Corpus-Induced Corpus Clean-up

Reynaert, M.W.C.

Published in:
Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC-06)

Publication date:
2006

Document Version
Peer reviewed version

Corpus-Induced Corpus Clean-up

Martin Reynaert

Induction of Linguistic Knowledge
Tilburg University
The Netherlands
reynaert@uvt.nl

Abstract

We explore the feasibility of using only unsupervised means to identify non-words, i.e. typos, in a frequency list derived from a large corpus of Dutch and to distinguish between these non-words and real-words in the language. We call the system we built and evaluate in this paper CICCL, which stands for 'Corpus-Induced Corpus Clean-up'. The algorithm on which CICCL is primarily based is the anagram-key hashing algorithm introduced by (Reynaert, 2004). The core correction mechanism is a simple and effective method which translates the actual characters which make up a word into a large natural number in such a way that all the anagrams, i.e. all the words composed of precisely the same subset of characters, are allocated the same natural number. In effect, this constitutes a novel approximate string matching algorithm for indexed text search. This is because by simple addition, subtraction or a combination of both, all variants within reach of the range of numerical values defined in the alphabet are retrieved by iterating over the alphabet. CICCL's input consists primarily of corpus derived frequency lists, from which it derives valuable morphological and compounding information by performing frequency counts over the substrings of the words. These counts are then used to perform decompounding, as well as for distinguishing between most likely correctly spelled words and typos.

1. Introduction

Visual inspection of a subset of the word unigram frequency list derived from the Reuters Corpus Volume 1 or RCV1 (Lewis et al., 2004) (initial character lowercased word types only) has taught us that over 21% of the word types in the list are in fact typos, i.e. word types orthographically unacceptable by any convention in the English language community. We marked 33,000 typos in a word type list of about 150,000 items. We have performed a comparable study on a subset of the unigram frequency list of the Dutch Twente Corpus, covering the year 2002, and identified over 13,000 typos.

On the basis of this material we conduct a study of unsupervised ways to identify these unacceptable word forms on the basis of the corpora themselves. To this end we employ the core spelling correction strategy as described in (Reynaert, 2004). Text-Induced Spelling Correction or TISC has been shown to be a viable alternative to existing approaches in (Reynaert, 2005), for both languages Dutch and English. Performance in terms of recall was shown to be comparable to the state-of-the-art systems available today, but levels of precision exceed those of the other systems evaluated by an order of magnitude.

As part of the PhD-dissertation, we described Corpus Induced Corpus Clean-up or CICCL. An English-specific module was built based on the core correction mechanism employed in TISC which is meant to clean up the lexicons used by TISC. In contrast to most spelling correction systems, the lexicon employed by TISC is not a 'trusted' dictionary, but contains noise in the form of recurrent typos found in any word type list derived from a large corpus of text. CICCL starts off with the most frequent words in the frequency list and systematically searches for typographical variants within the whole list. This it does within certain bounds, it can be specified at run-time how many edits are allowed to get from the input word to the variants retrieved.

In the rest of this paper we describe a new and simple version of CICCL and evaluate it on Dutch.

2. Unsupervised identification of non-words in a corpus

The goal of this paper is to show that it is feasible to identify, in order to subsequently remove, the majority of orthographically unacceptable word forms present in a large corpus of written text in a particular language using unsupervised means only. The resources used to do this are corpora of raw text in the particular language only: no use is made of pre-existing resources such as validated word lists or dictionaries. The assumption is that given a sufficiently large corpus, all information necessary to perform the task is present in the corpus. In this paper we focus on the identification of non-words only. Though in the vast majority of cases these non-words will be retrieved by way of their correct form, we do not aim at linking up correct forms with their own misspellings only. As (Reynaert, 2005) has shown, doing that in most cases requires the use of context. So far, we here employ in-word context only, in the form of 'the other part' of a compound word. Neither do we aim at retrieving all possible non-words in the corpus. We naturally hope to identify the bulk of the non-words present and will assess to what extent we manage to do so, but we know that the current implementation does not address all types of possible non-words.

