Near-Duplicate Detection in the IDS Corpora of Written German

Technical Report IDS-KT-2006-01

Marc Kupietz
Institut für Deutsche Sprache
<kupietz@ids-mannheim.de>

December 2005

1 Introduction

A problem often encountered in the preparation of very large text corpora is the existence of a certain –difficult to estimate– amount of (partial) copies. In other words, large text collections, whether they come from the world wide web or from newspapers' CMS-dumps, usually contain lots of texts that either do not differ at all or differ only slightly and typically stem from the same text production act. Such (partial) copies, or more neutrally and cautiously expressed: *(near) duplicates*, not only hamper 'manual' corpus queries but, more importantly, they may also bias statistical analyses in an unpredictable manner.[1]

This paper is concerned with the first step required to deal with such corpus contaminations. It presents an algorithm for the detection of (near) duplicates in large text collections by efficiently computing complete similarity matrices, that can serve as a good basis for later identification of unwanted (partial) copies. It further introduces some basic concepts and techniques, compares two different similarity metrics, describes the application of the algorithm to the IDS corpora of written German, and makes some notes on its computational complexity and its scalability.

[1] See [3] for further discussion.

2 Basic Concepts and Techniques

In this section the basic concepts and techniques that were used to identify (near) duplicate text pairs in the IDS corpora are introduced. The order of paragraphs, more or less, reflects the order of steps in the processing pipe.

2.1 Normalisation

To remove data irrelevant for linguistic purposes and to minimise computational effort right at the beginning of the processing pipe it is reasonable to *normalise* all texts that are to be compared. For the IDS corpora the following normalisations were carried out (see also example 1a):

- all markup was removed
- sequences of space and punctuation characters were subsumed as *word boundaries*
- sequences of numerical characters were subsumed by a *number token*
- remaining non-ASCII characters were deleted
- all alphabetical characters were capitalised
- 50 frequent German stop words were removed

2.2 Units of Comparison

As basic units of comparison token-pentagrams, i.e. sequences of 5 normalised words/ tokens, were used. Different sequence lengths could have been chosen as well, but pentagrams turned out be long enough to be sufficiently discriminative, on the one hand, and short enough to avoid overlooking finer grained similarities, on the other hand.

2.3 Similarity Metric: *Shared Shingle Ratio* (ssr)

As similarity metric two different functions were used and compared. The first one is based on *shingling* [introduced in 2]. The *n-shingling* of a text is defined as the set of token-n-grams it contains. For example the 4-shingling of

(a, rose, is, a, rose, is, a, rose)

is the set:

{ (a,rose,is,a), (rose,is,a,rose), (is,a,rose,is) }

The resemblance r of two texts A and B is then defined as their *shared shingle ratio (ssr)*:

$$r_{\mathrm{ssr}}(A,B) = \frac{|S_A \cap S_B|}{|S_A \cup S_B|}$$

2.4 Similarity Metric: *Shared Shingle Coverage Ratio* (sscr)

The similarity metric of two texts A and B suggested in this paper, the *ratio of coverage by shared shingles* (sscr), is also based on common shingles ($S_A \cap S_B$) computed as above. However, for each occurrence of a common shingle in A and B the tokens covered by that shingle occurrence are marked. The similarity of A and B is then defined as the ratio of marked tokens to total tokens:[2]

$$r_{\text{sscr}}(A, B) = \frac{\text{number of tokens in } A \text{ and } B \text{ covered by common shingles}}{\text{number of total tokens in } A \text{ and } B}$$

In example 1a, for instance, tokens covered by common 5-shingles are shown in bold. The example also illustrates an advantage of the sscr metric: Especially in short texts, small distances in terms of editing operations (insertion, deletion and replacement of single words) have a very large effect on r in the *ssr*-metric ($r_{\text{ssr}}(A, B) \approx 0.2857$) while in the *sscr*-metric small editing-distances tend to have small effects on r in an intuitively good proportion to text sizes ($r_{\text{sscr}}(A, B) \approx 0.9091$). The coverage calculation can be said to avoid the negative effect of the relatively coarse granularity of 5-grams in short texts. On the other hand, r_{ssr} can be computed more efficiently (see section 3.1). Another characteristic of r_{sscr}, that can be advantageous in the computation of upper bounds (see following sections), is that $r_{\text{sscr}} \geq r_{\text{ssr}}$.

2.5 Fingerprinting

To reduce the computational complexity of storing and comparing sets of shingles, shingles were not used directly. Rather a numerical *fingerprint* was associated with each shingle. Fingerprints have the property that if two fingerprints are different their represented objects are different, too. Therefore a resemblance r' based on fingerprints has the property

$$r' >= r$$

therefore r' can be used as an upper bound for r.

