Parsing Medical Text into De-identified Databases

Veronica Dahl[1,2], Sara Saghaei[2] and Oliver Schulte[2]

[1]Departament de Filologies Romaniques, Universidad Rovira i Virgili
43002 Tarragona, Spain

[2]School of Computing Science, Simon Fraser University
8888 University Drive, Burnaby, Canada

Abstract. De-identification is the process of automatic removal of all Private Health Information (PHI) from medical records. The main focus in this active and important research area is on semi-structured records. This narrow focus has allowed the development of standard criteria that formally determines the boundaries of privacy and can be used for evaluations. However, medical records include, as well as semi-structured data from filling in forms, etc., free text in which identifiers are more difficult to detect. In this article we address the problem of de-identification within unstructured medical records. We show how through the following methods we are able to recognize, in some cases, identifiers that currently go undetected: (1) Parsing free-form medical text into typed logical relationships including assumptions for candidate identifiers. (2) A novel use of the state-of-the-art engines for processing English queries to the web. A formal definition of our approach within a rigorous logical system that supports the implementation of our ideas, is also available on the website[1].

1 Introduction

De-identification is the process of automatic removal of all personally identifying Private Health Information (PHI [2]) from medical records, while preserving the integrity of the data as much as possible [15].

Despite the simple general rule of thumb for de-identification that suggests any piece of information that narrows down the search space to a small amount of identifiable targets should be recognized as a private identifier and removed; in practice, a break down into a list of possible identifiers and an elaborate treatment for each case is necessary. This enforces a seperate procedure as well as exception patterns for each item of PHI. In particular, a special procedure is required for removing dates, another for first names, and so forth. Most of the papers in this area are dedicated to addressing problems in identifying each of the PHI's in detail. A summary of these problems is given in section 2. In the current state of the art, although most of the performance metrics reported in every other paper, hits at least some 90% performance measure, most of

[1] http://www.cs.sfu.ca/ ssaghaei/personal/bilc

[2] Standardized by HIPAA (Health Insurance Portability and Accountability Act) as a list of seventeen categories of possible identifiers

Dahl V., Saghaei S. and Schulte O..
Parsing Medical Text into De-identified Databases.
DOI: 10.5220/0003309700770087
In *Proceedings of the 1st International Workshop on AI Methods for Interdisciplinary Research in Language and Biology* (BILC-2011), pages 77-87
ISBN: 978-989-8425-42-3

the times, several restrictions to the input text and the target output have been assumed that make effective comparisons rather difficult, if not impossible, and furthermore, prevent the elimination of the need for a human assistant to do at least the final scanning, if not re-processing.

In this article we propose a methodology that can be adjusted to the characteristics of input in terms of different types of medical records and takes into account as well the kind of research that is meant to follow the de-identification. The former feature allows us an ingenious hybrid approach that takes advantage of state-of-the-art de-identifiers, mostly addressing the semi-structured inputs. The latter feature facilitates the detection of research-specific types of identifiers, which a one-size-fits-all methodology might miss. Previous related work in this latter respect is [13], which is a sequel to the system developed in [12], allowing modifiable fields for identifiers. While this system is licenced and the code is not available for study, it does not seem that it goes beyond semi-structured inputs. We develop our idea mostly around one specific type of medical record, that of hospital admissions, which exhibits many semi-structured parts (common personal identifiers found in medical forms) as well as some free-text portions (e.g. Observations).

2 The Main Challenges

The problem of de-identification with free-form text can be seen as a conveniently narrower version of the general problem of parsing natural language. It is narrower in the following sense: (1) We can focus to some extent on noun phrases, through which natural language encodes most identifiers. (2) Since the domain of application is known, we can make use of domain-specific lexicons and ontologies in order to determine the meaning from the context. We now discuss the key challenges that arise in de-identification of general texts and more specifically in medical records.

Implicit Identifiers: Sometimes identifiers are not explicit. As an example taken from [10], “the patient’s trailer was blown away by a tornado the night before Christmas” is a statement that does not contain any terms that are outright PHI, but the date is obvious to a human and a news search could potentially reveal details on this newsworthy event and the identity of this special patient.

Ambiguities: PHI and non-PHI can lexically overlap: e.g., Huntington can be the name of a disease (non-PHI) as well as the name of a person (PHI). The ambiguity problem is even more crucial than it looks, since the final output should be suitable for medical research. That’s why in the example above, “Huntington’s disease“” should not be removed, while the name “Huntington” should be. This is a critical and particularly tough problem to deal with.