The motivation for this work is the finding that in a large collection of written text, e.g. for English the Reuters RCV1 corpus, up to one fifth of the word types beginning in a lowercased character are in fact non-words in English. These typos furthermore have a Zipfian distribution: some typos occur with high frequency, most occur but a few times and the majority only once. (Reynaert, 2005)

This work wishes to explore ways of providing an alternative to the established practice of hapaxing or employing

some form of frequency thresholding in language modeling, as we have shown that this removes far more correct words than undesirable word forms, leading to increased data sparseness, while still retaining the more frequent incorrect word forms, leading to less reliable corpus counts for the correct word forms.

3. The task

We believe retrieving the typographical variants for a particular word is only a first step. Many ways of doing so are available, an expert overview of these is in (Navarro, 2001). In the following we aim to show that a lot more factors influence the outcome of the endeavour.

In the following, we intend to retrieve those word forms that lie within the bounds we set for word forms occurring in the list we wish to examine with a frequency above that of the average frequency for a word form of the particular length in characters observed within a far larger background frequency list. Of those retrieved, we will say they are **variants** of the word string we focus on: the **focus word**.

The bounds set will be expressed in terms of the Levenshtein distance or LD (Levenshtein, 1965).

For all the variants retrieved, the task we address is determining whether the variant is in fact a perfectly acceptable word in the language in its own right, whether or not this is a perfectly acceptable morphological variant, a perfectly acceptable orthographical variant – perhaps to another portion of the language community, viz. English versus American usage – or whether the word variant retrieved constitutes a word form unacceptable to any sizeable portion of the language community. If the latter is the case, we will call the word variant a **non-word** in that particular language, or **typo** for short.

4. The means

4.1. Anagram-based Spelling Correction

We employ the core-correction mechanism first introduced in (Reynaert, 2004) and described in more depth in (Reynaert, 2005) which first uses bad hashing to identify all word strings in the corpus at hand that consist of the same subset of characters and assigns a large natural number to them, to be used as an index. It then uses the index values derived in the same way for the alphabet used, which can be single characters or combinations of two or more characters, to perform simple addition, subtraction or addition and subtraction in combination to retrieve typographical near-neighbours for the word string under consideration from the hash containing the word type list for the corpus.

For each word type in the word list under examination, we obtain a numerical value, which will serve as its hash key. The formula represents the mathematical function used to do this, where f is a particular numerical value assigned to each character in the alphabet and c_1 to $c_{|w|}$ the actual characters in the input string w.

$$Key(w) = \sum_{i=1}^{|w|} f(c_i)^n$$

In practice, we use the ISO Latin-1 code value of each character in the string raised to a power n, where n is currently:

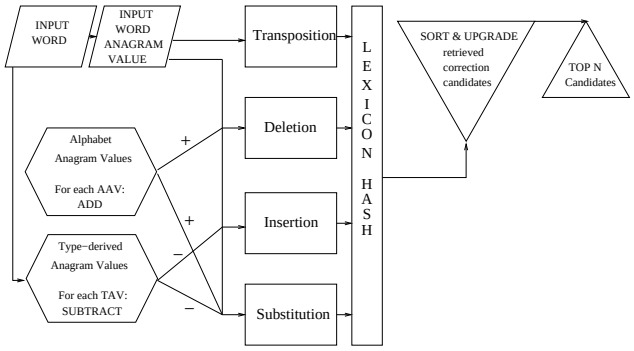

Figure 1: The core-correction mechanism. The lexicon built up from the input word list is queried for transpositions on the basis of the Input word Anagram Value (IAV) alone. Each AAV is added to the IAV to query for deletions. Each TAV is subtracted to query for insertions. Each AAV is added and each TAV subtracted to query for substitutions.

5. In effect, all anagrams, words consisting of a particular set of characters and present in the list, will be identified through their common numerical value. As the collisions produced by this function identify anagrams, we refer to this as an **anagram hash** and to the numerical values obtained as the **anagram values**, further abbreviated as AVs, and **anagram keys**, when we discuss these in relation to the hash.

Based on a word form's anagram key it thus becomes possible to systematically query the list for any variants present, be they morphological, typographical or orthographical.

