Hash Table Sizes for Storing N-Grams for Text Processing

Zhong Gu and Daniel Berleant
Electrical and Computer Engineering
2215 Coover Hall
Iowa State University
Ames, Iowa 50011
{zhonggu,berleant}@iastate.edu

Abstract

N-grams have been widely investigated for a number of text processing tasks. However n-gram based systems often labor under the large memory requirements of naïve storage of the large vectors that describe the many n-grams that could potentially appear in documents. This problem becomes more severe as the number of documents (and hence the number of vectors to store and process) rises. A natural approach to reducing vector size is to hash the large number of possible n-grams into a smaller vector. We address this problem by identifying good and bad hash table sizes over a wide range of sizes. We show that English, French, and German n-grams behave similarly when hashed, and that this is unlike the behavior of randomly generated n-grams. Therefore the difference in behavior is due to properties of the languages themselves. We then investigate different table sizes and identify which sizes are particularly good when hashing n-grams during processing of these languages.

Keywords

N-grams, polygrams, multigraphs, cosine metric, document vectors, information retrieval.

1 Introduction

Natural language text processing applications such as information retrieval, document comparison, and document clustering make extensive use of string comparisons. Because texts generally are constructed from words, such comparisons have frequently relied on comparing the word profiles of the texts. However pure word comparison has significant shortcomings. Words have different lengths leading to computational efficiency deficits in speed and memory relative to computations on constant-length strings. Furthermore, assessing similarity between different but related words (such as different inflections of the same word root) can be important in text processing, and the naïve approach of recording a match between two words only if they are fully identical frequently supports merely a baseline performance level from which significant improvement can be sought.

Stemming algorithms can alleviate the word match problem, but at the cost of additional computation. N-grams can both alleviate the word match problem and support improved computational efficiency compared to word based processing, because n-grams are simply strings of length n. Thus 4-grams all have length 4, 5-grams have length 5, etc. N-grams support partial matching when texts contain different but similar words because the similarities between the words cause the passages to have n-grams in common. For example, "computer" and "computation" are different words but share two 5-grams, "compu" and "omput".

Uses of n-grams. N-grams were investigated for tasks related to information retrieval at least as early as 1979 (Suen[21]). Since then they have been investigated in such tasks as language identification (Damashek 1995[6]; Sibun and Reynar 1996[19]), spelling correction (Zamora et al. 1981[23]; Salton 1989[20]), document categorization (Huffman and Damashek 1994[13]; Labrou and Finin 1999[14]), document comparison (Damashek 1995[6]), robust handling of noisy (misspelled, OCR'ed etc.) texts (Grossman et al. 1995[11]; Pearce and Nicholas 1996[18]; Pearce and Miller 1997[5]), topic highlighting (Cohen 1995[3]), document space visualization (Fox et al. 1999[9]; Huffman 1995[12]; Charoenkitkarn et al. 1994[2]), spoken document retrieval (Ng and Zue 2000[17]), and other information retrieval related applications (Grossman 1994[10]; Cavnar 1994[1]). 5-grams have been most thoroughly investigated (e.g. Damashek 1995[6]) and have emerged as an n-gram size capable of supporting even higher information retrieval precision and recall than words as shown using the TREC-7 Ad Hoc task (Mayfield and McNamee 1998[15]). Unfortunately, n-grams that are too long will fail to capture similarity between different but similar words, and n-grams that are too short will tend to find similarities between words that are due to factors other than semantic relatedness.

Problem. Despite the advantages of 5-gram based text processing, the number of different 5-grams creates its own challenges. There are $26^5 \approx 10^7$ possible sequences of 5 letters and roughly 50x more than that if other common characters (spaces, digits, punctuation) are included. To implement a table containing the number of occurrences of each possible 5-gram in a particular document in an array containing one entry per possible n-gram would thus require a very large array. Memory requirements have posed significant problems (Ng and Kantor, 1995[16]), yet as memory becomes more available, the number of documents to process tends to increase as well. Thus, it would be useful to encode documents in terms of their 5-grams in a more memory-efficient manner (Crowder and Nicholas 1996[4]).