Privacy Deadlock: This is best described in [15] as a *chicken and egg type of problem:* “systems cannot be effectively developed without access to clinical records, but clinical records cannot be readily made available for research (even for de-identification) without being de-identified.” Although it is not stated in most research how such access to the dataset has been provided, it is rather obvious that it is a somewhat limited one.

Misspellings: PHI can include misspelled and/or foreign words that cannot be found in dictionaries. The most common approach in dealing with misspelled words is through

using a spellchecker. However, it is shown not very helpful in [10] as it gave "little improvement in sensitivity, but a large increase in the number of false positives". Instead, they have been partially able to take care of misspellings by relying on contextual rules; Such that, if an identifier is missed in dictionary lookup, it will be caught in contextual template matching. An idea for dealing with misspelling is to make use of a list of common misspellings as a reference, along with the other dictionaries. Better yet, it might be possible to extract common patterns from that list and feed only those patterns to the system (e.g. receive & recieve, tomorrow and tommorrow, etc.). But if it should be possible to extract heuristics from simple observations, as humans often do, using this approach along with the methods that require an initial learning phase (such as in [14]), this would not only bring great enhancements to the machine's ability in performing de-identification, unrestricted to the detection of mispellings, but also, would eventually, serve as a big step toward the advancement of the Natural Language Understanding (NLU) field.

Re-identification: After identifiers have been recognized, they should be substituted in a generic way that does not violate privacy while maintaining as much of the information as possible. Two best approaches in this regard are: (1) In [10] identifiers are replaced by phrases such as [***first name***], or any other category/subcategory of PHI, which enhances the readability and is needless of further computation, since the category has already been detected as part of the identifier's recognition process. (2) A smoother approach is to substitute them with surrogate information, such as John Doe for a first name.

3 Main General Approaches to De-identification

Most approaches to recognition of PHI are either lexical or contextual:

Lexical. This approach mostly involves using dictionaries and gazeteers and doing string matching type of search throughout the text. Obviously, the larger the dictionaries the better the result. Particularly, in recognition of first names and locations most of the methods rely on dictionaries.

Contextual. This approach can take different forms. It can be as limited as associating a number of common-sense templates with each of PHI categories, as explained in [12], such as a [firstname lastname] template for a person's name, or [lastname, firstname], etc., or as sophistiated as in [11], which applies advanced natural language techniques using a framework called MEDTAG, to categorize words and recognize parts of speech. MEDTAG is a system of tags with an ontology of the medical domain which aims at disambiguation in the "word-sense" level. One example from [11] is the word "miss" which can be taken to mean an action (=fail) or a person (=a young lady). The tagging system considers the context and distinguishes the semantics. Their system also does a parts-of-speech tagging, which would again in the case of "miss" determine whether it's a name or a verb in the sentence to help eliminate potential ambiguities. However, in the word-sense level their work is restricted by the 40 medical tags in the MEDTAG framework and another custom-designed set of "anonymization-specific" tags. This set disambiguates the words taken as possible PHI candidates and was extracted from investigation of particular cases.

To some extent, both these types of systems augment rules with machine learning techniques. Among the systems that use machine learning, those that employ regular expression templates as features perform significantly better than those that do not, from which we can conclude that the gains are not related to machine learning itself but to machine learning conditioned to the use of regular expression templates– a limitation which makes the approach not too suitable for unstructured data such as natural language input.

Results from [15] reveal that the de-identification problem cannot be solved through one single algorithm or approach and that a hybrid approach that combines the lexical and the contextual approaches is necessary to gain better results. However, the scope of application of these two approaches is different too. For example in [11], dictionary check only comes in after the whole text has been tagged and specific zones are marked as "Identity markers". But in many others, the dictionary check comes first. Hybrid systems such as Hara's that employ rules for certain PHI categories and machine learning with regular expression template features for others generally perform worse than the systems that use regular expression template features for all PHI categories [15]. Where contextual resolutions are used along with probability assignments (as in [12] and [14]), they usually yield more accurate results and particularly lower rates of false positivity.