The fingerprints were computed using Rabin's efficient mowing-window method [4] based on strings (i.e. 5-shingles) consisting of the 4-bit characters produced by the normalisation (letters A-Z, number-symbol, word-boundary-symbol). The initially 64-bit long Rabin fingerprints were then further reduced using a constant prime modulus P, which was chosen according to available RAM.

2.6 Indexing

To avoid the re-computation of fingerprints during the process of comparing a set of texts and to speed up the recall of fingerprint occurrences an index of fingerprints was build up gradually as follows. For each text in the input stream the corresponding 5-shingling-fingerprints were recorded in a table using the fingerprints as indices and unique text-IDs

[2] Based on shingling of course also the grade of *containment* of two texts can be defined in analogy to both similarity metrics by simply using the shorter text as reference.

(a) Two short newspaper texts and their normalisations. Tokens in the normalised texts covered by common 5-shingles (see below) are displayed in bold:

T02/NOV.53095 die tageszeitung, 01.11.2002, S. 12, Ressort: Meinung und Diskussion; Betr.: Dieter Rulff	*T03/JUL.31966 die tageszeitung, 01.07.2003, S. 12, Ressort: Meinung und Diskussion; Betr.: Dieter Rulff*
Dieter Rulff ist freier Journalist in Berlin. Nach langen Jahren bei der taz war er zuletzt leitender Redakteur der Wochenzeitung „Die Woche“. Sein Interesse gilt seit langem der Entwicklung der deutschen Innen- und Parteipolitik.	Dieter Rulff ist freier Journalist in Berlin. Nach vielen Jahren bei der taz war er zuletzt leitender Redakteur der Zeitung „Die Woche“. Sein Interesse gilt seit langem der Entwicklung der deutschen Innen- und Parteipolitik.
DIETER RULFF FREIER JOURNALIST BERLIN LANGEN **JAHREN TAZ ZULETZT LEITENDER REDAKTEUR** WOCHENZEITUNG **WOCHE INTERESSE GILT SEIT LANGEM ENTWICKLUNG DEUTSCHEN INNEN PARTEIPOLITIK**	**DIETER RULFF FREIER JOURNALIST BERLIN** VIELEN **JAHREN TAZ ZULETZT LEITENDER REDAKTEUR** ZEITUNG **WOCHE INTERESSE GILT SEIT LANGEM ENTWICKLUNG DEUTSCHEN INNEN PARTEIPOLITIK**

(b) Shared 5-shingles of the normalised texts:

$$
\begin{aligned}
S_A \cap S_B = \{&(\text{DIETER, RULFF, FREIER, JOURNALIST, BERLIN}),\\
&(\text{JAHREN, TAZ, ZULETZT, LEITENDER, REDAKTEUR}),\\
&(\text{WOCHE, INTERESSE, GILT, SEIT, LANGEM}),\\
&(\text{INTERESSE, GILT, SEIT, LANGEM, ENTWICKLUNG}),\\
&(\text{GILT, SEIT, LANGEM, ENTWICKLUNG, DEUTSCHEN}),\\
&(\text{SEIT, LANGEM, ENTWICKLUNG, DEUTSCHEN, INNEN}),\\
&(\text{LANGEM, ENTWICKLUNG, DEUTSCHEN, INNEN, UND}),\\
&(\text{ENTWICKLUNG, DEUTSCHEN, INNEN, UND, PARTEIPOLITIK}) \}
\end{aligned}
$$

(c) Similarity according to the ssr and the sscr metric, respectively:

$$
\begin{aligned}
r_{\text{ssr}}(A,B) &= \frac{|S_A \cap S_B|}{|S_A \cup S_B|} = \frac{8}{28} \approx \mathbf{0.2857}\\
r_{\text{sscr}}(A,B) &= \frac{\text{marked tokens}}{\text{total tokens}} = \frac{40}{44} \approx \mathbf{0.9091}
\end{aligned}
$$

Example 1: Comparison of two short newspaper texts.

and –in case of computing sscr-similarities– the positions of the respective shingles in the normalised text as values. With this index the occurrences of shingles/ fingerprints could be recalled in constant time.

2.7 Validation

Since in the first pass of comparison only upper bounds of similarities were computed, because of possible fingerprint collisions, the results had to be validated. The validation procedure used 5-shingles of the normalised text instead of fingerprints and was conducted on pairs of texts with a first pass resemblance value $r' \geq 0.7$.

