Automatic Code Assignment to Medical Text

Koby Crammer and **Mark Dredze** and **Kuzman Ganchev** and **Partha Pratim Talukdar**
Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA
{crammer|mdredze|kuzman|partha}@seas.upenn.edu

Steven Carroll
Division of Oncology, The Children's Hospital of Philadelphia, Philadelphia, PA
carroll@genome.chop.edu

Abstract

Code assignment is important for handling large amounts of electronic medical data in the modern hospital. However, only expert annotators with extensive training can assign codes. We present a system for the assignment of ICD-9-CM clinical codes to free text radiology reports. Our system assigns a code configuration, predicting one or more codes for each document. We combine three coding systems into a single learning system for higher accuracy. We compare our system on a real world medical dataset with both human annotators and other automated systems, achieving nearly the maximum score on the Computational Medicine Center's challenge.

1 Introduction

The modern hospital generates tremendous amounts of data: medical records, lab reports, doctor notes, and numerous other sources of information. As hospitals move towards fully electronic record keeping, the volume of this data only increases. While many medical systems encourage the use of structured information, including assigning standardized codes, most medical data, and often times the most important information, is stored as unstructured text.

This daunting amount of medical text creates exciting opportunities for applications of learning methods, such as search, document classification, data mining, information extraction, and relation extraction (Shortliffe and Cimino, 2006). These applications have the potential for considerable benefit to the medical community as they can leverage information collected by hospitals and provide incentives for electronic record storage. Much of the data generated by medical personnel is unused past the clinical visit, often times because there is no way to simply and quickly apply the wealth of information. Medical NLP holds the promise of both greater care for individual patients and enhanced knowledge about health care.

In this work we explore the assignment of ICD-9-CM codes to clinical reports. We focus on this practical problem since it is representative of the type of task faced by medical personnel on a daily basis. Many hospitals organize and code documents for later retrieval using different coding standards. Often times, these standards are extremely complex and only trained expert coders can properly perform the task, making the process of coding documents both expensive and unreliable since a coder must select from thousands of codes a small number for a given report. An accurate automated system would reduce costs, simplify the task for coders, and create a greater consensus and standardization of hospital data.

This paper addresses some of the challenges associated with ICD-9-CM code assignment to clinical free text, as well as general issues facing applications of NLP to medical text. We present our automated system for code assignment developed for the Computational Medicine Center's challenge. Our approach uses several classification systems, each with the goal of predicting the exact code configuration for a medical report. We then use a learning system to combine our predictions for superior performance.

This paper is organized as follows. First, we explain our task and difficulties in detail. Next we describe our three automated systems and features. We combine the three approaches to create a single superior system. We evaluate our system on clinical reports and show accuracy approaching human performance and the challenge's best score.

2 Task Overview

The health care system employs a large number of categorization and classification systems to assist data management for a variety of tasks, including patient care, record storage and retrieval, statistical analysis, insurance, and billing. One of these systems is the International Classification of Diseases, Ninth Revision, Clinical Modification (ICD-9-CM) which is the official system of assigning codes to diagnoses and procedures associated with hospital utilization in the United States. [1] The coding system is based on World Health Organization guidelines. An ICD-9-CM code indicates a classification of a disease, symptom, procedure, injury, or information from the personal history. Codes are organized hierarchically, where top level entries are general groupings (e.g. "diseases of the respiratory system") and bottom level codes indicate specific symptoms or diseases and their location (e.g. "pneumonia in aspergillosis"). Each specific, low-level code consists of 4 or 5 digits, with a decimal after the third. Higher level codes typically include only 3 digits. Overall, there are thousands of codes that cover a broad range of medical conditions.

Codes are assigned to medical reports by doctors, nurses and other trained experts based on complex coding guidelines (National Center for Health Statistics, 2006). A particular medical report can be assigned any number of relevant codes. For example, if a patient exhibits a cough, fever and wheezing, all three codes should be assigned. In addition to finding appropriate codes for each condition, complex rules guide code assignment. For example, a diagnosis code should always be assigned if a diagnosis is reached, a diagnosis code should never be assigned when the diagnosis is unclear, a symptom should never be assigned when a diagnosis is present, and the most specific code is preferred. This means that codes that seem appropriate to a report should be omitted in specific cases. For example, a patient with hallucinations should be coded 780.1 (hallucinations) but for visual hallucinations, the correct code is 368.16. The large number of codes and complexity of assignment rules make this a difficult problem for humans (inter-annotator agreement is low). Therefore, an automated system that suggested or assigned codes could make medical data more consistent.