4 An Intuitive Description of Our Proposed Methodology

Our approach is based on concepts and techniques that have been developed in the logic programming community. One of our key innovations is a new representation of identifiers: we represent them as a pair $(term, type)$, where $term$ is a logical expression mapped to a possible identifier in the text, and $type$ represents the semantic type of the term. Thus we model not only whether a part of the text identifies an individual, but also what type of individual it represents. We also use the logic programming concept of *assumption*, which allows us to make temporary guesses as to what part of the text may be denoting which individuals.

4.1 Logic Programming Tools

For processing language, we use a logic grammar [5] together with a taxonomy appropriate to the targeted domain [7], and in particular, two main techniques: assumptions [8] and incomplete types [4]. These were inspired by natural language processing problems, with assumptions having recently proved useful as well for computational molecular biology [9]. We use them both in the traditional way (i.e., for processing language) and in a novel way which allows us to a) keep track of medical identifiers found, and b) re-identify to humans without revealing individuals.

Assumptions can be thought of as globally available, dynamic information that we can hypothesize (technically called "assume") whenever needed (e.g. at the point in the grammar in which we postulate that some noun phrase's referent is an identifier) and is withdrawn either through backtrack or by programming design (if at some point we program their disappearance, technically called "consumption"). We note the assumption of p(X), where X is a vector of n arguments and p an n-ary relation, as +p(X), and

its consumption, as -p(X). Assumptions have become part of the logic programming folklore and are now provided by several logic programming platforms, most notably Hyprolog [3]. Concretely, we use them to (1) keep all potential identifiers we come across until they are either confirmed or discarded as such (through other parts of the analysis confirming them or rejecting them), (2) produce a database representation of the input in which a) every term suspected of being an identifier has been replaced by a surrogate plus its semantic type (e.g., the patient John Smith is now referred to by say, "n1" and the information that n1's semantic type is "patient", so that the new term now looks like "n1-patient"), and b) every term containing an identifier is marked. (3) use these surrogates to consult the information at hand and (if warranted) the world wide web in order to try and find the original identifier. If found, further purge the information that allowed the system to reconstruct the identifier until it can no longer be reconstructed.

In the next section we examine each of these uses in greater detail. Incomplete types were originally developed for static databases, in order to provide semantic compatibility checkups at parsing time, for savings at database consultation time [4]. Section 5.4 describes them informally. A formal logical respresentation of our approach incorporating these incomplete types is also devised and is available on our website.

4.2 Main Components of Our Approach

Extracting Identifiers and their Semantic Types. *Within Noun Phrases:* Proper names are considered by our grammar as inherently identifying, so they are marked as such by the unary function "id", whose presence as an argument of a relation tuple will induce the tuple's recording as an assumption. A noun is interpreted by our parser as the relational symbol of a semantic relationship to be extracted and the arguments of this relation are constructed from the noun's various complements, appropriately typed after consultation of a domain-dependent ontology. E.g., the noun phrase: "The activation of NF-kappa-B via CD-28" parses into:

```
+activation(protein-id('NF-kappa-B'), gene-id('CD-28')).
```

The relationship, the types associated with its arguments by our concept hierarchy (protein and gene) and the fact that the arguments are 'identifiers' to be preserved, and therefore assumed rather than entered in the database, are visible in this encoding.
Within verb phrases. Verbs also induce relationships whose arguments are the semantic representations of the verb's syntactic arguments. E.g., in the sentence: "Retinoblastoma proteins negatively regulate transcriptional activation", the verb *regulate* marks a relation between two concepts – *retinoblastoma proteins* and *transcriptional activation*. In this case, our parser does not mark the relationship's arguments as identifiers, since they are induced from nouns rather than from proper names. Finer distinctions can be made in the grammar taking into account the specific domain of application, but for our exemplifying purposes the following representation suffices:

```
regulate(retinoB-protein, transActivation-process).
```

Representing the Input as a (Fairly) De-identified Database. As a result of our parsing process, a database is created which contains predicate definitions, plus assumptions representing the information in the input that contains terms marked as identifiers. In this first phase of our approach, we use a subset of natural language as the allowed input language. However this subset is both natural and correct and usually sufficient.

Fine-tuning the De-identification and Erasing All Traces of Identifiers. We now query other available sources in order to find further possible identifiers that do not directly result from say, lexical mandate, but from the meaning of the sentences. For instance, if a phrase such as "the patient's trailer was blown away by a tornado the night before Christmas" produces, as it will, no concrete referents upon being translated into typed logic, the system will invoke state-of-the-art natural language based web mining systems (e.g. Hakia[3], Powerset[4], Watson[5]), with the question: "Whose trailer was blown away by a tornado the night before Christmas?" If a concrete identifier is produced as an answer, the entire phrase will be replaced by a surrogate phrase (possibly empty). Notice by the way that even if the sentence were to fall outside our natural language subset, we could still transform it into the relevant question to be given to web consultation, thereby extending our subset effectively beyond that formally treated within our grammar.

