Replacing Personally-Identifying Information in Medical Records, the Scrub System

Latanya Sweeney

Clinical Decision Making Group
Laboratory for Computer Science
Massachusetts Institute of Technology
Cambridge, Massachusetts

We define a new approach to locating and replacing personally-identifying information in medical records that extends beyond straight search-and-replace procedures, and we provide techniques for minimizing risk to patient confidentiality. The straightforward approach of global search and replace properly located no more than 30-60% of all personally-identifying information that appeared explicitly in our sample database. On the other hand, our Scrub system found 99-100% of these references. Scrub uses detection algorithms that employ templates and specialized knowledge of what constitutes a name, address, phone number and so forth.

INTRODUCTION

One of the biggest challenges facing medical informatics is the sharing and dissemination of medical records while maintaining a commitment to patient confidentiality. Retrospective research, reduced institution costs, improved medical care and the development of electronic medical record systems[1,2,3] are some of the benefits possible when the content of medical records are reviewed by professionals, but these researchers, administrators, medical students, and computer scientists stretch the general notion of patient confidentiality. What is needed is a mechanism to provide only the information necessary to the professional who has a need to know. The problem is not as simple as searching for the patient's name and replacing all occurrences with a pseudo name. Identifying information is often hidden in the shorthand notes of clinicians and in letters exchanged between doctors. References to the patient are often quite obscure, consider for example, "he developed Hodgkins while acting as the U.S. Ambassador to England and was diagnosed by Dr. Frank at Brigham's." The goal of the Scrub system described in this paper is to provide a methodology for removing personally identifying information in medical records so that the integrity of the medical information remains intact even though the identity of the patient remains confidential. We term this process "scrubbing."

BACKGROUND

We worked with machine and hand-scrubbed samples from two different computer-based patient record systems.[4,5] A close examination quickly revealed that much of the medical content resided in the letters between physicians and in the shorthand notes of clinicians since this is where providers discussed findings, explained current treatment and furnished an overall view of the medical condition of the patient. In these cases, clinicians wrote text with little regard to word-choice and in many cases without concern to grammar or spelling. While the resulting "unrestricted text" is valuable to understanding the medical condition and treatment of the patient, it poses tremendous difficulty to scrubbing since the text often includes names of other care-takers, family members, employers and nick names. In the case of notes, the recorded messages are often cryptic abbreviations specific to the institution or known only among a group of physicians within the facility. The traditional approach to scrubbing appears to be straightforward search and replace which misses these references.

Computational Architectures

Our Scrub system utilizes numerous detection algorithms competing in parallel to label contiguous characters of text as being a proper name, an address block, a phone number, and so forth. Each detection algorithm recognizes a specific entity, where recognizable entities are fields such as first name, last name, street address, and date. There is only one detection algorithm for each entity and at first glance the entities may appear to overlap. For example, there is a detection algorithm for full names and another for first names and yet another for last names. The reason is quite simple. If you encountered "John Smith" you may know that "John" is a common first name and "Smith" is a common last

0195—4210/96/$5.00

name and so deciding that "John Smith" is a full name seems to rely on recognizing the first and last names. However, you may also recognize "A.W. Gross" as a full name because of the practice of using initials before a last name and because of the use of capitalization and punctuation, and so identifying a full name is not the same as independently recognizing its constituent parts.

Detection algorithms in Scrub use local knowledge sources, share results and compete based on the certainty of their findings. Knowledge sources are lists such as those for area codes and first names and helping routines such as those that determine whether words "sound" like medical terms or last names. Each algorithm tries to identify occurrences of its assigned entity, where an entity is a date, a city and so on, and reports for each character in the text how likely it is that the character is part of an instance of the algorithm's assigned entity. The algorithm with the highest precedence and the greatest certainty above a minimal threshold prevails and its results may be made available to all detection algorithms for future use.

In the area of speech recognition, Hearsay-II's blackboard architecture[6] engages multiple knowledge sources that work in parallel. Adjacent sources communicate with each other using a message center called a blackboard. This is similar to our Scrub system except communication is central to Hearsay-II because each level is believed to be so uncertain that a collaborative effort is required and not a competitive one. Another similar approach in speech recognition is the BeBe system[7] where parallel detection circuits compete to identify phonemes in the original sound, but the BeBe system has no global communication at all.

In Ether[8] decentralized parallel processing was shown to be an effective alternative to many kinds of heuristic search that implemented backtracking. Parallelism in Ether, as in Scrub, is design-based and does not necessarily require parallelism in its implementation. However, Ether doesn't use probabilities as a scoring system between competing processes; instead the first process to complete the task solves the problem.