Because most of the possible sequences of n characters rarely or never occur in practice for n=5, a table of the n-grams occurring in a given text tends to be sparse, with the majority of possible n-grams having a frequency of zero even for very large amounts of texts. For example, 40 MB of text from the Wall Street Journal were found to contain only $2.7*10^5$ different 5-grams out of a possible $7.5*10^{18}$ (based on an alphabet of 27 characters, Ebert et al. 1997[8]). Even the entire Web is quite sparse (Table 1).

Ten random letter 5-grams	Number found on the Web
obive thjfs jpomz aqzmk owsop znqfm xifiq mkxre zgwhb jclur	2 (jclur & obive)

Table 1. Ten random letter 5-grams were generated, then searched for on the Web. Only two were found by the Alta Vista search engine (as of 8/2/00) anywhere on the entire Web in a non-binary document or its URL, in any human language. For a single document, a vastly lower percentage of all possible n-grams would be present.

Sparseness is further increased by the fact that the number of distinct n-grams in a given text is strictly limited by the document's length. A document containing M characters contains a maximum of $M-n+1\approx M$ distinct n-grams, and normally contains fewer since some n-grams occur more than once.

Solution approach. The sparseness of the n-gram table suggests compacting it. One way to reduce the size of the table of n-grams in a text is to simply store a list of the n-grams that actually occur, in association with the number of times each appears in the text. This provides an accurate accounting. Another way to reduce the table size is to hash the large space of possible n-grams into a considerably smaller array. This risks collisions, which may be duly recorded for accuracy or, alternatively, ignored for the sake of computational efficiency, in which case a given location in the array could contain a number expressing the summed occurrences of two (or more) n-grams.

The computational advantages of ignoring collisions are substantial. Even if only the 26 English letters are considered, there are still 26^5 potential 5-grams. Interpreting the bit pattern of each 5-gram as an integer without hashing leads to a sparse 256^5-element array for storing the n-gram content of any given document. Because of its sparseness, a great amount of memory can be saved by hashing, with tolerably few collisions. For example, with a hash table of size$\approx 2^{18}$, a document can be represented by an array with about 2^{18} elements, which is 0.000024% of 256^5. If a hash table of size$\approx 2^{25}$ is used, that size is still only 0.0031% of 256^5. As a result, the strategy of hashing 5-grams and ignoring any resulting collisions to support speed and simplicity has been found useful (Damashek 1995[6]).

This paper investigates collisions in 5-gram hash tables, with the goal of minimizing their occurrence. We show that, using the commonly employed hash function *h(k)=k%tablesize* for prime *tablesize*, different table sizes exhibit wide variation in collision rate. We show that this pattern of variation is similar for the three languages investigated, English, French, and German, but not for randomly generated n-grams. In order to support systems that use hash tables of 5-grams, we empirically determine and provide a list of table sizes covering a range from approximately 2^{16} to 2^{40} that have low collision rates. Choosing from the table sizes in this list avoids the possibility of inadvertently choosing a table size with an average or a high collistion rate.

2 Methodology

We investigated the effect of hash table size on collision rate for 5-grams. To hash them, each was converted to an equivalent 5-byte integer *i* from 0 to 2^{40}-1, as follows:

Let the characters in the n-gram be named $a_0 \ldots a_4$, starting from the leftmost character.

Let $num(a_n)$ be the 1-byte integer equivalent of character a_n.

(1) $i= a_0+256a_1+256^2a_2+256^3a_3+256^4a_4$

The resulting values were hashed using the standard hash function *h(i)=i%tablesize* where *tablesize* is prime. For each language tested, 100 documents from the Web were hand-picked and checked to ensure that each was written in the desired language and was a real document of reasonable length. This resulted in a total of 436,950 bytes of English, 469,638 bytes of German, and 446,032 of French. The English documents were from an ad hoc, diverse set of sources, while the French and German documents were from various university sites in those countries.