The list of anagram values for the character(s) collected from the input type we further refer to as the **Type-derived Anagram Values**, abbreviated: TAVs. Given e.g. the type *lolita* we collect the AVs for the single characters. Then we add a space front and back (the space is represented as an underscore, here): *_lolita_*, derive the AVs for the character bigrams: _l, lo, ol, li, it, ta, a_ and store these. So, in all, we derive n unigram values and n+1 bigram values. Given the number of characters $c_{|w|}$ in the string w we get $c_{|w|} + (c_{|w|} + 1)$.

The alphabets used in the present work contain AVs representing single characters for the tests limited to single character edits. For the tests performed when allowing up to two character edits, the alphabet contains the AVs for single characters and all possible two-character combinations. We further refer to the AVs in the alphabet as the AAVs.

Retrieval of typographical variants Figure 1 shows a schematic representation of the core-correction mechanism. The lexicon is a hash built up at run-time having the AVs as keys and chained anagrams as values. We use the AV for the focus word and the list of TAVs and the longer list of AAVs to query the lexicon for variants of the focus word. These variants can all be seen as variations of the usual error type taxonomy due to (Damerau, 1964):

transpositions These we get for free: they have the same anagram key value, so when queried for the input word AV, the lexicon returns the correct form and its anagrams (if any).

deletions We iterate over the alphabet and query the lexi-

con for the input word anagram value plus each AAV.

insertions We iterate over the TAVs and query the lexicon for the input word anagram value minus each TAV.

substitutions We iterate over both TAV and AAV lists adding each value from the AAVs and subtracting each value of the TAVs to the input word anagram value and repeatedly query the lexicon.

By systematically querying the lexicon hash we retrieve all possible variants that fall within reach. The actual reach is defined by the alphabet used.

Secondly, for each variant retrieved, we use a separate subroutine which calculates the LD between the focus word and the variant retrieved. This is required because even with the LD limit imposed by the alphabet, variants of great LD are retrieved, e.g. *goniometer* for *goverment, with an LD of 6. By discarding the variants retrieved that have an LD larger than the limit we set at run-time, less plausible variants are removed.

4.2. Corpus-Induced Corpus Clean-up: the algorithm

The resources made available to CICCL in its current version are: an alphabet, lists of corpus-derived word types and their associated raw corpus frequencies for Dutch, English and French and the Twente 2002 word types and associated corpus frequencies list, in original, unannotated format.

We here discuss the algorithm informally; we envisage an in-depth technical paper covering all the details at the end of the current project, when the final version is due to become available through the Dutch TST-Centrale[1].

First the program reads in and studies the information available in the background word frequency list of Dutch. Studying amounts to tallying frequency counts for all the substrings seen in the alphabetically sorted frequency list. The top *n*, where *n* is 200 in this work, are then stored and made available to the rest of the process as a list of corpus-derived prefixes and suffixes.

Next we handle each word in the list to be examined. For each word, if its frequency is higher than the average frequency for its length, and if the word has not been accepted as a variant before, we let the core-correction module retrieve all its variants within the limit set.

For each of these variants, if their LD does not exceed the limit set and if the frequency of the focus word in the background Dutch frequency list is greater than that of the variant and if the background frequency of the variant is greater than the average frequency for its word length, we discard them. For all the variants not discarded, if their frequency in the English or French background frequency lists are greater than in the Dutch, we discard them. If they are retained, we mark them as having been seen, to prevent them from being retrieved time and again as variants for other focus words.

On the basis of a split into two decompounding parts, which was effected when the list to be examined was studied, it is then decided whether to discard the variant on the basis of the fact that both compound parts' background corpus frequencies are higher than the average frequency for their particular word lengths in the background frequency list.

For all the variants retained still, we check whether the focus word has, instead of the split, an extra linking 's', or 'n' or dash. If they do, we retain these. We further check whether when we subtract the one from the other starting from the front of the word, whether the remaining 'suffix', if it is no longer than 4 characters, appears in the corpus-derived top 200 list of suffixes. If so, we discard the variant. All variants retained are output linked to the focus word which retrieved them.