METHODS

The database we used was a scrubbed subset of a pediatric medical record system[4]. It consisted of 275 patient records and included 3,198 letters to referring physicians. Many of the letters were delimited notes but most were proper letters with a heading block, salutation and well-formed sentences.

Experiment: Human Approach

We conducted an experiment to determine how well humans locate personally-identifying information in letters between physicians. The subjects were 5 adults. None of the subjects had any medical experience or experience with the information contained in the database.

Each of the adults were given a marker that writes in a read-through yellow color and seven (7) printed letters. One of the letters appeared with all its text in uppercase but consisted of complete sentences. The other letters were in standard letter format with upper-lower case. Each subject was asked to highlight all information in each letter that personally identified any person and to do so within 30 seconds per letter.

All the subjects found all obvious references to names, addresses, organizations, cities, states, zip codes and phone numbers (100%). More obscure occurrences such as nick names, abbreviations, identification numbers and incorrect capitalization were sometimes missed (99%). References embedded in the text that did not appear in upper-lower case were sometimes missed (95%) and performance on identifying obvious references in the upper case letter was much worse than in the upper-lower case counterparts (94% compared to 100%). Subjects reported reviewing most words in the letters but all subjects stated they did not read the letters.

Computer Approach: Design and Implementation

We sought to model the human approach because it did not require a complete semantic model. The subjects used templates and localized knowledge to recognize personally-identifying information. Consider the list of names, phone numbers and dates in Table 1. The writing conventions and immediate context help identity the kind of information presented.

Names	Phone numbers	Dates
Frank Graves	255-1423	March 1, 1991
F. R. Graves, MD	(304) 255-1423	3/1/91
Dr. Graves	304/ 255-1423	first of March
Frank Red Graves	255-1000 ext 1423	1-MAR-91
"Red" Graves	phone: 255-1423	03-01-91
frank red graves	extension 1423	March 1st

Table 1. Samples of personal information.

In fact, the recognition of personally identifying information requires exposure to the common recording practices of society with respect to personal information and to enough examples of personal information to determine what occurs frequently. For example, Fred and Bill are common first names and Miller and Jones are common last

names and knowing these facts makes it easier to recognize them as likely names. Common facts along with their accompanying templates of use are considered commonsense knowledge[9] and the itemization and use of commonsense knowledge is the backbone of our Scrub system. We include lists of commonly known information such as first names, last names, nick names, abbreviations for U.S. states, and so forth and the algorithms themselves embed recognition templates. Table 2 lists some of the types of entities detected by Scrub.

Scrub	Entities
1. identification block	*{6, 7, 8, 9, 10, 15, 11, 12, 13, 14, 25, 16, 17, 18, 20, 21, 25}*
2. mailing label	*{6, 7, 8, 9, 15, 11, 12, 13, 14, 17, 18}*
3. address block	*{6, 7, 8, 9, 15}*
4. full name	*{11, 12, 13, 14, 17}*
5. location	*{7, 8, 15}*
6. street	15. country
7. city	16. social security
8. state	17. title
9. zip	18. organization
10. phone	19. measurement
11. first name	20. age
12. middle name	21. date
13. last name	22. medical term
14. nick name	25. reference number

Table 2. Some of the entities recognized by Scrub are listed above in relative order of precedence.

For each entity there is a detection algorithm and the precedence of the algorithm is based on the number of entities that constitute the algorithm's assigned entity. Examples of constituent entities are listed in braces in Table 2. For example, detecting a geographical location may make it possible to identify a city, a state or a country. The more constituents an entity has, the higher its precedence. Table 2 shows five levels of precedence with identification block having the highest and entities 6 through 25 all having the same low precedence.

Detection algorithms can be executed sequentially in order of precedence to avoid parallel execution. For each character in the input text the detection algorithm with the highest precedence reporting the greatest likelihood above a threshold value is considered to have identified an instance of its entity. Diagram 1 provides an overview.

Knowing what instances have already been found in the text can be quite useful in reducing ambiguity. For example, if the system encountered the name "Virginia P. Weston" then later encountered a sentence that read "After seeing Virginia this time, I feel much better," the system could more reliably interpret the second reference to Virginia as a person's name and not the state. When an instance of an entity is found in the text, its corresponding detection algorithm can post its results -- making them available to all detection algorithms while processing the remainder of the document. In these cases, an entity can only post values for its constituent entities and if there are no constituents, it can post for itself.