In order to determine when a collision occurred, the hash codes were first stored in an array in which one element stored the hash code of each n-gram in the file. The hash code of the *k*th n-gram in the document was stored at array index *k*. This required *b-4* array elements for the *b-4* n-grams in a file of *b* bytes. Then, the array was scanned for recurring hash codes. A recurring code could indicate a collision, if the two occurrences of the code derived from different n-grams, or a non-collision, if the two occurrences derived from different occurrences of the same n-gram. This was determined by using the array index at which a hash code was stored as the offset into the document where the corresponding n-gram would be located.

3 Results and Discussion

We investigated a variety of table sizes between 2^15 to 2^40 for each of the three languages, as described next.

Collision rates, table sizes and languages. Different table sizes exhibit wide variation in collision rates (Figure 1). These variations are similar for English, French and German. Observe the *'s in the three overlapping graphs of Figure 1. Each * represents an average of the collision rates for the 20 prime table sizes nearest to but below a given power of 2. In all three graphs the pattern of *'s is generally descending, but exhibits certain upward exceptions, for example at 2^{24}. For all three languages, the average collision rate that occurred when hashing the n-grams in a document is substantially higher, on average, for table sizes just under 2^{24} than for table sizes just under 2^{23}, 2^{22}, and even 2^{21}. Similarly, upward exceptions to the generally descending trend occur at 2^{31} and 2^{32} for all three languages. This contrasts with a control condition in which a similarly size set of randomly generated n-grams were processed and graphed the same way (Figure 2).

