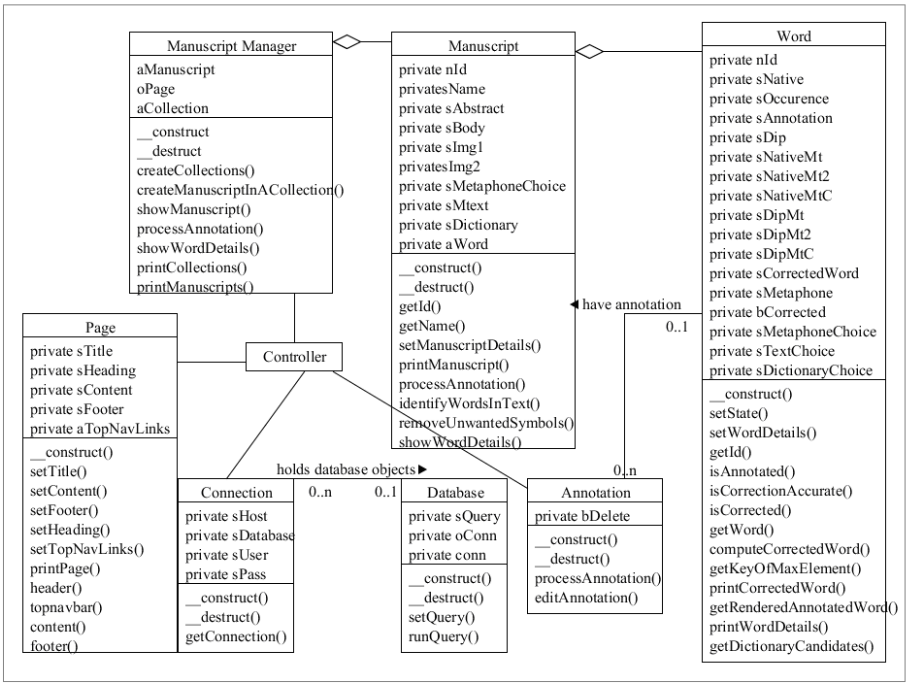

Figure 3: *Class Diagram of CAMM*

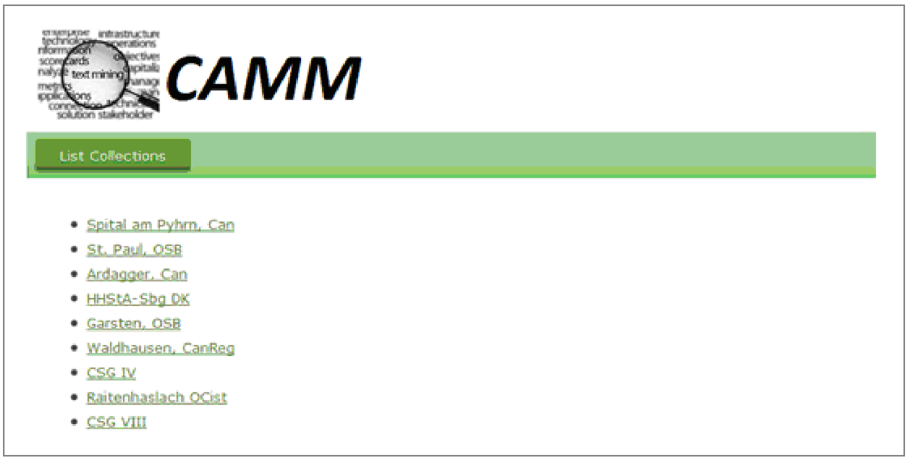

Figure 4: *Listing of Manuscript Collections*

to annotate word by word. Only an authorized user (who has the key) will be able to annotate the word(s) with further options. The user can correct the annotated word in case when the word is not annotated correctly at first place, where the user can use the delete option and re-annotate that word. The result