[1] http://www.tst.inl.nl/

Corpus	Lang.	Mb	Tokens	Types
NYT	AE	5,570	1,106,376,695	1,863,802
R-RCV1	IE	714	134,031,130	1,626,038
ILK	D	1,748	314,051,047	2,747,341
TWC	D	2,014	365,545,491	2,607,305
TWC2	D	510	92,793,519	914,026
ROUL	F	273	52,722,253	422,682

Table 1: Corpora Statistics: Corpus, language (AE: American English, D: Dutch, F: French), size in Megabytes, number of word tokens, number of word types.

4.3. Corpora used

4.3.1. Background Corpora

We here briefly describe the corpora used.

Dutch For Dutch we used both the ILK Corpus[2] and the Twente Corpus[3] (TWC). The ILK Corpus is a collection of southern Dutch regional newspapers, expanded with 5 years of Dutch Roularta magazines. The Roularta magazines constitute a series of Belgian Dutch weekly magazines devoted to current affairs, industry, financial affairs and leisure. The TWC comprises a number of national Dutch newspapers, teletext subtitling and autocues of broadcast news shows and news data downloaded from the WWW.

English The American English corpus we used was the New York Times (1994-2002) material available in the LDC Gigaword Corpus (Graff, 2003). We further refer to this corpus as NYT.

French For French we used 8 years ('91-'98) of Roularta Magazines[4]. These constitute the Belgian French counterpart of the Flemish Roularta magazines.

From each of the background corpora, after tokenization by means of a rule-based tokenizer, a unigram frequency list was derived. The lists of word types with their raw corpora frequencies are then made available to CICCL, one list per language. We present statistics regarding corpora sizes and numbers of words in these corpora in Table 1.

4.3.2. Evaluation Corpus and test set

Evaluation Corpus We used the Twente Corpus 2002, TWC2, which represents national Dutch newspapers from

[2] http://ilk.uvt.nl/ilkcorpus/

[3] http://wwwhome.cs.utwente.nl/∼druid/TwNC/TwNC-main.html

[4] http://www.roularta.be/en/products/

the year 2002 as the evaluation corpus. We proofread part of the unigram frequency list derived from this corpus in order to derive evaluation material for CICCL. Details about the typos identified in this list are presented in Table 2. We limited ourselves to the unigram word types beginning in a lowercased character. For comparison purposes we also supply the statistics obtained from a similar list for English, obtained from the Reuters RCV1 Corpus. More in-depth information on the English non-words and on how to obtain these statistics on lists of pairs of correct words and typos is to be found in (Reynaert, 2005).

Large sections of the Twente 2002 frequency list were annotated for non-words, foreign words, missing diacritics and eye-dialect in an exhaustive fashion. The longest running stretch we annotated was from the word *i* up to the word *kroeg* (pub), running for 41,955 word types. This part we will here use as the evaluation set. Other parts of the list were visited in a more random fashion, just by dipping into and starting to annotate. When we encountered an error pattern that looked likely to be productive, we searched for it throughout the list and marked all its occurrences. Patterns like this would be successions of three identical characters, e.g. *zegggen for *zeggen* (to say), but would also include highly productive misspellings such as *commisie for *commissie* (commission).

In this way we annotated 13,151 items as being non-words. The greater part of these, 9,152 items, were provided with their correct form as dictated by the context they appeared in. The typo and hapax *krijgsen, for instance, might have to be resolved to either *krijgen* (to get, receive) or to *krijsen* (to screech), but on the basis of its context 'verhullen is niet het exclusieve voorrecht van krijgsen en villaheren' (source: Algemeen Dagblad 2002-06-29) (to conceal is not the exclusive right of war lords and manor lords) should be corrected as 'krijgs- en'.

Discussion of non-words in English and Dutch In their essence, the two tables of statistics on non-words in two non-trivial corpora – one English, the other Dutch – have a very similar story to tell. In both corpora deletions occur most often, followed by insertions. While we see that in English substitutions and transpositions are on a par, in Dutch we observe far less transpositions. We surmise that this is the result of more careful proofreading: the Dutch corpus consists mainly of published newspaper stories, the English exclusively of raw newswire stories. In Dutch we see some heavier manglings of words. Upon examination we found that these occur in the teletext subpart of the TWC2. The capturing of teletexts from the TV-signal is apparently subject to transmission errors due to synchronisation problems. This may produce character strings that are beyond spelling correction: one can only try to guess what the word may have been meant to be.