Once all identifiers possible have been found and substituted, all remaining assumptions whose id's have been replaced by surrogates are firmed into proper relations. At the end of this process, any remaining assumptions indicate the presence of identifiers that could not be replaced (this would be the case for instance if a human could find in a newspaper the identity of the patient whose trailer was blown away by a tornado the night before Christmas, yet none of the state-of-the art web mining systems could), and can be thus erased in order to leave no trace of information that the machine could not de-identify but a human might.

5 Solving Common De-identification Problems

5.1 Ambiguity

Semantic types are not only useful for re-identification, but, can also serve to convey the appropriate meaning of an ambiguous word or phrase. Because our domain is known, we can augment terms with domain-dependent type hierarchy information and include them directly in the grammar's lexicon. By doing so, a quick semantic compatibility check, performed as a side effect of normal unification, will ensure disambiguation on the fly.

Example. In the hospital admission records terminology, if we allow "enter" as a synonym for "admitted", there will be at least two lexical entries for that verb, exemplified as:

[3] Hakia homepage, http://company.hakia.com/new

[4] Powerset homepage, http://www.crunchbase.com/company/powerset

[5] Watson homepage, http://kmi-web05.open.ac.uk/WatsonWUI/

```
enter(patient-X,hospital-Y).
enter(patient-X,state-Y)  (as in ‘‘entered into a comma")
```

As a second example, consider the word huntington in the two sentences: “Huntington entered the hospital on April 16, 2010.”, and “Smith should be tested for Huntington.” and their corresponding disambiguated typed logic representations:

```
entered(patient-huntington,hospital-X,date-16-04-2010).
must(test-for(patient-smith,disease-huntington)).
```

As a more complex example, consider the use of the word *binding site* in the biomedical domain; It usually refers to a *DNA* domain or region, but exceptionally to a *protein* domain or region too. Catching the latter meaning is not trivial, since both *c-Myc* and *G28-5* are protein molecules. However, our parser looks for semantic clues from surrounding words in order to disambiguate: in sentence (1) in Table 1, *promoters* points to the DNA region binding site, whereas in sentence (2), *ligands* points to the protein meaning of binding site. Our parser can calculate an entity’s appropriate type by consulting domain-specific clues.

Table 1. “binding site” in two different contexts.

(1) Transcription factors USF1 and USF2 up-regulate gene expression via interaction with an E box on their target **promoters**, which is also a **binding site** for **c-Myc**.
(2) The functional activity of **ligands** built from the **binding site** of **G28-5** is dependent on the size and physical properties of the molecule both in solution and on the cell surfaces.

5.2 Anaphoric References

Our parser keeps track of potential referents for pronouns or other referential terms, also through the use of assumptions. This technique has already been described for natural language analysis in general, using timeless assumptions in order to allow for the original referent to appear in preceding or following sentences of the referential term (in technical terms, we can treat both forward and backward anaphora). Interested readers are referred to [3] and [8] for details.

Note that disambiguation and anaphora resolution can cooperate with each other: Semantic types allow us to differentiate between a patient named Huntington and a disease named so; thus, further ensure the correct identification of a referent, as the following discourse and corresponding representations exemplify.

“Huntington entered the hospital on April 16, 2010. This patient should be tested for Huntington.”

```
+entered(patient-id(huntington), hospital-id(universalcures),
date-id(16-04-2010)).
must-test-for(patient-P,disease-huntington)
```

Our parser’s anaphora resolution system will instantiate P with id(huntington) and correspondingly mark the relation “must-test-for” as an assumption. The explicit mentioning of the type (“patient”) in the subject of the second sentence serves as a corroboration

to the anaphora resolution system that we are referring indeed to the Huntington typed as a patient, in the first sentence. If it was marked otherwise, the two types would not have matched. If the second sentence was “He should be tested for Huntington”, the type gleaned from the first sentence for this individual would simply carry over, together with his name, into the term representing it. Of course, even for humans there will be cases in which even context leaves us clueless, as in “Huntington won”. We are content if our proposed methodology allows us to deal with ambiguity with at least as much success as humans can.