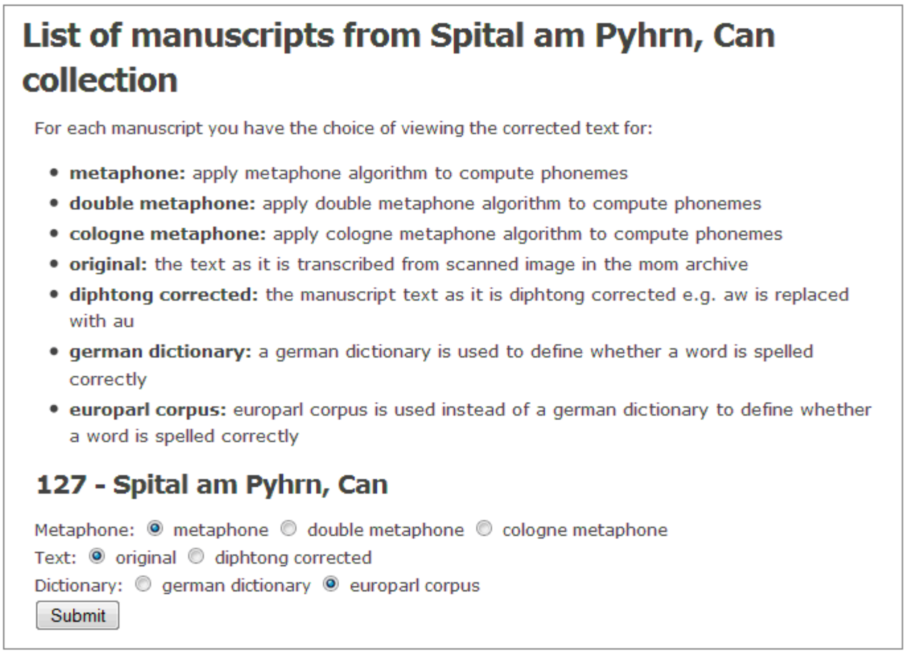

Figure 5: *Manuscript List from the '127 Spital am Pyhrn Can' Collection*

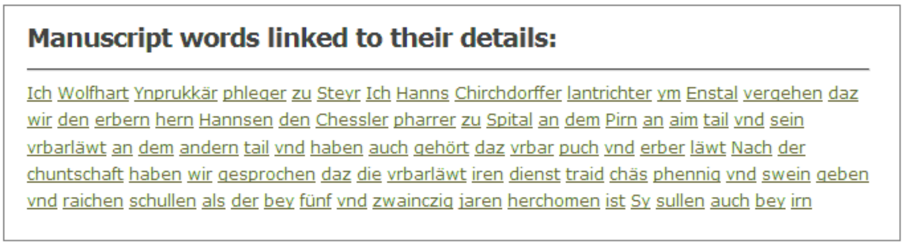

Figure 6: *Links to lexical and statistical Analysis Results per Word*

of the selected options will be as shown in the Figure 11. Word coverage only shows how many words the algorithm attempted to correct. The result of the selected algorithm to compute the phonemes by using the dictionary shows the words spelled correctly and incorrectly. Figure 12 shows a snippet of a scanned manuscript from the middle ages.

5 Assessment of Mapping Accuracy

We have selected the two manuscripts named '469 Spital am Pyhrm Can' and '127 Spital am Pyhrn Can' from the list of available manuscripts, to observe the results of words corrected accurately, along with the words corrected inaccurately. The user has seven unique options, but must select three simultaneously