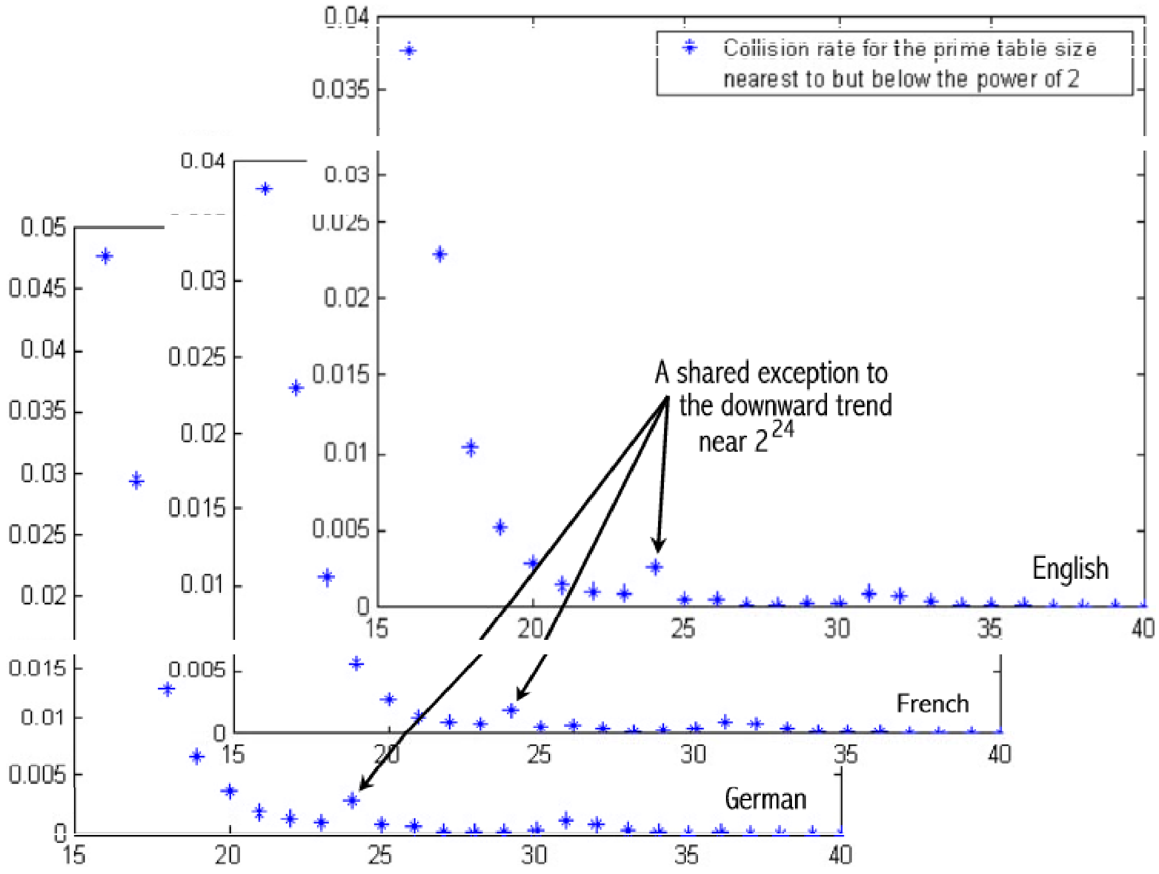

Figure 1. Graphs showing collision rates (y-axis) vs. hash table sizes (x-axis). Each plotted point is the average for the 20 largest prime table sizes below 2 to the powers 15 through 40. English, French, and German all show a trend of generally decreasing collision rate with increasing table size, consistent with well known properties of hash tables. Obvious upward exceptions to the downward trend are evident in the *'s for table sizes near the same powers of 2 for all three languages, though not for randomly generated n-grams (as shown in Figure 2).

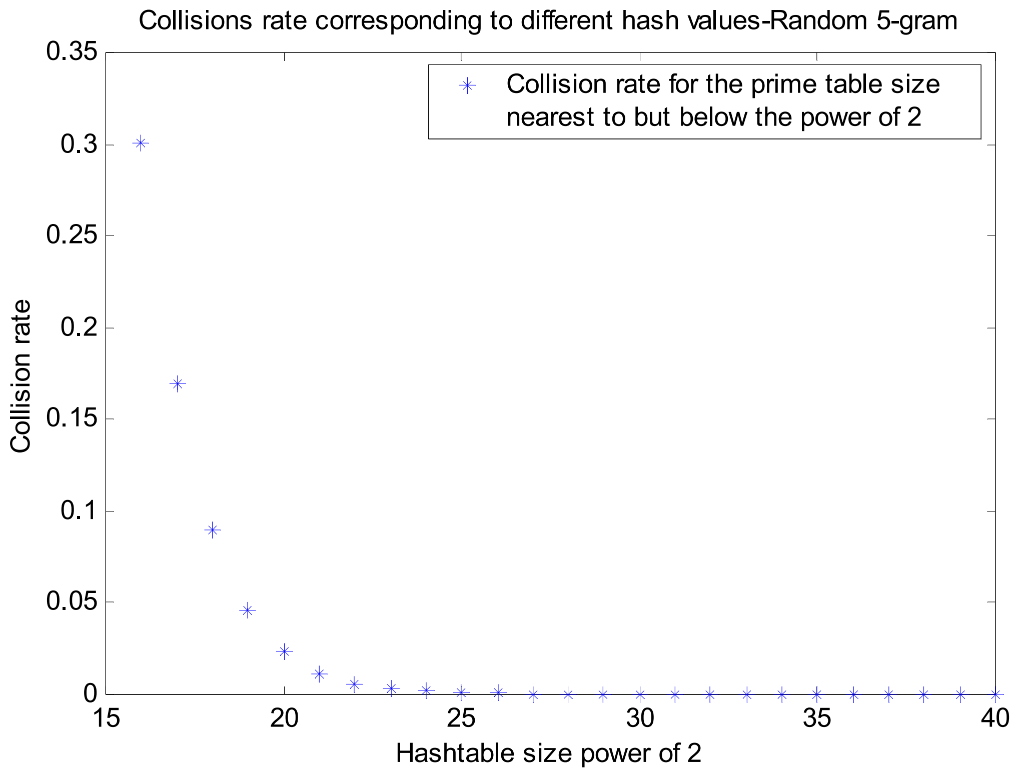

Figure 2. The collision rates corresponding to different hash table sizes for randomly generated 5-grams. This is the control condition, which we compare with the graphs for the English, French and German shown in Figure 1.