3 Application and Results

Near duplicate detection analysis using the techniques outlined above has been implemented using *Perl*, *Flex*, and *C*.[3] The system has been run against all IDS-corpora that were suspected to contain (near) duplicates. In particular, newspaper-corpora were analysed both newspaper-wise, i.e. all volumes of a particular newspaper were searched for duplicates, and year/volume-wise, i.e. all newspaper articles of a particular year were analysed. Although doublets had already been coarsely sorted out of the IDS-corpora at preparation time, the analyses revealed a total of more than 26 million pairs of near duplicates on a 90%-level according to the sscr metric (involving about 200,000 different texts) and about 8 million pairs according to the ssr metric in about 7 million texts.[4]

3.1 Performance

As shown in figure 1a, the computation of similarity matrices for the tageszeitung-corpus —one of the largest newspaper corpora at the IDS with almost one million texts and ca. 170 million normalised tokens in total— took 2h09m using the sscr metric and 1h23m using the ssr metric.[5] These times do not include the validation passes, which were conducted on all pairs exceeding a similarity ratio of 0.7. The validation, which was parallelised on six CPUs, took about 5 and 4 minutes, respectively, corresponding to about 30 and 20 minutes CPU time. In both cases, the mean difference of the upper bound similarities compared to the validated similarities was $< +0.001$.

Polynomial regression revealed that in the given interval both first pass run time curves approximate well to second polynomials with a low coefficient in the quadratic part:

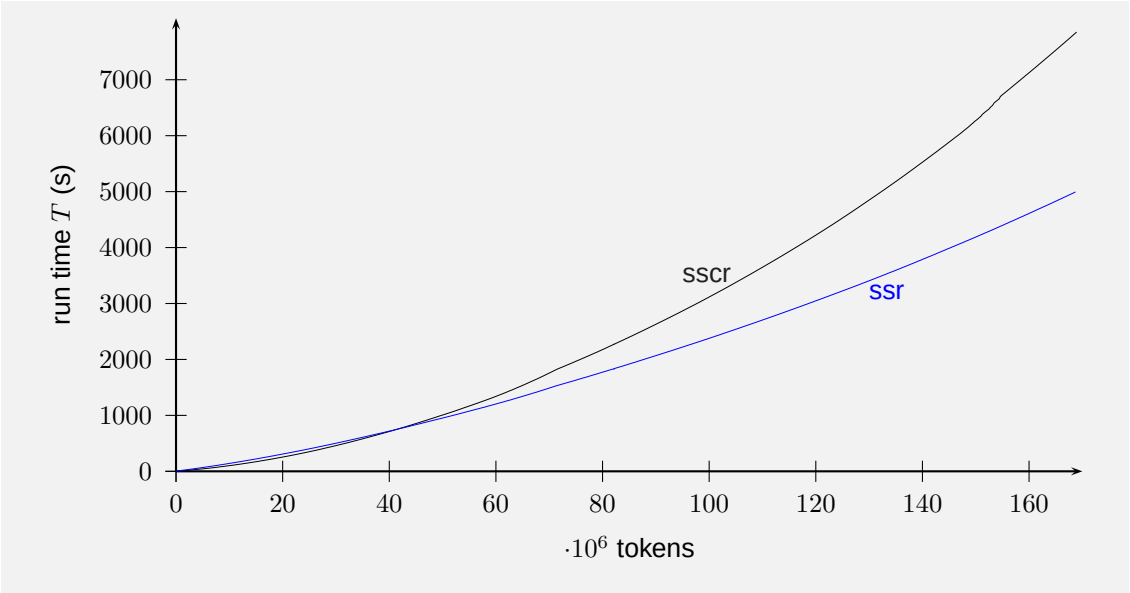
$$T(R_{\text{ssr}}, n) \approx 8.30 \cdot 10^{-14} n^2 + 16 \cdot 10^{-5} n$$
$$T(R_{\text{sscr}}, n) \approx 2.18 \cdot 10^{-13} n^2 + \ 9 \cdot 10^{-6} n$$

[3] The source code can be downloaded from `ftp://ftp.ids-mannheim.de/kt/`. Please note that the downloadable version –for optimisation reasons– slightly differs from the algorithms outlined here. In particular, *coverage ratios* are only estimated based on the common shingle coverage of only one text.

[4] The 26 million pairs corresponded to approximately 140,000 similarity clusters. See [3] for details.

[5] The analyses were run on a sun fire v880 using two CPUs à 1.2 GHz, one of them dedicated to normalisation and fingerprinting, and 4 GB RAM.