5.3 Domain-Dependent Semantic Constraints

Some sentences that are syntactically correct should still be ruled out because they do not make sense in the domain of application. For example, in our biomedical domain, it would not make sense to have a term of the form ‘regulation of protein name’, e.g. regulation of actin, because a protein is neither a process nor a function. Our system codes such semantic constraints in terms of selectional restrictions on the relations induced by grammar analysis, e.g. “regulation” induces a relationship of same name whose argument must be typed by either “process” or “function”. Selectional restrictions are implemented through incomplete types.

5.4 Contextual Semantic Interpretation through Incomplete Types

Consider the query: *does TB Meningitis need antibiotics?*, given a database where TB Meningitis is a type of Meningitis, Meningitis is an infectious disease, and infectious diseases need antibiotics for treatment. In Prolog, a positive answer to the query:

```
?- needsAntibiotics(tbMeningitis).
```

can be obtained in three resolution steps from the database:

```
needsAntiobiotics(D):- infectiousDisease(D).
infectiousDisease(D) :- meningitis(D).
meningitis(tbMeningitis).
```

Instead, we propose to use typed Horn-clause logic: the given information would be represented as in equations (1), below. A compiler would tranform them into the form (2), so that eventually, we can obtain the same property inheritance in just one resolution step as done in (3).

$$\begin{gathered} needsAntibiotics(D \in infectiousDisease). \\ meningitis \subset infectiousDisease. \\ tbMeningitis \in meningitis. \\ ? - needsAntibiotics(tbMeningitis). \end{gathered} \tag{1}$$

$$needsAntibiotics(D \in [infectiousDisease \supset Y]). \tag{2}$$

$$\begin{gathered} ? - needsAntiobiotics(tbMeningitis \in \\ [infectiousDisease \supset meningitis \supset tbMeningitis]). \end{gathered} \tag{3}$$

Types have been replaced by a representation of their relevant set inclusion relationships. It is *incomplete* in that it contains a tail variable which allows for further instantiation. Thus, D in equation 2 is of "at least an infectious disease" type. Since the representation of constants should not allow for further instantiation, the constant itself closes the representation, as in the query in equation 3 (we abusively keep the inclusion sign for uniformity of notation). Resolving 2 with 3 unifies Y with infectiousDisease $\supset$ meningitis, thus making the type further known as *both* infectiousDisease and meningitis. For strictly hierarchical taxonomies, we have thus reduced type checking from a chain of n set inclusion relationships to *one* resolution steps.

5.5 Re-identification

In our approach, which routinely adds semantic types to all terms, it is easy to replace a name with a surrogate and augment it with the appropriate semantic type, to further improve readability and reduce ambiguity.

6 Discussion

We have proposed a methodology for de-identification that can be tailored to the characteristics of specific types of medical records, e.g. by consulting appropriate domain taxonomies. Our proposal deals with privacy sensitive texts that are rich sources of research information by extracting the knowledge these texts represent and feeding them into a database, and by marking any sensitive fields for eventual removal by the system. Thus, researchers, instead of accessing the text, can query the database for the information they require. Private fields can either be disguised under surrogate names while maintaining knowledge of their semantic type, or protected by the system from being queried; for example, if the researcher wants to know how many of the patients with some kind of disease smoke, the answer is a number, which wouldn't threaten anybody's privacy. But if they query the information about the name of the person whose medical condition is such and such, they would receive an error.

Our approach combines the lexical and the contextual approaches in a novel way, in that it turns text into typed logical relations that express exactly what was said. This is achieved through a parser which benefits from lexical, syntactic and semantic processing of information. Since it addresses restricted but natural language, as opposed to semi-structured data, it is hard to evaluate with respect to existing systems which one is more restricted. However, once the parser, presently under construction, is finished, we will at least test it for statistics regarding how it improves the de-identification task of medical records with free text. We have not directly used machine learning since for parsing the fairly substantial subset of natural language that we allow, it is not easily profitable; however, we use it indirectly through consulting the other approaches on the semi-structured portions of input. On the other hand, our proposed methodologies for disambiguation and semantic type compatibility enforcement could be profitably transfered into other approaches too: given that they are useful for a bigger subset of language than they attack, they are sure to be effective also on semi-structured data. Robustness is an issue we have not yet addressed: our system for the time being will

not work if there are grammatical errors or misspelling. Future work includes a pre-processor which will consult the web for possible misspelling, and will work bottom-up to test for grammaticality before invoking the parser. In this respect previous work using constraints has proved promising [2], [6]. Google suggestions for correcting a misspelled word, as it amounts, in fact, to a summary of common knowledge shared among users of the internet, is something to consider. for example, if the word “small” is misspelled as “smll”, using only dictionary matching, it might be mapped to any number of possible permutations of missing letter(s). But if queried into google, the third suggestion is the word “small”, which in its own way, shows the common sense knowledge of humans when faced with the word “smll”, and the two first suggestions are not valid dictionary entries.