To determine the statistical significance of these results, observe that for each language, the highest magnitude exceptions to the downward trend for the *'s are at the same three points out of the 25 plotted, 2^{24}, 2^{31}, and 2^{32}. Given that a language L_1 exhibits this property, we seek the probability of a statistical null hypothesis that both other languages L_2 and L_3, exhibit the same property due to chance.

1) Assumption: L_2 and L_3 each have at least three exceptions to the downward trend. This assumption is conservative, therefore allowable, and is met in the current case. Call these exceptions E_1, E_2, and E_3.
2) *$P(E_1$* is in the same place in the sequence of 25 plotted points as one of the three highest magnitude exceptions in *$L_1)=3/25$.* Remove that place from further consideration.

3) *P(E*$_2$ is in the same place in the sequence of the remaining 24 plotted points as one of the two remaining highest magnitude exceptions in *L*$_1$*)=2/24*. Remove that place from further consideration.

4) *P(E*$_3$ is in the same place in the sequence of the remaining 23 plotted points as the remaining highest magnitude exception in *L*$_1$*)=1/23.*

5) *P(*all three highest magnitude exceptions in *L*$_2$ correspond to the exceptions in *L*$_1$*)*= (3/25)*(2/24)*(1/23)=0.000435.

6) For the third language *L*$_3$, the analogous probability, *P(*all three highest magnitude exceptions in *L*$_3$ correspond to the exceptions in *L*$_1$ and *L*$_2$) is also 0.000435.

7) *P(*both *L*$_2$ and *L*$_3$ have the same pattern of exceptions*)*=0.000435^2=$1.9*10^{-7}$ which effectively rules out the hypothesis that the results are due to chance.

Empirical determination of good table sizes. In order to support systems that use hash tables of 5-grams, we empirically investigated hash tables whose sizes are the 20 largest primes below powers of 2, for each power of 2 from 2^{16} to 2^{40}. From this we identified which of the 20 table sizes resulted in the lowest collision frequency, for each power of 2. Given a power of 2 representing the approximate table size desired, using the table size identified means both using a table size with a collision rate lower than for nearby table sizes, and also avoiding the possibility of inadvertently choosing a table size with a particularly high collision rate. Table 2 shows these table sizes together with the collision rate for each. To choose a good table size for a particular application, pick a table size from the "Best hash table" column. The other columns compare these table sizes with similar, naïvely chosen table sizes and with the worst table sizes from the set of 20 from which the best table size was identified.

Power of 2	Best hash table size	Collision rate	Worst hash table size	Collision rate
16	65393	0.03453484380364	65407	0.04291337681657
17	130843	0.01656482435061	131071	0.06934431857192
18	262051	0.00804668726399	262139	0.01564938780181
19	524203	0.00379448449479	524287	0.01173131937293
20	1048423	0.00181714154938	1048571	0.00437807529466
21	2096957	0.00082847007667	2097143	0.00254262501430
22	4194181	0.00039134912461	4194301	0.00307586680398
23	8388283	0.00030667124385	8388427	0.00293855132166
24	16776931	0.00041423503833	16776961	0.02023343631995
25	33554239	0.00028836251287	33554347	0.00181485295800
26	67108747	0.00013502689095	67108859	0.00378761872068
27	134217493	0.00002517450509	134217467	0.00080787275432
28	268435091	0.00002517450509	268435331	0.00029522828699
29	536870791	0.00001373154823	536870909	0.00124957088912
30	1073741467	0.00002288591372	1073741419	0.00116947019110
31	2147483249	0.00002288591372	2147483647	0.01076324522257
32	4294966769	0.00004348323607	4294967291	0.00602357249113
33	8589934289	0.00003890605332	8589934583	0.00220162489987
34	17179868809	0.00001830873098	17179869107	0.00038906053324
35	34359737821	0.00000228859137	34359738299	0.00028378533013
36	68719476619	0.0	68719476731	0.00070259755121
37	137438953331	0.0	137438953403	0.00007323492390
38	274877906813	0.0	274877906687	0.00003890605332
39	549755813869	0.0	549755813797	0.00001830873098
40	1099511627689	0.0	1099511627689	0.0