A few detection algorithms work differently. Some classify the format of the document as being a letter, notes, or delimited text. These detectors continuously report findings. There are also special detectors like those for medical terms and verbs whose instances are typically not replaced but are detected because having their results reduces the number of false positives. At run-time the user sets the threshold and use of special detectors.

Table 3 repeats the second column of Table 1 but includes associated templates and probabilities. During a training session on the database, template probabilities are adjusted and their effectiveness measured. If there is not enough variation between templates then performance will deteriorate. If templates use features that are not present in the database, performance may deteriorate. For example, if name templates expect names to be written in upper-lower case then these templates will be useless if all text appears in uppercase. The training session pinpoints problem areas and weaknesses beforehand.

Phone numbers	Templates	Likelihood
255-1423	ddd - dddd	40
(304) 255-1423	(ddd) ddd - dddd	85
304/ 255-1423	ddd / ddd - dddd	50
255-1000 ext 1423	ddd - dddd ext* d*	70
extension 1423	ext* d*	40
phone: 255-1423	{tel*, ph*} ddd - dddd	90

Table 3. Samples of templates and their probabilities. The d is a digit, the asterisk matches any wild character and the set notation denotes possibilities.

As we've shown, the detection algorithms employ a host of lists. For example, detecting a first name may use a stored list of common first names, the first names of all patients, words that sound like first names or all three depending on the user's specifications. These lists are compiled beforehand. Storage requirements and speed are dramatically reduced using multiple hashed Boolean lookup tables[10] or in the case of words that sound like a group of words, using a table of orthographic rules.[11]

With Boolean look-up tables, look-ups are done in constant time, O(10) since there are 10 binary checks per word. Using orthographic rules, look-ups require O(2n) time where n is the number of syllables in the word. Storage using Boolean look-up tables require roughly 30 bits per word which is a tiny fraction of a typical word list or dictionary.[10]

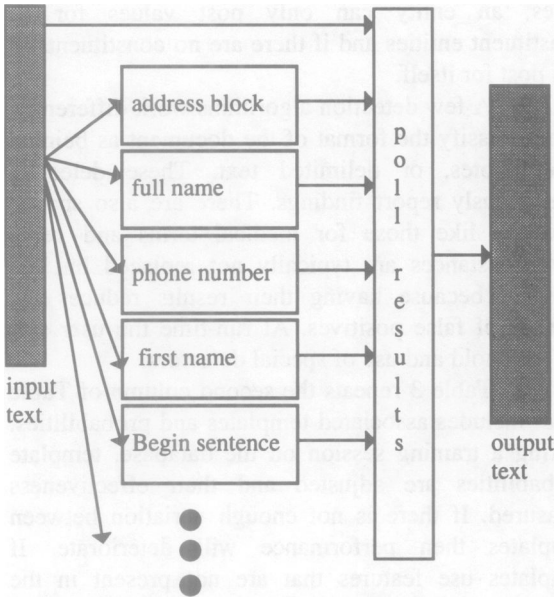

Diagram 1. Block diagram of Scrub detection system.

Replacement Strategies.

Once personally identifying information is detected, it must be replaced with some pseudo-value. There are several strategies for accomplishing this feat. Associated with each detection algorithm in Scrub is a replacement algorithm that is responsible for producing the replacement text. In general, the format of the replacement text matches the template that was recognized. If the detected entity was a date, for example, the replacement date may involve lumping days to the first of the nearest month or some other grouping. On the other hand if the detected entity was a first name, the typical strategy is to perform a hash-table lookup using the original name as the key. The result of the look-up is the replacement text. This provides consistent replacements since every time a particular name is encountered, it maps to the same replacement. In terms of the replacement content, several other strategies are available including the use of orthographic rules called Sprees[11] that replace personally identifying information with fictitious names that sound like reasonable names but in fact belong to no known person.

RESULTS

Numerous tests were conducted on the database to demonstrate the utility of the Scrub system. Highlights are shown in Table 4. The database contained nicely formatted letters as well as delimited notes. The straightforward approach of global search and replace used the previously stored patient information in the database to locate patient names, addresses, phone numbers and so forth in the text but it located no more than 30-60% of all personally-identifying information that appeared explicitly. The higher figure includes using additional information stored in the database to help identify the attending physician's name, identifying number and other information as well as the patient's. Since the letters were properly formatted, the heading block was easily detected and compositional cues were available using keywords like "Dear." This dramatically improved the results of the search-and-replace method; however, most references to family members, additional phone numbers, nick names and references to the physician receiving the letter were not detected. On the other hand, our Scrub system found 99-100% of all personally-identifying information.