Note that, because in our approach a typed database is created from text, we can define specific ways of querying it according to how the researchers want to use it. For instance, if the researchers are interested in general answers, specialized evaluation primitives will aim at intensional kinds of answers. For instance, given a rule such as “Diabetics benefit from exercise”, and the query

“Who benefits from exercise?”, an intensional evaluation will produce the reply, “Diabetic people”, rather than a list of individuals’ names. Eventually, we can even consider mining the re-identified databases to obtain statistics allowing us to extract further knowledge than the one present in the records per se, e.g. the percentage of diabetics that benefit from how much exercise, etc. With this work we hope to contribute to an entirely novel and fruitful way of rethinking the problem of de-identification, and to stimulate further research along these lines.

Acknowledgements

We are very grateful for the anonymous reviewers’ comments on a first draft of this article. This work was supported by the European Commission in the form of V. Dahl′s Marie Curie Chair of Excellence and by two Discovery Grants from the Natural Sciences and Engineering Research Council of Canada (NSERC) to Dahl and Schulte each.

References

1. Berman, J. (2003). Concept-match medical data scrubbing. how pathology text can be used in research. In Arch Pathol Lab Med, volume 127(6), 680-686.
2. Christiansen, H. and Dahl, V. (2003). Logic grammars for diagnosis and repair. *International Journal on Artificial Intelligence Tools*, 12(3):227–248.
3. Christiansen, H. and Dahl, V. (2005). Hyprolog: a new logic programming language with assumptions and abduction. In *International Conference on Logic Programming (ICLP)*.
4. Dahl, V. (1991). Incomplete types for logic databases. Applied Mathematics Letters, 4(3):25–28.
5. Abramson, H. and Dahl, V. (1989) Logic Grammars. Computation AI Series, Springer-Verlag, 1–234.
6. Dahl, V. and Blache, P. (2005). Extracting selected phrases through constraint satisfaction. In 2nd Intl. Workshop on Constraint Solving and Language Processing.

7. Dahl, V. and Gu, B. (2008). On semantically constrained property grammars. In Constraints and Language Processing (CSLP), pages 20–32.
8. Dahl, V. and Tarau, P. (2004). Assumptive logic programming. In Argentine Symposium on Artificial Intelligence.
9. Enguix, G. B., Dahl, V., and Jiménez-López, M. D. (2009). DNA and natural languages - text mining. In KDIR, pages 140–145.
10. Neamatullah, I., Douglass, M., Lehman, L., Reisner, A., Villarroel, M., Long, W., Szolovits, P., Moody, G., Mark, R., and Clifford, G. (2008). Automated de-identification of free-text medical records. BMC Medical Informatics and Decision Making, 8(32).
11. Ruch, P., Baud, R., Rassinoux, A., Bouillon, P., and Robert., G. (2000). Medical document anonymization with a semantic lexicon. In Proceedings of the AMIA Symposium, page 729. American Medical Informatics Association.
12. Sweeney, L. (1996). Replacing personally-identifying information in medical records, the scrub system. In Proceedings of the AMIA Annual Fall Symposium, pages 333–7. American Medical Informatics Association.
13. Sweeney, L. (1997). Guaranteeing anonymity when sharing medical data, the datafly system. In Proceedings of the AMIA Annual Fall Symposium, pages 51–5. American Medical Informatics Association.
14. Taira, R., Bui, A., and Kangarloo, H. (2002). Identification of patient name references within medical documents using semantic selectional restrictions. In *Proceedings of the AMIA Symposium*, pages 757–61. American Medical Informatics Association.
15. Uzuner, O., Luo, Y., and Szolovits, P. (2007). Evaluating the state-of-the-art in automatic de-identification. Journal of the American Medical Informatics Association, 14(5):550–63.