Table 2. Hash table sizes and collision rates. The best hash table size is the table size with the lowest collision rate of the 20 largest prime table sizes below 2 to the power indicated in the left hand column. The worst hash table size is the table size with the highest collision rate of the same 20 table sizes.

4 Conclusion

5-grams have been found useful for profiling documents in a number of text processing tasks. To best exploit the computational advantages that hash tables provide in storing the 5-gram profiles of documents, a common and simple hashing scheme should be used, and collisions should be ignored but their rate should be minimized by appropriate choice of a hash table size. We have shown that the properties of English, French, and German are similar for this purpose, and have empirically identified good hash table sizes over a wide range.

References

1. Cavnar, W., Using An N-Gram-Based Document Representation With A Vector Processing Retrieval Model, in NIST Special Publication 500-226: Overview of the Third Text REtrieval Conference (TREC-3), 1994, 269-278. http://trec.nist.gov/pubs/trec3/t3_proceedings.html.
2. Charoenkitkarn, N., M. Chignell, and G. Golovchinsky, Interactive Exploration as a Formal Text Retrieval Method: How Well can Interactivity Compensate for Unsophisticated Retrieval Algorithms, in NIST Special Publication 500-226: Overview of the Third Text REtrieval Conference (TREC-3), 1994, 179-199. http://trec.nist.gov/pubs/trec3/t3_proceedings.html.
3. Cohen, J. D. 1995. Highlights: Language- and Domain-Independent Automatic Indexing Terms for Abstracting. Journal of the American Society for Information Science 46(3), 162--174.
4. Crowder, G. and C. Nicholas, Resource Selection in CAFÉ: An Architecture for Network Information Retrieval, Proceedings of the Network Information Retrieval Workshop, SIGIR 96, August 1996.
5. Pearce, C. and E. Miller. The telltale dynamic hypertext environment: Approaches to scalability. In James Mayfield and Charles Nicholas, editors, Advances in Intelligent Hypertext, Lecture Notes in Computer Science. Springer-Verlag, 1997.
6. Damashek, M., Gauging similarity with n-grams: Language-independent categorization of text, Science, 267 (1995), pp. 843 – 848.
7. D'Amore, R. and C. Mah, One-time complete indexing of text: Theory and practice, in Proceedings of SIGIR 1985, 155-164, 1985.
8. Ebert, D.S., C. D. Shaw, A. Zwa, E.L. Miller, and D.A. Roberts, Interactive Volumetric Information Visualization for Document Corpus Management, Proceedings of Graphics Interface '97, Kelowna, B.C., May 1997, 121-128. http://www.dgp.toronto.edu/gi/gi97/proceedings/.
9. Fox, K., O. Rieder, M. Knepper, and E. nowberg "SENTINEL: A Multiple Engine Information Retrieval and Visualization System," Journal of the American Society of Information Science, 50(7), May 1999. http://www.csam.iit.edu/~ophir/infret.html.
10. Grossman, D.A., A Parallel DBMS Approach to IR in TREC-3, in NIST Special Publication 500-226: Overview of the Third Text REtrieval Conference (TREC-3), 1994, 279-288. http://trec.nist.gov/pubs/trec3/t3_proceedings.html.
11. Grossman, D.A., D.O. Homes, O. Frieder, M.D. Nguyen, and C.E. Kingsbury, Improving Accuracy and Run-Time Performance for TREC-4, in NIST Special Publication 500-236: The Fourth Text REtrieval Conference (TREC-4), 1995, 433-448. http://trec.nist.gov/pubs/trec4/t4_proceedings.html.
12. Huffman, S., Acquaintance: Language-Independent Document Categorization by N-Grams, in NIST Special Publication 500-236: The Fourth Text REtrieval Conference (TREC-4), 1995, 359-372. http://trec.nist.gov/pubs/trec4/t4_proceedings.html.
13. Huffman, S. and M. Damashek, Acquaintance: A novel vector-space n-gram technique for document categorization, in NIST Special Publication 500-226: Overview of the Third Text REtrieval Conference (TREC-3), 1994, 305-310. http://trec.nist.gov/pubs/trec3/t3_proceedings.html.
14. Labrou, Y. and T. Finin, Experiments on using Yahoo! categories to describe documents, in IJCAI-99 Workshop on Intelligent Information Extraction (July 1999). http://www.aifb.uni-karlsruhe.de/WBS/dfe/iii99/.