These complexities make the problem difficult for NLP systems. Consider the task as multi-class, multi-label. For a given document, many codes may seem appropriate but it may not be clear to the algorithm how many to assign. Furthermore, the codes are not independent and different labels can interact to either increase or decrease the likelihood of the other. Consider a report that says, "patient reports cough and fever." The presence of the words cough and fever indicate codes 786.2 (cough) and 780.6 (fever). However, if the report continues to state that "patient has pneumonia" then these codes are dropped in favor of 486 (pneumonia). Furthermore, if the report then says "verify clinically", then the diagnosis is uncertain and only codes 786.2 and 780.6 apply. Clearly, this is a challenging problem, especially for an automated system.

2.1 Corpus

We built and evaluated our system in accordance with the Computational Medicine Center's (CMC) 2007 Medical Natural Language Processing Challenge.[2] Since release of medical data must strictly follow HIPAA standards, the challenge corpus underwent extensive treatment for disambiguation, anonymization, and careful scrubbing. A detailed description of data preparation is found in Computational Medicine Center (2007). We describe the corpus here to provide context for our task.

The training corpus is comprised of 978 radiological reports taken from real medical records. A test corpus contains 976 unlabeled documents. Radiology reports have two text fields, clinical history and

[1] `http://www.cdc.gov/nchs/about/otheract/icd9/abticd9.htm`

[2] `www.computationalmedicine.org/challenge`

impression. The physician ordering the x-ray writes the clinical history, which contains patient information for the radiologist, including history and current symptoms. Sometimes a guess as to the diagnosis appears ("evaluate for asthma"). The descriptions are sometimes whole sentences and other times single words ("cough"). The radiologist writes the impression to summarize his or her findings. It contains a short analysis and often times a best guess as to the diagnosis. At times this field is terse, ("pneumonia" or "normal kidneys") and at others it contains an entire paragraph of text. Together, these two fields are used to assign ICD-9-CM codes, which justify a certain procedure, possibly for reimbursement by the insurance company.

Only a small percentage of ICD-9-CM codes appear in the challenge. In total, the reports include 45 different codes arranged in 94 configurations (combinations). Some of these codes appear frequently, while others are rare, appearing only a single time. The test set is restricted so that each configuration appears at least once in the training set, although there is no further guarantee as to the test set's distribution over codes. Therefore, in addition to a large number of codes, there is variability in the amount of data for each code. Four codes have over 100 examples each and 24 codes have 10 or fewer documents, with 10 of these codes having only a single document.

Since code annotation is a difficult task, each document in the corpus was evaluated by three expert annotators. A gold annotation was created by taking the majority of the annotators; if two of the three annotators provided a code, that code is used in the gold configuration. This approach means that a document's configuration may be a construction of multiple annotators and may not match any of the three annotators exactly. Both the individual and the majority annotations are included with the training corpus.

While others have attempted ICD-9 code classification, our task differs in two respects (Section 7 provides an overview of previous work). First, previous work has used discharge reports, which are typically longer with more text fields. Second, while most systems are evaluated as a recommendation system, offering the top k codes and then scoring recall at k, our task is to provide the *exact* configuration. The CMC challenge evaluated systems using an F1 score, so we are penalized if we suggest any label that does not appear in the majority annotation.

To estimate task difficulty we measured the inter-annotator score for the training set using the three annotations provided. We scored two annotations with the micro average F1, which weighs each code assignment equally (see Section 5 for details on evaluation metrics). If an annotator omitted a code and included an extra code, he or she is penalized with a false positive (omitting a code) and a false negative (adding an extra code). We measured annotators against each other; the average f-measure was 74.85 (standard deviation of .06). These scores were low since annotators chose from an unrestricted set of codes, many of which were not included in the final majority annotation. However, these scores still indicate the human accuracy for this task using an unrestricted label set. [3]

3 Code Assignment System

We developed three automated systems guided by our above analysis. First, we designed a learning system that used natural language features from the official code descriptions and the text of each report. It is general purpose and labels all 45 codes and 94 configurations (labels). Second, we built a rule based system that assigned codes based on the overlap between the reports and code descriptions, similar to how an annotator may search code descriptions for appropriate labels. Finally, a specialized system aimed at the most common codes implemented a policy that mimics the guidelines a medical staffer would use to assign these codes.