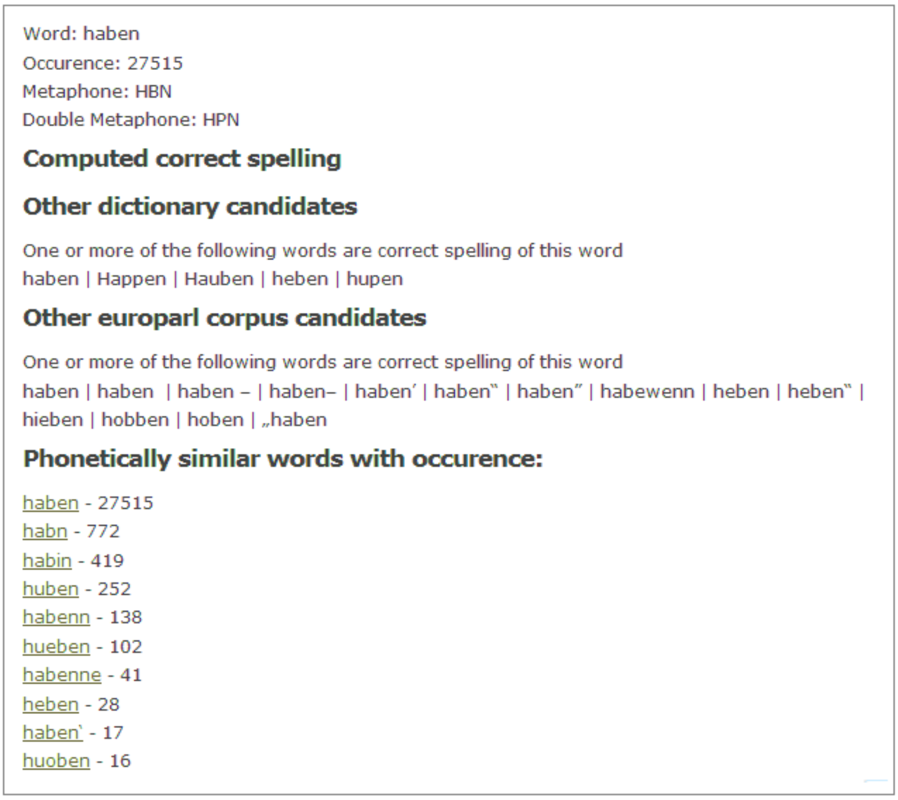

Figure 7: *Lexical and Statistical Analysis of the Grapheme Sequence*

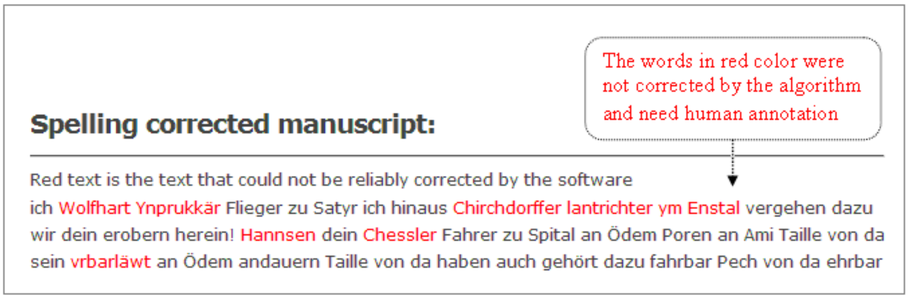

Figure 8: *Spelling partially corrected by the chosen Algorithm*

to view the output results. As shown by the twelve rows of the table in Figure 13, the user must run the sequence twelve times to confirm the results.

In the table of Figure 13 the sum of words corrected accurately and words corrected inaccurately is not equal to the word coverage because correction is based on the words annotated, which should be slightly different from word