15. J. Mayfield, P. McNamee, Indexing Using Both N-Grams and Words, in NIST Special Publication 500-242: The Seventh Text REtrieval Conference (TREC 7), 1998, 419-424. http://trec.nist.gov/pubs/trec7/t7_proceedings.html.
16. Ng, K.B. and P.B. Kantor, Two Experiments on Retrieval With Corrupted Data and Clean Queries in the TREC-4 Adhoc Task Environment: Data Fusion and Pattern Scanning, in NIST Special Publication 500-236: The Fourth Text REtrieval Conference (TREC-4), 499-508, 1995. http://trec.nist.gov/pubs/trec4/t4_proceedings.html.
17. Ng, K. and V.W. Zue, Subword-Based Approaches for Spoken Document Retrieval, *Speech Communications* (to be published, 2000). http://www.sls.lcs.mit.edu/~kng/papers/speechcomm2000.pdf.
18. Pearce, C. and C. Nicholas, TELLTALE: Experiments in a Dynamic Hypertext Environment for Degraded and Multilingual Data, Journal of the American Society for Information Science (JASIS), April 1996.
19. Sibun, P., & Reynar, J. (1996). Language identification: Examining the issues. In Symposium on Document Analysis and Information Retrieval, pp. 125-135, Las Vegas.
20. Salton, G., Automatic Text Processing, Addison Wesley Pub. Co., 1989.
21. Suen, C.Y., N-gram statistics for natural language understanding and text processing, IEEE Trans. on Pattern Analysis and Machine Intelligence, PAMI 1 (2) (1979) 164-172.
22. Zamora, E.M., J.J. Pollock, and A. Zamora, The use of trigram analysis for spelling error detection, Information Processing and Management 17 (6) (1981) 305-316.

Author Biographies

Zhong Gu received the BS degree and MS degrees in Electrical Engineering in 1995 and 1998, respectively, from Xidian University. He is now a master's degree candidate in Computer Engineering, in the Department of Electrical and Computer Engineering at Iowa State University. His research interests include multimedia browsing systems and use of software engineering techniques to support pedagogy across the EE and CSE curricula.

Daniel Berleant received the BS degree in Computer Science and Engineering in 1982 from the Massachusetts Institute of Technology, and the MS and PhD degrees in 1990 and 1991, respectively, in Computer Science from the University of Texas at Austin. He joined the Department of Electrical and Computer Engineering at Iowa State University in 1999 as an Associate Professor. His research interests include multimedia browsing and text mining systems, arithmetic on random variables of unknown dependency, use of software engineering techniques to support pedagogy across the EE and CSE curricula, and technology foresight. He is a member of ACM, IEEE, and the IEEE Computer Society.