3.1 Learning System

We begin with some notational definitions. In what follows, x denotes the generic input document (radiology report), Y denotes the set of possible labelings (code configurations) of x, and $y^*(x)$ the correct labeling of x. For each pair of document x and labeling $y \in Y$, we compute a vector-valued feature representation $f(x, y)$. A linear model is

[3]We also measured each annotator with the majority codes, taking the average score (87.48), and the best annotator with the majority label (92.8). However, these numbers are highly biased since the annotator influences the majority labeling. We observe that our final system still exceeds the average score.

given by a weight vector w. Given this weight vector w, the score $w \cdot f(x, y)$ ranks possible labelings of x, and we denote by $Y_{k,w}(x)$ the set of k top scoring labelings for x. For some structured problems, a factorization of $f(x, y)$ is required to enable a dynamic program for inference. For our problem, we know all the possible configurations in advance (there are 94 of them) so we can pick the highest scoring $y \in Y$ by trying them all. For each document x and possible labeling y, we compute a score using w and the feature representation $f(x, y)$. The top scoring y is output as the correct label. Section 3.1.1 describes our feature function $f(x, y)$ while Section 3.1.2 describes how we find a good weight vector w.

3.1.1 Features

Problem representation is one of the most important aspects of a learning system. In our case, this is defined by the set of features $f(x, y)$. Ideally we would like a linear combination of our features to exactly specify the true labeling of all the instances, but we want to have a small total number of features so that we can accurately estimate their values. We separate our features into two classes: label specific features and transfer features. For simplicity, we index features by their name. Label specific features are only present for a single label. For example, a simple class of label specific features is the conjunction of a word in the document with an ICD-9-CM code in the label. Thus, for each word we create 94 features, i.e. the word conjoined with every label. These features tend to be very powerful, since weights for them can encode very specific information about the way doctors talk about a disease, such as the feature "contains word pneumonia and label contains code 486". Unfortunately, the cost of this power is that there are a large number of these features, making parameter estimation difficult for rare labels. In contrast, transfer features can be present in multiple labels. An example of a transfer feature might be "the impression contains all the words in the code descriptions of the codes in this label". Transfer features allow us to generalize from one label to another by learning things like "if all the words of the label description occur in the impression, then this label is likely" but have the drawback that we cannot learn specific details about common labels. For example, we cannot learn that the word "pneumonia" in the impression is negatively correlated with the code cough. The inclusion of both label specific and transfer features allows us to learn specificity where we have a large number of examples and generality for rare codes.

Before feature extraction we normalized the reports' text by converting it to lower case and by replacing all numbers (and digit sequences) with a single token $\langle NUM \rangle$. We also prepared a synonym dictionary for a subset of the tokens and n-grams present in the training data. The synonym dictionary was based on MeSH[4], the Medical Subject Headings vocabulary, in which synonyms are listed as terms under the same concept. All ngrams and tokens in the training data which had mappings defined in the synonym dictionary were then replaced by their normalized token; e.g. all mentions of "nocturnal enuresis" or "nighttime urinary incontinence" were replaced by the token "bedwetting". Additionally, we constructed descriptions for each code automatically from the official ICD-9-CM code descriptions in National Center for Health Statistics (2006). We also created a mapping between code and code type (diagnosis or symptom) using the guidelines.

Our system used the following features. The descriptions of particular features are in quotes, while schemes for constructing features are not.

- "this configuration contains a disease code", "this configuration contains a symptom code", "this configuration contains an ambiguous code" and "this configuration contains both disease and symptom codes".[5]
- With the exception of stop-words, all words of the impression and history conjoined with each label in the configuration; pairs of words conjoined with each label; words conjoined with pairs of labels. For example, "the impression contains 'pneumonia' and the label contains codes 786.2 and 780.6".
- A feature indicating when the history or impression contains a complete code description

[4]`www.nlm.nih.gov/mesh`

[5]We included a feature for configurations that had both disease and symptom codes because they appeared in the training data, even though coding guidelines prohibit these configurations.

for the label; one for a word in common with the code description for one of the codes in the label; a common word conjoined with the presence of a negation word nearby ("no", "not", etc.); a word in common with a code description not present in the label. We applied similar features using negative words associated with each code.

- A feature indicating when a soft negation word appears in the text ("probable", "possible", "suspected", etc.) conjoined with words that follow; the token length of a text field ("impression length=3"); a conjunction of a feature indicating a short text field with the words in the field ("impression length=1 and 'pneumonia' ")

- A feature indicating each n-gram sequence that appears in both the impression and clinical history; the conjunction of certain terms where one appears in the history and the other in the impression (e.g. "cough in history and pneumonia in impression").