What we think is more important, is that both tables show that typos have a Zipfian distribution (Zipf, 1935). Not only do we see that the smaller phenomena, i.e. typos having LD 1, occur very frequently and the larger phenomena, i.e. typos having LD 4 and more) very rarely. We also see that there seems to be a power law governing these occurrences, which is particularly finely drawn for the top three typo categories: deletion, insertion and substitution. This means that if you see 1,000 LD 1 typos, you will likely see about 100 LD 2 typos and about 10 LD 3 typos.

If one were able to fully automatically identify and replace the typos of only LD 1 and 2, one would remove 97,58% of the typos in the Dutch text and 98,69% of the typos in the English corpus. In the evaluations we will see how well we fare on these two categories and what the consequences are of trying to recover typos of greater LD.

Run-ons and split words The vast majority of run-ons involve concatenations of words with highly frequent short words such as articles, prepositions and verbs. Split words, on the other hand, seem to involve primarily longer words and are in this corpus no doubt the result of infelicitous preprocessing of the texts. The texts as delivered by the various newspaper publishers come in various formats and are subsequently converted to a uniform format. Things clearly go wrong in this process, among other things the handling of hyphenation. In some cases, hyphens seem to be converted to spaces, resulting in elevated numbers of split words, the splits sometimes running up to 4 or more for a single word, resulting in scores of loose 'syllables' in the frequency list. These then typically have no single resolution. We found splits of *afschuwelijk* (hideous) into 'af' (ready, off) 'schuwe' (shy) and 'lijk' (corpse), *mogelijkheden* into 'moge' (may, as in: 'May this never happen!'), 'lijk' (corpse) en 'heden' (today). The translations show that these splits may very well, though not necessarily, result in existing words, thereby constituting a great source of confusables. Their frequency counts are of course tallied with those of the valid uses of the words. They furthermore affect the present study in two ways: first, they clutter the frequency list with short word strings, which confound the search for proper misspellings of valid short words. Second, in the approach taken here, they will be linked to the wrong correct word. The recurrent split '*gesigna leerd' provides two Dutch non-words. The first does not seem to fall within LD 1 of any valid word form. The second will most likely be linked to the word *leed* (noun: sorrow or verb as in 'zij leed' (she suffered)).

While clearly important problems to be addressed eventually, we do not here attempt to resolve run-ons and split words. As it stands, we actually think that these two types should be the first to be addressed if one were to perform a complete and fully automatic clean-up of a large corpus.

Compound-confusables Another fertile source of confusables was found to lie in compounding. In Dutch, in contrast to English where most often compounds are written as separate words, compounds are written as single words, as in German. A single insertion or deletion may then very well affect the meaning of the compound, turning it in a different word altogether. This is a very productive phenomenon, in cases leading to hilarious results. Good examples here are 'vakbonen' (trade beans) for *vakbonden* (trade unions) and medewekers (co-soakers) for *medewerkers* (co-workers). The fact that compounds in Dutch can be formed at will as required or desired, implies that no dictionary of Dutch can ever be complete. This simple but important observation is one of the motivations for us to want derive the lexicons we use from corpora, so as to at least

Category	LD 1	LD 2	LD 3	LD 4	LD 5	LD 6	Total	%
deletion	3,288	331	30	24			3,673	40.13
insertion	2,441	198	33	8			2,680	29.28
substitution	964	81	8				1,053	11.51
transposition		440		2			442	4.83
multiple		240	74	18	5	2	339	3.70
space deletion	785						785	8.58
multisingle	76	66	14	3	1		160	1.75
capitalisation	16	1					17	0.19
dash to space	3						3	0.03
total	7,573	1,357	159	55	6	2	9,152	
%	82.75	14.83	1.74	0.60	0.07	0.02		100

Table 2: Dutch: TWC2: Statistics of the error categories in the 9,152 typo/correction list.