(a) Run time behaviour:



(b) Memory usage:

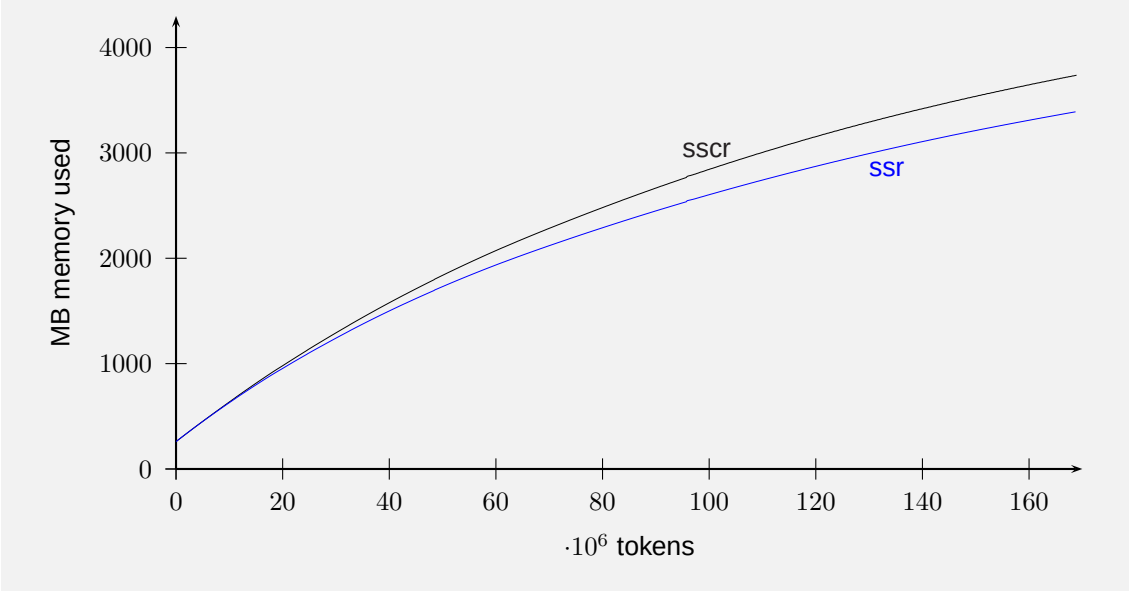


Figure 1: Performance comparison of the first pass of (near) duplicate detection runs on one million texts from the German newspaper *die tageszeitung* (CMS dump of the volumes 1986–2004).

where n is the number of total normalised tokens.

Of course, these run time functions are not very predictive for much larger n as they presuppose that the fingerprint index tables are completely held in RAM rather than swapped to disk. However, the memory complexity of both algorithms is $O(n)$ (see figure 1b),[6] so that when using the given implementations with 16GB RAM for collections containing a billion words (corresponding to about 800 million normalised tokens), run time estimates of about 41 and 19 hours, respectively, should not be too unrealistic.

4 Conclusions

With the sscr metric, a new metric describing the similarity of two texts, based on the ratio of coverage by shared shingles, has been proposed in this paper. An algorithm using this metric for the computation of similarity matrices on text collections and its application to the IDS corpora has been described and compared to an algorithm using a ssr (shared shingle ratio) metric. While the ssr algorithm is slightly more efficient in terms of time and memory usage, the similarity function of the sscr algorithm has proved to be more appropriate as basis for further copy detection analysis, due to a better near duplicate recall, especially for shorter texts. However, it remains unclear whether similar properties could be achieved by other modifications of an ssr based algorithm involving a smaller degradation in performance.

Furthermore, it has been shown that both algorithms are suitable for text collections containing up to one billion words on current standard server hardware. However, since the time complexity of both algorithms is $O(n^2)$, for very large unstructured corpora like for example web corpora, pre-structuring into sub-collections or other techniques, like the use of sampling mechanisms [see 1], are required.

5 Other Applications

The efficient computation of pair-wise similarities in data collections and the detection of recurring n-grams certainly also has other important applications than (near) duplicate detection in text corpora. Some of these are:

- detection of plagiarisms
- filtering of search engine hits
- detection of quotations
- identification of frequently recurring phrases in corpora
- identification of text templates
- news clustering and filtering for news aggregators

[6] where in case of the sscr algorithm the linear factor is slightly bigger because for each token at least two more bytes are needed to store positions of shingles in a text

References

[1] A. Z. Broder. On the resemblance and containment of documents. In *SEQS: Sequences '91*, 1998.

[2] A. Z. Broder, S. C. Glassman, M. S. Manasse, and G. Zweig. Syntactic clustering of the web. In *Proceedings of the Sixth International World Wide Web Conference*, pages 391–404, 1997.

[3] M. Kupietz. Detection of clusters of (near) duplicates, versions, and variations in the IDS-corpora. Technical report, Institut für Deutsche Sprache, forthcoming.

[4] M. O. Rabin. Fingerprinting by random polynomials. Technical Report TR-15-81, Center for Research in Computing Technology, Harvard University, 1981.