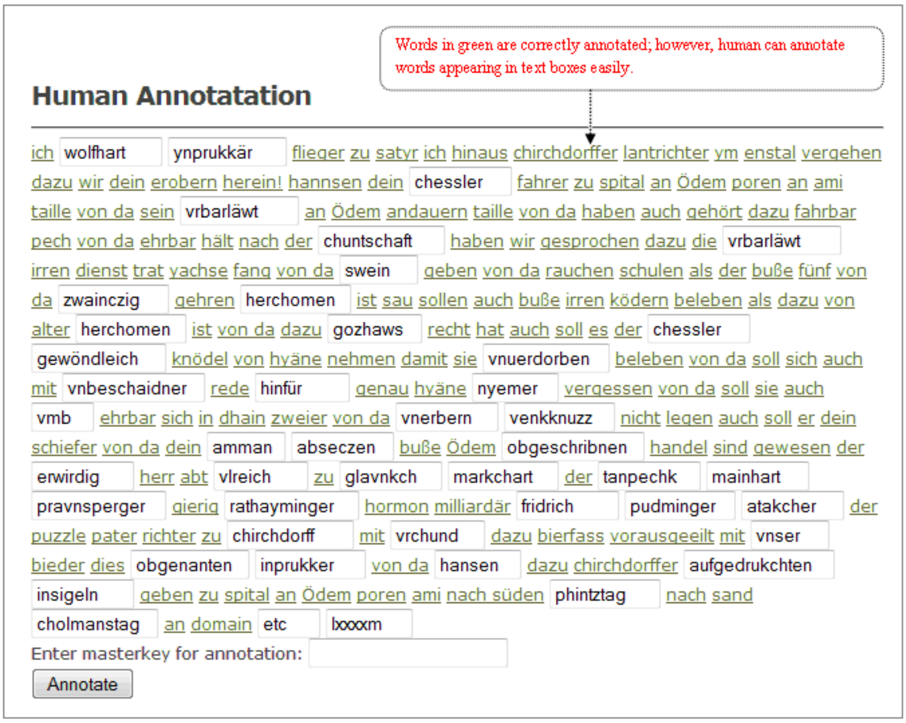

Figure 9: *Annotation Window for the Expert (Historian)*

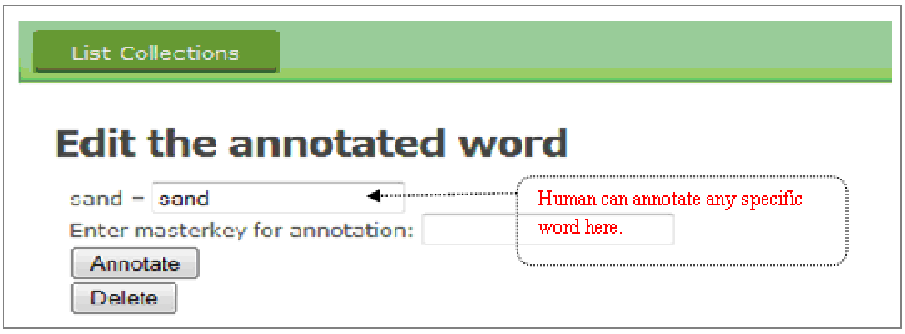

Figure 10: *Input Mask for Word Annotations*

coverage. For example: With *m.o.p* the word coverage is 195, and the sum of words corrected accurately and words corrected inaccurately is 156. So there are 39 words which were not annotated during that process. The calculation behind that table is based on the total number of original text words, here 248. The legends for rows in tables and X-Axis for the subsequent graphs are as follows:

— m.o.g: Metaphone algorithm, original text, German dictionary

— m.o.e: Metaphone algorithm, original text, Europarl corpus

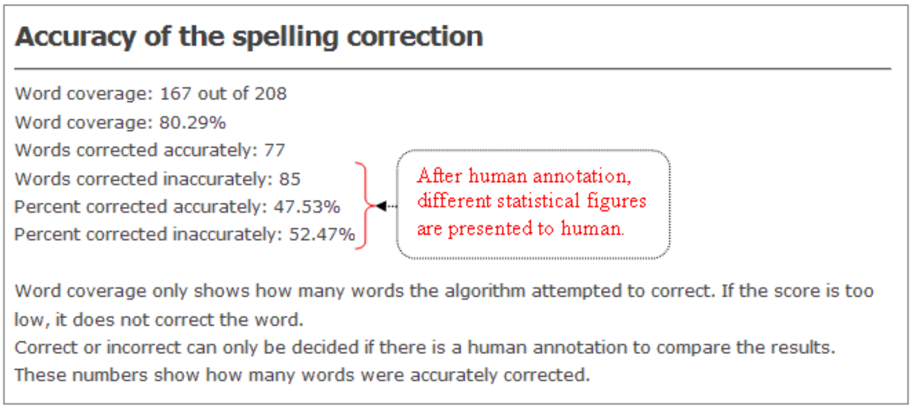

Figure 11: *Result Display with Metaphone and German Dictionary*

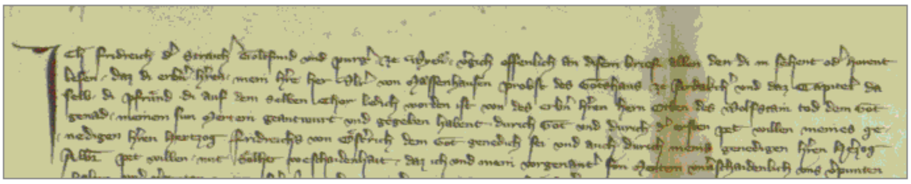

Figure 12: *Scanned Original from Collection '5466 Ardagger Can'*

— m.d.g: Metaphone algorithm, diphtong corrected text, German dictionary
— m.d.e: Metaphone algorithm, diphtong corrected text, Europarl corpus
— d.o.g: double Metaphone algorithm, original text, German dictionary
— d.o.e: double Metaphone algorithm, original text, Europarl corpus
— d.d.g: double Metaphone algorithm, diphtong corrected text, German dict.
— d.d.e: double Metaphone algorithm, diphtong corrected text, Europarl corp.
— c.o.g: Cologne Metaphone, original text, German dictionary
— c.o.e: Cologne Metaphone, original text, Europarl corpus
— c.d.g: Cologne Metaphone, diphtong corrected text, German dictionary
— c.d.e: Cologne Metaphone, diphtong corrected text, Europarl corpus

The Cologne phonetic algorithm, shown in the graph of Figure 14, is a suitable selection wherein a 90% percent or higher word coverage dictionary is consulted. It does however, produce adequate results with regards to validity of revision since the correction accuracy is nearly 4% percent less than *m.o.g.* Metaphone, utilizing a German dictionary seems to have a higher effectiveness than the other two options (double Metaphone and Cologne Phonetic).

Amongst others we also experimented with another manuscript identified as '127 Spital am Pyhrn Can'. The table shown in Figure 15 outlines the output in terms of word(s) corrected accurately, word(s) corrected inaccurately, as well as word coverage. We observed that the Europarl corpus and the Cologne Meta-

	Word coverage	Word coverage %	Words corrected accurately	Words corrected inaccurately	Corrected accurately %	Corrected inaccurately %
m.o.g	195	79	78	78	50	50
m.o.e	209	84	91	65	58	42
m.d.g	196	79	78	78	50	50
m.d.e	210	85	91	65	58	42
d.o.g	213	86	76	80	49	51
d.o.e	222	90	91	65	58	42
d.d.g	213	86	76	80	49	51
d.d.e	222	90	91	65	58	42
c.o.g	230	93	72	84	46	54
c.o.e	235	95	91	65	58	42
c.d.g	230	93	72	84	46	54
c.d.e	234	95	91	65	58	42