Category	LD 1	LD 2	LD 3	LD 4	LD 5	LD 6	Total	%
deletion	4,026	239	9	2			4,276	35.36
insertion	3,370	196	17	2			3,585	29.64
transposition		1,566		5			1,571	12.99
substitution	1,447	91	4	1	1		1,544	12.77
multiple		398	89	11			498	4.12
run-on	314						314	2.60
capitalization	168	2		1			171	1.41
multisingle		52	15	2			69	0.57
split word	51						51	0.42
dash to space	15						15	0.12
total	9,391	2,544	134	24	1	0	12,094	
%	77.65	21.04	1.11	0.20	0.01	0		100

Table 3: English: Reuters RCV 1: Statistics of the error categories in the 12,094 typo/correction list.

have a more informed idea of what is in fact being produced and productive in the language. Compound confusables cannot be detected on orthographical grounds alone: their composing parts are valid words and may very well be highly productive and frequent. They can be detected on the basis of orthographical neighbourhood to a probably far more frequently occurring compound. We found that all the compound confusables we identified are at LD 1 from their intended compounds. What is furthermore the case is that what at first sight appears to be a compound confusable, may very well been intended, perhaps as a pun, by the author. We came across a great many of these, it turned out the corpus contains a lengthy article on the Dutch translation of James Joyce's 'Finnegan's wake', containing lengthy passages of both original and translated versions. So we came across 'hinderdaad' (literally: hinder deed, in contrast to *inderdaad* (indeed)) for Joyce's original 'as a marrer of fact'. Further we find 'zonderzoeker' (unsearcher) for *onderzoeker* (researcher) in this article, but also 'wonderzoeker' (wonder searcher) in an altogether different article.

It is one thing to retrieve typographical near-neighbours, it is quite another thing to decide, using nothing but unsupervised techniques and in the absence of validated word-lists, whether the near-neighbours retrieved are in fact other real-words or typos. In the next section we examine how well CICCL manages both tasks.

5. Evaluation

5.1. How we evaluate

We use as the evaluation set the section of the TWC2 frequency list stretching between the words *i* and *kroeg*. The section comprises 41,948 word types, 2,160 of which constitute typos.

We evaluate in terms of recall and precision, resulting in the combined F-score (van Rijsbergen, 1975). These metrics are derived from the numbers of True Positive or TPs, False Positives or FPs and false negatives or FNs returned by the system.

- Recall = $\boldsymbol{R} = \frac{TP}{TP+FN}$
- Precision = $\boldsymbol{P} = \frac{TP}{TP+FP}$

The harmonic mean of $\boldsymbol{R}$ and $\boldsymbol{P}$, the simplified F measure, $\boldsymbol{F}$, is given by:

- $\boldsymbol{F} = \frac{\boldsymbol{2PR}}{\boldsymbol{R}+\boldsymbol{P}}$

On the basis of the evaluation set we determine recall: to what extent has CICCL been able to identify the typos in the list? Precision then expresses to what extent CICCL has been able to discern between valid and invalid word forms. In other words, we desire the list returned by CICCL to contain as many as possible of the typos in the list and as few

as possible of the correct words. In fully-automatic mode we would take the list as returned and discard all items in it from the lexicon we provide to TISC. This would require very high precision as we would otherwise throw away scores of valid words. Barring a sufficiently high level of precision, we might opt to visually inspect the list, remove valid words and then proceed to clean our lexicon or corpus. In this case we would still like to have reasonable levels of recall and precision, otherwise we might as well opt to visually inspect the full corpus-derived frequency list.

True Positives here are of course those typos present (and marked as such) in the evaluation set and retrieved as variants for focus words by CICCL. **False Positives** come in two kinds: first there are those focus words which happen to be marked as typos in the evaluation set. Second, there the valid words in the evaluation set which are incorrectly returned as variants by the system. **False negatives** are those items in the list of invalid words that are absent from the list of variants returned.