3.1.2 Learning Technique

Using these feature representations, we now learn a weight vector w that scores the correct labelings of the data higher than incorrect labelings. We used a k-best version of the MIRA algorithm (Crammer, 2004; McDonald et al., 2005). MIRA is an online learning algorithm that for each training document x updates the weight vector w according to the rule:

$$\begin{aligned} w_{\text{new}} &= \arg\min_{w} \|w - w_{\text{old}}\| \\ &\text{s.t.} \quad \forall y \in Y_{k,w_{old}}(x) : \\ w \cdot f(x, y^*(x)) &- w \cdot f(x,y) \geq L(y^*(x), y) \end{aligned}$$

where $L(y^*(x), y)$ is a measure of the loss of labeling y with respect to the correct labeling $y^*(x)$. For our experiments, we set k to 30 and iterated over the training data 10 times. Two standard modifications to this approach also helped. First, rather than using just the final weight vector, we average all weight vectors. This has a smoothing effect that improves performance on most problems. The second modification is the introduction of slack variables:

$$\begin{aligned} w_{\text{new}} &= \arg\min_{w} \|w - w_{\text{old}}\| + \gamma \sum_i \xi_i \\ &\text{s.t.} \quad \forall y \in Y_{k,w_{old}}(x) : \\ w \cdot f(x, y^*(x)) &- w \cdot f(x,y) \geq L(y^*(x), y) - \xi_i \\ &\forall i \in \{1 \ldots k\} : \quad \xi_i \geq 0. \end{aligned}$$

We used a γ of 10^{-3} in our experiments.

The most straightforward loss function is the 0/1 loss, which is one if y does not equal $y^*(x)$ and zero otherwise. Since we are evaluated based on the number of false negative and false positive ICD-9-CM codes assigned to all the documents, we used a loss that is the sum of the number of false positive and the number of false negative labels that y assigns with respect to $y^*(x)$.

Finally, we only used features that were possible for some labeling of the test data by using only the test data to construct our feature alphabet. This forced the learner to focus on hypotheses that could be used at test time and resulted in a 1% increase in F-measure in our final system on the test data.

3.2 Rule Based System

Since some of the configurations appear a small number of times in our corpus (some only once), we built a rule based system that requires no training. The system uses a description of the ICD-9-CM codes and their types, similar to the list used by our learning system (Section 3.1.1). The code descriptions include between one and four short descriptions, such as "reactive airway disease", "asthma", and "chronic obstructive pulmonary disease". We treat each of these descriptions as a bag of words. For a given report, the system parses both the clinical history and impression into sentences, using "." as a sentence divider. Each sentence is the checked to see if all of the words in a code description appear in the sentence. If a match is found, we set a flag corresponding to the code. However, if the code is a disease, we search for a negation word in the sentence, removing the flag if a negation word is found. Once all code descriptions have been evaluated, we check if there are any flags set for disease codes. If so, we remove all symptom code flags. We then emit a code corresponding to each set flag. This simple system does not enforce configuration restrictions;

we may predict a code configuration that does not appear in our training data. Adding this restriction improved precision but hurt recall, leading to a slight decrease in F1 score. We therefore omitted the restriction from our system.

3.3 Automatic Coding Policies

As we described in Section 2, enforcing coding guidelines can be a complex task. While a learning system may have trouble coding a document, a human may be able to define a simple policy for coding. Since some of the most frequent codes in our dataset have this property, we decided to implement such an automatic coding policy. We selected two related sets of codes to target with a rule based system, a set of codes found in pneumonia reports and a set for urinary tract infection/reflux reports.

Reports related to pneumonia are the most common in our dataset and include codes for pneumonia, asthma, fever, cough and wheezing; we handle them with a single policy. Our policy is as follows:

- Search for a small set of keywords (e.g. “cough”, “fever”) to determine if a code should be applied.
- If “pneumonia” appears unnegated in the impression and the impression is short, or if it occurs in the clinical history and is not preceded by phrases such as “evaluate for” or “history of”, apply pneumonia code and stop.
- Use the same rule to code asthma by looking for “asthma” or “reactive airway disease”.
- If no diagnosis is found, code all non-negated symptoms (cough, fever, wheezing).

We selected 80% of the training set to evaluate in the construction of our rules. We then ran the finished system on both this training set and the held out 20% of the data. The system achieved F1 scores of 87% on the training set and 84% on the held out data for these five codes. The comparable scores indicates that we did not over-fit the training data.