Figure 13: *Words Statistics for Manuscript '469 Spital am Pyhrm Can'*

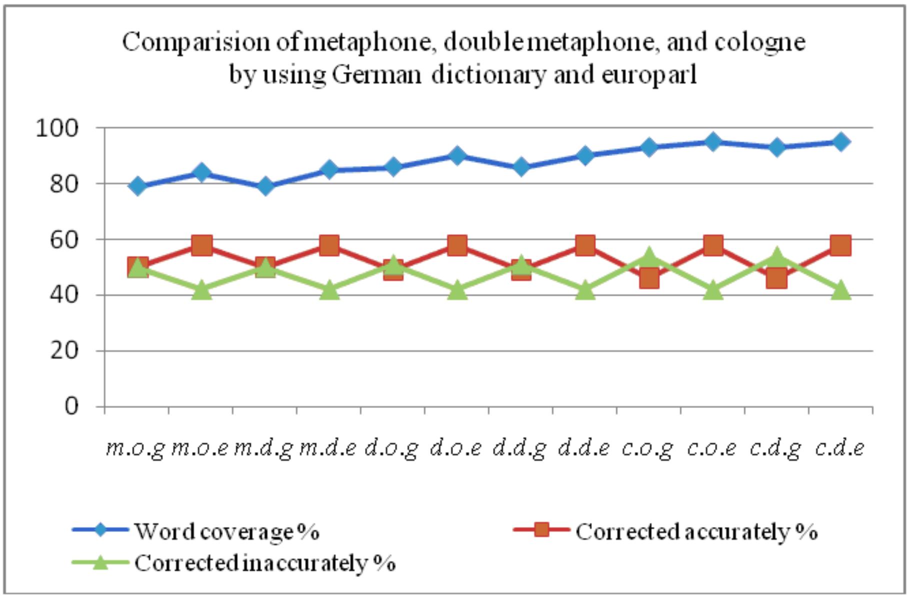

Figure 14: *Words Statistics for Manuscript '469 Spital am Pyhrm Can'*

	Word coverage %	Corrected accurately %	Corrected inaccurately %
m.o.g	75	51	49
m.o.e	80	62	38
m.d.g	75	50	50
m.d.e	80	61	39
d.o.g	85	48	52
d.o.e	89	62	38
d.d.g	85	50	50
d.d.e	89	62	38
c.o.g	90	48	52
c.o.e	92	62	38
c.d.g	91	47	53
c.d.e	94	61	39

Figure 15: *Words Statistics for Manuscript '127 Spital am Pyhrn Can'*

phone on the diphthong algorithm had the highest word coverage. Likewise, an improvement is noted in the corrected inaccurately word(s), at 61% in word(s) corrected accurately; at 39% an improvement in corrected inaccurately word(s). The graph of Figure 16 portrays that *c.d.e*, *c.o.e*, *d.d.e*, *d.o.e*, and *m.o.e* are producing about 60% word(s) accurately corrected.

6 Conclusion and Outlook to Future Work

Digitalization of historical texts is essential to save existing volumes of historical text from ruin and to preserve it for progeny. Digitalization should be conducted in a 'smart' way such that it enables further and more subtle information extraction. However, even when digitalization is conducted properly, computational analysis and information extraction is obstructed by the spelling variations problem. The automated normalization of medieval spelling variations is a rather new sub-field of History-Informatics wherein only little research has been conducted so far.

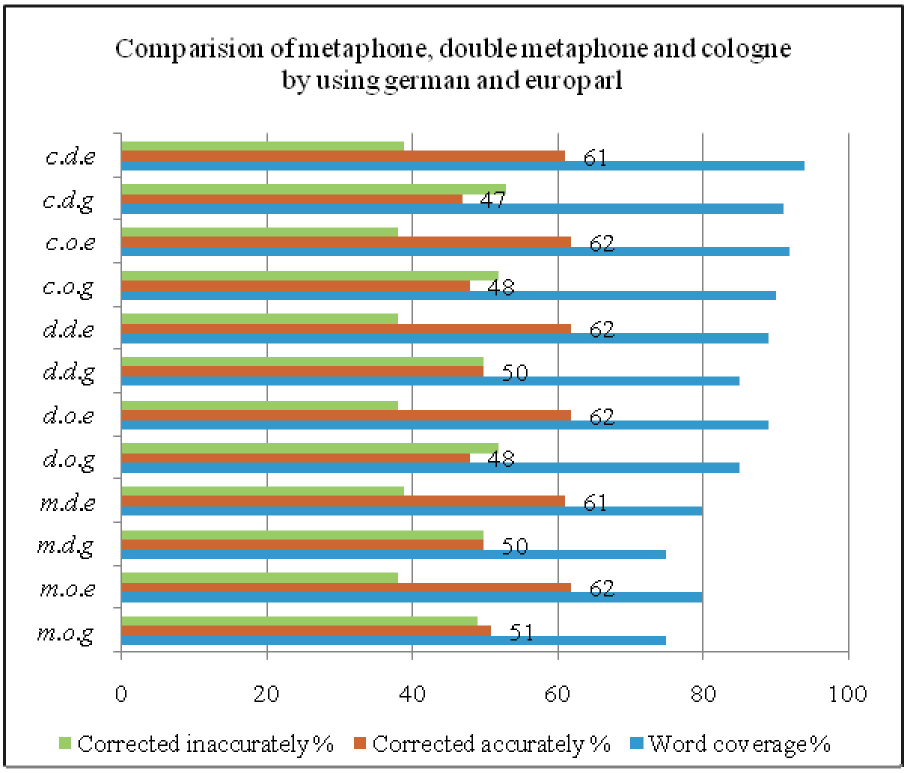

Figure 16: *Words Statistics for Manuscript '127 Spital am Pyhrn Can'*

CAMM is, as far as we know, the first tool available to provide support in this regard. In preliminary earlier work [ARG11] we had developed the Phonetic Orthography Mapper (POM) with phonetic analysis and machine learning techniques. Our current CAMM system, as described in this paper, extends our previous work in several ways: a number of new algorithms were implemented and evaluated, a more comprehensive online analysis tool kit was developed, several dictionaries were incorporated, etc.