5.2. Evaluation results

The scores obtained by CICCL on Dutch are: recall: 0.464, precision: 0.463, F-score: 0.464, when run with the limit set at LD = 1. With the limit set at LD 2, we obtain: recall: 0.560, precision: 0.344, F-score: 0.426. We thus gain recall, but lose precision using higher retrieval bounds. The results obtained here are very much in line with the results we achieved by means of fully-fledged spelling correction of Dutch with Test-Induced Spelling Correction or TISC (Reynaert, 2005). However, the present results were achieved with far simpler means, after next to no optimalisation. TISC relies heavily on a lexicon which apart from single word forms as used here, also contains word bigrams, combinations of two words as derived from the corpora. We believe that extending the current version of CICCL with that information will eventually allow us to increase its performance. While low, we believe these scores are nevertheless valuable. The object of evaluating systems, apart from having objective scores by means of which one may compare systems addressing the same task(s), is to have a valid measure of a particular system's achievement in order to be able to gauge the impact on achievement extensions or changes have.

Recall is lower than was perhaps expected. Examination of why this should be so showed us that the system still loses highly recurrent typos. We see, e.g. that while the variants for *commission* include the highly frequent typo *commisie, this is validated on the basis of its frequency being higher than the threshold set by the Zipf filter, i.e. here the average frequency per particular word length observed in the Dutch background frequency list. Nevertheless, several of the typo's occurrences as part of compounds are correctly retained: their frequency is not so high. This suggests an obvious, recursive way of recovering highly frequent typos: on the basis of the retained compounds we should be able to determine that the first part of the compound is typographically correct and that therefore the second is incorrect and is to be added to the list of typos returned.

Precision tells us a different story: on examining the False Positives we have to conclude that it is exceedingly hard to build a consistent evaluation set of the type used here for a language such as Dutch. Dutch has a history of spelling changes enforced by law. The official word list of Dutch (Taal, 1954) in its version of 1954 lists three allowable variations for the compound *cultuurprodukt* (i.e. product of culture, civilisation): i.e. *cultuurproduct* and *kultuurprodukt*. The interchangeability of *c* and *k* has since officially been restricted, but the fact remains that probably any corpus of Dutch to-date contains examples of all possible allowable variants. We have not even attempted to annotate this, which means that any such allowable (whether currently or previously allowed or not) variant returned by our system may be considered as a false positive in our evaluation. A solution to this problem lies in modelling for these kinds of systematic typographical variations, as with did in the earlier version of CICCL we described in (Reynaert, 2005). We there modeled for instance the systematic variations in the suffixes *-ise/-ize* and *-isation/-ization* in English. We did none of that here, but modeling for these language-specific variations seems warranted.

5.3. Conclusions

Using very simple means, unsupervised identification of non-words in Dutch is possible to the extent shown. We have now identified ways in which to proceed in order to make the process more effective. In future work we will draw on the lessons learnt here.

Acknowledgments This work was undertaken in the framework of D-CoI (Dutch Language Corpus Initiative), a project in the STEVIN programme of the Dutch Language Union.

6. References

Fred J. Damerau. 1964. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, Volume 7, Issue 3 (March 1964):171–176.

David Graff. 2003. The New York Times Newswire Service. *English Gigaword LDC-2003T05*.

V.I. Levenshtein. 1965. Binary codes capable of correcting deletions, insertions, and reversals. In *Cybernetics and Control Theory*, volume 10(8), pages 707–710. Original in: Doklady Nauk SSSR 163(4): 845–848 (1965).

D. Lewis, Y. Yang, T.G. Rose, and F. Li. 2004. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397.

Gonzalo Navarro. 2001. A guided tour to approximate string matching. *ACM Computing Surveys*, 33(1):31–88.

Martin Reynaert. 2004. Text Induced Spelling Correction. In *Proceedings COLING 2004, Geneva*.

Martin Reynaert. 2005. *Text-Induced Spelling Correction*. Ph.D. thesis, Tilburg University.

Woordenlijst Nederlandse Taal. 1954. *Samengesteld in opdracht van de Nederlandse en de Belgische regering*. SDU Uitgevers, Den Haag.

C. J. van Rijsbergen. 1975. *Information Retrieval*. Butterworths, London.

George Kingsley Zipf. 1935. *The psycho-biology of language: an introduction to dynamic philology*. The M.I.T. Press, Cambridge, MA, 2 edition.