We designed a similar policy for two other related codes, urinary tract infection and vesicoureteral reflux. We found these codes to be more complex as they included a wide range of kidney disorders. On these two codes, our system achieved 78% on the train set and 76% on the held out data. Overall, automatically applying our two policies yielded high confidence predictions for a significant subset of the corpus.

4 Combined System

Since our three systems take complimentary approaches to the problem, we combined them to improve performance. First, we took our automatic policy and rule based systems and cascaded them; if the automatic policy system does not apply a code, the rule based system classifies the report. We used a cascaded approach since the automatic policy system was very accurate when it was able to assign a code. Therefore, the rule based system defers to the policy system when it is triggered. Next, we included the prediction of the cascaded system as a feature for our learning system. We used two feature rules: “cascaded-system predicted exactly this label” and “cascaded-system predicted one of the codes in this label”. As we show, this yielded our most accurate system. While we could have used a meta-classifier to combine the three systems, including the rule based systems as features to the learning system allowed it to learn the appropriate weights for the rule based predictions.

5 Evaluation Metric

Evaluation metrics for this task are often based on recommendation systems, where the system returns a list of the top k codes for selection by the user. As a result, typical metrics are “recall at k” and average precision (Larkey and Croft, 1995). Instead, our goal was to predict the exact configuration, returning exactly the number of codes predicted to be on the report. The competition used a micro-averaged F1 score to evaluate predictions. A contingency table (confusion matrix) is computed by summing over each predicted code for each document by prediction type (true positive, false positive, false negative) weighing each code assignment equally. F1 score is computed based on the resultant table. If specific codes or under-coding is favored, we can modify our learning loss function as described in Section 3.1.2. A detailed treatment of this evaluation metric can be found in Computational Medicine Center (2007).

System	*Precision*	*Recall*	*F1*
BL	61.86	72.58	66.79
RULE	81.9	82.0	82.0
CASCADE	86.04	84.56	85.3
LEARN	85.5	83.6	84.6
CASCADE+LEARN	87.1	85.9	86.5

Table 1: Performance of our systems on the provided labeled training data (F1 score). The learning systems (*CASCADE+LEARN* and *LEARN*) were evaluated on ten random split of the data while *RULE* was evaluated on all of the training data. We include a simple rule based system (*BL*) as a baseline.

6 Results

We evaluated our systems on the labeled training data of 978 radiology reports. For each report, each system predicted an exact configuration of codes (i.e. one of 94 possible labels). We score each system using a micro-averaged F1 score. Since we only had labels for the training data, we divided the data using an 80/20 training test split and averaged results over 10 runs for our learning systems. We evaluated the following systems:

- *RULE* : The rule based system based on ICD-9-CM code descriptions (Section 3.2).
- *CASCADE* : The automatic code policy system (Section 3.3) cascaded with *RULE* (Section 4).
- *LEARN* : The learning system with both label specific and transfer features (Section 3.1).
- *CASCADE+LEARN* : Our combined system that incorporates *CASCADE* predictions as a feature to *LEARN* (Section 4).

For a baseline, we built a simple system that applies the official ICD-9-CM code descriptions to find the correct labels (*BL*). For each code in the training set, the system generates text-segments related to it. During testing, for each new document, the system checks if any text-segment (as discovered during training) appears in the document. If so, the corresponding code is predicted. The results from our four systems and baseline are shown in Table 1.

System	*Train*	*Test*
CASCADE	85.3	84
CASCADE+LEARN	86.5	87.60
Average	-	76.6
Best	-	89.08

Table 2: Performance of two systems on the train and test data. Results obtained from the web submission interface were rounded. *Average* and *Best* are the average and best f-measures of the 44 submitted systems (standard deviation 13.40).

Each of our systems easily beats the baseline, and the average inter-annotator score for this task. Additionally, we were able to evaluate two of our systems on the test data using a web interface as provided by the competition. The test set contains 976 documents (about the same as the training set) and is drawn the from same distribution as the training data. Our test results were comparable to performance on the training data, showing that we did not over-fit to the training data (Table 2). Additionally, our combined system (*CASCADE+LEARN*) achieved a score of 87.60%, beating our training data performance and exceeding the average inter-annotator score. Out of 44 submitted systems, the average score on test data was 76.7% (standard deviation of 13.40) and the maximum score was 89.08%. Our system scored 4th overall and was less than 1.5% behind the best system. Overall, in comparison with our baselines and over 40 systems, we perform very well on this task.