In [HH+07] eight categories of the spelling variation problems were discussed. The new CAMM tool tackles two of those eight problem categories, namely graphemic-phonetic variations, and new-character problem. Moreover, CAMM also allows scholars of medieval history to annotate manuscripts through a user-friendly web interface, according to the practical needs that had been identified in [BVG08]. Currently CAMM allows users to experiment with different dictionaries, text editing, and Metaphone algorithms. A learn-algorithm adapts itself to the user's choices. For every experiment the user is provided with detailed word-by-word lexical and statistical analysis results. CAMM currently provides access to 11275 manuscripts organized into 54 collections with a total of 242446 distinctly spelled words. In its current version, CAMM accurately corrects spelling of approximately 55% percent of the verifiable words.

From a technical perspective, the CAMM tool kit is characterized by a modular and thus extensible software architecture. Further dictionaries, algorithms, statistics packages, and other features can be easily added without compromising the existing structure. The performance and accuracy of CAMM is thus expected to improve over time. To date the most critical limitation in the tool is the scarcity of human annotations in the documents to be processed. CAMM normalizations are partly based on POM, which is based on a learning algorithm. As the quantity and quality of human-generated annotation in the input documents increases, CAMM would also yield better normalization results.

Acknowledgements

Thanks to *Georg Vogeler* for his valuable suggestions about the algorithms. Thanks also to *Jochen Graf* and the Monasterium consortium for having given us access to the medieval dataset and for sharing valuable information about the existing EditMOM tools. Thanks to the Athabasca University, for providing a server to launch this tool, and thanks to the Web Unit of the Computing Services Department at Athabasca for keeping the link alive.

Definitions

Corpus: Collection of linguistic data compiled from written or transcribed text
Grapheme: Sequence of lexical symbols to represent a phoneme
Orthography: Set of lexical norms for spelling words consistently
Phoneme: Distinct unit of sound in a natural language
Word: Sequence of graphemes to represent a phoneme or a phoneme sequence

References

[ARG11] Mushtaq Ahmad, Nazim Rahman, Stefan Gruner: "A Phonetic Approach to Handling Spelling Variations in Medieval Documents", Proceedings SAICSIT'11 Annual Conf. of the South Afr. Inst. of Comp. Sc. and Inf. Techn., Cape Town, South Africa, pp. 263-267, 2011.

[Atk11] Kevin Atkinson: "GNU Aspell", 2011, http://aspell.net/

[BRa07] Thomas Burch, Andrea Rapp: "Das Wörterbuch-Netz: Verfahren, Methoden, Perspektiven", Post-Proceedings .hist'2006, Historisches Forum 10/1, pp. 607-627, 2007.

[BVG08] Benjamin Burkard, Georg Vogeler, Stefan Gruner: "Informatics for Historians: Tools for Medieval Document XML Mark-Up, and their Impact on the History Sciences", Journal of Universal Computer Science 14/2, pp. 193-201, 2008.

[EGF06] Andrea Ernst-Gerlach, Norbert Fuhr: "Generating Search Term Variants for Text Collections with Historic Spellings", Proceedings ECIR 28th European Conf. on Inf. Retrieval Research, pp. 49-60, 2006.

[Fer07] Raul Fernandes: "KTranslator", 2007, http://ktranslator.sourceforge.net/

[HH+07] Andreas Hauser, Markus Heller, Elisabeth Leiss, Klaus U. Schulz, Christiane Wanzeck: "Information Access to Historical Documents from the Early New High German Period", Proceedings Dagstuhl Seminars #06491, Schloss Dagstuhl, Germany, 2007.