Method	Other documents	Letters only
Straight search	32%	37%
Search with cues	32%	84%
Scrub(threshold 0.7)	99%	99%
Scrub(threshold 0.5, false positive reduction)	100%	100%

Table 4. Comparisons of Scrub to standard techniques.

DISCUSSION

Reverse scrubbing is the name we give to the ability to identify the real person from scrubbed materials. A person with inside knowledge (e.g., doctor, patient, nurse, or even a friend of the patient) poses the worst threat of reverse scrubbing a database. Any single uniquely occurring identifier can be used to unravel the scrub. We found numerous strategies to combat this problem such as not using any unusual records. Remember "unusual" is not based solely on diagnosis and treatment. It could be a date or some other little detail or combination of details. We used the computer to compute the popularity of data sets, where a data set is one or more fields of information to be scrubbed. The fields were grouped into sets that reflected the information available to the memory of a patient, a doctor, or

some other person. When possible we processed all possible combinations of fields, but this only worked on small numbers since such computing required exponential time. Any record that had a field in a set whose population was below a certain critical level was not a candidate for scrubbing. For dates, we grouped them together so that all dates within a time period were lumped to the same date. For example, lumping dates by week required changing them all to that Monday's date, and similarly for lumping by month. The idea of providing the most general data keeps others from being able to map the scrubbed result to a real person with confidence.

In concluding, we can reliably remove explicit personally-identifying information. Some risks remain since care must be taken to select generally occurring records. Even then however, we still cannot scrub implicit information where an overall sequence of events whose preponderance of details identify a particular individual. This is often the case in mental health data and discharge notes.

Acknowledgments

The author gratefully acknowledges the support of Professor Peter Szolovits at MIT for providing comments and criticisms and an environment that made it possible for me to explore my own ideas. The author also thanks Isaac Kohane, M.D. Ph.D., for the use of his sample database and discussions, Octo Barnett, M.D., for the use of his sample database and Sylvia Barrett for editorial suggestions. This work has been supported by a Medical Informatics Training Grant (1 T15 LM07092) and also in part by grant U01 LM05877-01 both from the National Library of Medicine.

References

1. Kohane IS, Greenspun P, Fackler J, Cimino C, Szolovits P. Building National Electronic Medical Record Systems via the World Wide Web. *Journal of the American Medical Informatics Association* 1996;Volume 3, number 3.
2. Willard KE, Hallgren JH, Sielaff B, Connelly DP. The deployment of a World Wide Web (W3) based medical information system. In: Gardner RM, ed. *Symposium on Computer Applications in Medical Care*. New Orleans, Louisiana: Hanley & Belfus, Inc, 1995:771-775.
3. Jagannathan V, Reddy YV, Srinivas K, et al. An overview of the CERC ARTEMIS Project. In: Gardner RM, ed. *Symposium on Computer Applications in Medical Care*. New Orleans, Louisiana: Hanley & Belfus, Inc., 1995:12-16.
4. Kohane IS. Getting the Data In: Three-Year Experience with a Pediatric Electronic Medical Record System. In: Ozbolt JG, ed. Proceedings, *Symposium on Computer Applications in Medical Care*. Washington, DC: Hanley & Belfus, Inc, 1994:457-461.
5. Barnett GO. The application of computer-based medical-record systems in ambulatory practice. *New England Journal of Medicine*, 1984;310(25): 1643-1650.
6. Erman, L., Hayes-Roth, F., Lesser, V. and Reddy, DR. The Hearsay-II speech-understanding system: integrating knowledge to resolve uncertainty. *Computing Surveys*, 1980;12(2): 213-251.
7. Sweeney, LA. *Speech recognition using real-time phoneme trait detection, the BeBe system.* Massachusetts Institute of Technology, Artificial Intelligence Laboratory: Working paper. 1996
8. Kornfeld, WA. *Using parallel processing for problem solving.* Massachusetts Institute of Technology, Artificial Intelligence Laboratory, 1979; Memo 561.
9. Minsky, M. *The society of mind.* New York: Simon and Schuster. 1985
10. Sweeney, LA. *Multiple hashed binary storage of words -- tiny, fast and almost perfect.* Massachusetts Institute of Technology, AI Laboratory: Working paper. 1996
11. Sweeney, LA. *Automatic acquisition of orthographic rules for recognizing and generating spellings.* Massachusetts Institute of Technology, Artificial Intelligence Laboratory: Working paper. 1996