7 Related Work

There have been several attempts at ICD-9-CM code classification and related problems for medical records. The specific problem of ICD-9-CM code assignment was studied by Lussier et al. (2000) through an exploratory study. Larkey and Croft (1995) designed classifiers for the automatic assignment of ICD-9 codes to discharge summaries. Discharge summaries tend to be considerably longer than our data and contain multiple text fields. Additionally, the number of codes per document has a larger range, varying between 1 and 15 codes. Larkey and Croft use three classifiers: K-nearest neighbors, relevance feedback, and bayesian independence. Similar to our approach, they tag items as negated and try to identify diagnosis and symptom terms. Additionally, their final system combines all three models. A direct comparison is not possible due to the difference in data and evaluation metrics; they use average precision and recall at k. On a comparable metric, "principal code is top candidate", their best system achieves 59.9% accuracy. de Lima et al. (1998) rely on the hierarchical nature of medical codes to design a hierarchical classification scheme. This approach is likely to help on our task as well but we were unable to test this since the limited number of codes removes any hierarchy. Other approaches have used a variety of NLP techniques (Satomura and Amaral, 1992).

Others have used natural language systems for the analysis of medical records (Zweigenbaum, 1994). Chapman and Haug (1999) studied radiology reports looking for cases of pneumonia, a goal similar to that of our automatic coding policy system. Meystre and Haug (2005) processed medical records to harvest potential entries for a medical problem list, an important part of electronic medical records. Chuang et al. (2002) studied Charlson comorbidities derived from processing discharge reports and chest x-ray reports and compared them with administrative data. Additionally, Friedman et al. (1994) applies NLP techniques to radiology reports.

8 Conclusion

We have presented a learning system that processes radiology reports and assigns ICD-9-CM codes. Each of our systems achieves results comparable with an inter-annotator baseline for our training data. A combined system improves over each individual system. Finally, we show that on test data unavailable during system development, our final system continues to perform well, exceeding the inter-annotator baseline and achieving the 4th best score out of 44 systems entered in the CMC challenge.

9 Acknowledgements

We thank Andrew Lippa for his extensive medical wisdom. Dredze is supported by an NDSEG fellowship; Ganchev and Talukdar by NSF ITR EIA-0205448; and Crammer by DARPA under Contract No. NBCHD03001. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the DARPA or the Department of Interior-National Business Center (DOI-NBC).

References

W.W. Chapman and P.J. Haug. 1999. Comparing expert systems for identifying chest x-ray reports that support pneumonia. In *AMIA Symposium*, pages 216–20.

JH Chuang, C Friedman, and G Hripcsak. 2002. A comparison of the charlson comorbidities derived from medical language processing and administrative data. *AMIA Symposium*, pages 160–4.

Computational Medicine Center. 2007. The computational medicine center's 2007 medical natural language processing challenge. http://computationalmedicine.org/challenge/index.php.

Koby Crammer. 2004. *Online Learning of Complex Categorial Problems*. Ph.D. thesis, Hebrew Univeristy of Jerusalem.

Luciano R. S. de Lima, Alberto H. F. Laender, and Berthier A. Ribeiro-Neto. 1998. A hierarchical approach to the automatic categorization of medical documents. In *CIKM*.

C Friedman, PO Alderson, JH Austin, JJ Cimino, and SB Johnson. 1994. A general natural-language text processor for clinical radiology. *Journal of the American Medical Informatics Association*, 1:161–74.

Leah S. Larkey and W. Bruce Croft. 1995. Automatic assignment of icd9 codes to discharge summaries. Technical report, University of Massachusetts at Amherst, Amherst, MA.

YA Lussier, C Friedman, L Shagina, and P Eng. 2000. Automating icd-9-cm encoding using medical language processing: A feasibility study.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. Flexible text segmentation with structured multilabel classification. In *HLT/EMNLP*.

Stephane Meystre and Peter J Haug. 2005. Automation of a problem list using natural language processing. *BMC Medical Informatics and Decision Making*.

National Center for Health Statistics. 2006. Icd-9-cm official guidelines for coding and reporting. http://www.cdc.gov/nchs/datawh/ftpserv/ftpicd9/ftpicd9.htm.

Y Satomura and MB Amaral. 1992. Automated diagnostic indexing by natural language processing. *Medical Informatics*, 17:149–163.

Edward H. Shortliffe and James J. Cimino, editors. 2006. *Biomedical Informatics: Computer Applications in Health Care and Biomedicine*. Springer.

P. Zweigenbaum. 1994. Menelas: an access system for medical records using natural language. *Comput Methods Programs Biomed*, 45:117–20.