[Hei10] Karl Heinz: "Monasterium.net: Auf dem Weg zu einem europäischen Urkundenportal". Proceedings 12th Commission Internationale de Diplomatique, pp. 139-145, 2010.

[JL+02] Haifeng Jiang, Hongjun Lu, Wei Wang, Jeffrey Xu Yu: "XParent: An Efficient RDBMS-Based XML Database System", Proceedings 18th Internat. Conf. on Data Eng., San Jose, USA, pp. 335-336, 2002.

[Kes04] Brett Kessler: "A Spelling Corrector incorporating Knowledge of English Orthography and Pronunciation", 2004, http://spell.psychology.wustl.edu/correct/

[Knu73] Donald Knuth: "The Art of Computer Programming" (Vol. 3), Addison-Wesley, 1973.

[Koe05] Philipp Koehn: "Europarl: A Parallel Corpus for Statistical Machine Translation", Techn. Rep., 2005, http://www.statmt.org/europarl/

[Kra09] Adelheid Krah: "Monasterium.net: Das virtuelle Urkundenarchiv Europas", Archivalische Zeitschrift 91 (Sonderdruck), pp. 221-246, 2009.

[LNg04] Hoi Kit Lau, Vincent Ng: "INode: an effective Approach for storing XML using Relational Database", Internat. Journ. of Web Eng. and Techn. 1/3, pp. 338-352, 2004.

[Lev66] Vladimir Levenshtein: "Binary Codes capable of correcting Deletions, Insertions and Reversals", Soviet Physics Doklady 10/8, pp. 707-710, 1966.

[MLC08] Jun-Ki Min, Chun-Hee Lee, Chin-Wan Chun: "XTRON: An XML Data Management System using Relational Databases", Inf. and Softw. Techn. 50/5, pp. 462-479, 2008.

[Mok06] Gary Mokotoff: "Soundexing & Genealogy", 2006, http://www.avotaynu.com/soundex.htm

[Phi90] Lawrence Philips "Metaphone Algorithm" 1990, http://aspell.net/metaphone/

[PE+08] Thomas Pilz, Andrea Ernst-Gerlach, Sebastian Kempken, Paul Rayson, Dawn Archer: "The Identification of Spelling Variants in English and German Historical Texts: Manual or Automatic?", Lit. Linguist Comp. 23/1, pp. 65-72, 2008.

[PL+06] Thomas Pilz, Wolfram Luther, Norbert Fuhr, Ulrich Ammon: "Rule-based Search in Text Databases with Nonstandard Orthography", Lit. Linguist Comp. 21/2, pp. 179-186, 2006.

[PZa84] Joseph J. Pollock, Antonio Zamora: "Automatic Spelling Correction in Scientific and Scholarly Text", Comm. of the ACM 27/4, pp. 358-368, 1984.

[Pos69] Hans Joachim Postel: "Die Klner Phonetik: Ein Verfahren zur Identifizierung von Personennamen auf der Grundlage der Gestaltanalyse", IBM-Nachrichten 19, pp. 925-931, 1969.

[QZ+05] Jie Qin, Shu-Mei Zhao, Shu-Qiang Yang, Wen-Hua Dou: "XPEV: A Storage Model for Well-formed XML Documents", LNCS 3613, pp. 360-369, 2005.

[MLC08] Jun-Ki Min, Chun-Hee Lee, Chin-Wan Chun: "XTRON: An XML Data Management System using Relational Databases", Inf. and Softw. Techn. 50/5, pp. 462-479, 2008.

[YA+01] Masatoshi Yoshikawa, Toshiyuki Amagasa, Takeyuki Shimura, Shunsuke Uemura: "XRel: A Path-Based Approach to Storage and Retrieval of XML Documents using Relational Databases", ACM Trans. on Internet Techn. 1/1, pp. 110-141, 2001.

[ZHS10] Hasan Zafari, Keramat Hasani, M. Ebrahim Shiri: "XLight: An Efficient Relational Schema to Store and Query XML Data", Proceedings DSDE Internat. Conf. on Data Storage and Data Engineering, Bangalore, India, pp. 254-257, 2010.