Electronic medical records (EMRs) promise to revolutionize the health care industry by making health care better and cheaper.\(^1\)\(^2\) A key aspect of this promise is comprehensive capture of patient-specific clinical data; the EMR provides a vehicle for communicating information about the patient across time and providers. In both electronic and paper-based medical records, narratives produced by providers about their patients play a critical role in this communication.\(^3\) Adopting the EMR also may improve health care and reduce costs by enabling process control and decision support via computations using data for individual patients as well as populations. These computations promote evidence-based quality and efficiency in the delivery of health care.

In actual practice, these two key aspects of the EMR (supporting patient-specific clinical communications and workflow while enabling general process controls) often conflict with each other.\(^4\) For those care activities that heavily affect organizational goals, the care organization strives to increase coded data entry to enhance process control. However, clinicians often find coded data entry cumbersome because it interferes with individualized patient care and adds steps to an already busy workflow.\(^5\)\(^6\) Compared to traditional natural language narrative, coded entry captures only a fraction of the information produced in the clinical encounter (although what is coded can be less ambiguous). As with other mechanisms for coding complex real-world phenomena, there is a trade-off between sensitivity and specificity in clinician coding. Some phenomena will be coded with high accuracy, which will come at the expense of information captured about other aspects of the care. For instance, standardized coding of diagnoses has traditionally been a high-priority activity that efficiently communicates the common and well-understood health conditions that clinicians ascribe in their assessment of each patient. On the other hand, recording the important details of counseling about patient behavior is much more difficult to accomplish with a standardized coding scheme, in part because these details involve assessing and capturing the patient’s motivations and intentions. However, a clinician can relatively easily recall and record such details about the encounter in a progress note that quickly covers multiple aspects of the clinical encounter.

We need new technologies that allow optimization of both clinical practice and process control in the care delivery system. MediClass (a "medical classifier") is a knowledge-based system that automatically classifies the content of a clinical encounter captured in the medical record. MediClass accomplishes this by applying a set of application-specific logical rules to the medical concepts that are automatically identified in both the free-text (e.g., counseling activities in progress notes) and precoded data elements (e.g., medication orders) of potentially any EMR. As explained in detail below, the MediClass system can process data from any EMR system for which data can be expressed in the Clinical Document Architecture (CDA) data standard that is managed by the Health Level Seven health care standards organization. Our initial target implementation of the system addressed

---

**Abstract** MediClass is a knowledge-based system that processes both free-text and coded data to automatically detect clinical events in electronic medical records (EMRs). This technology aims to optimize both clinical practice and process control by automatically coding EMR contents regardless of data input method (e.g., dictation, structured templates, typed narrative). We report on the design goals, implemented functionality, generalizability, and current status of the system. MediClass could aid both clinical operations and health services research through enhancing care quality assessment, disease surveillance, and adverse event detection.

detection of smoking cessation care delivery in accord with a widely adopted, evidenced-based care guideline. The guideline consists of a five-step, sequential program called the “5 A’s”: (1) ASK patients about smoking status at every visit, (2) ADVISE all tobacco users to quit, (3) ASSESS a patient’s willingness to try to quit, (4) ASSIST the patient’s quitting efforts (provide smoking-cessation treatments or referrals), and (5) ARRANGE follow-up (provide or arrange for supportive follow-up contacts). In a companion paper, we present results from an evaluation study of the MediClass system’s performance at assessing smoking cessation care using EMR data from four health plans. We found the system to be similar in accuracy to trained human medical record abstractors. Using the trained abstractors as gold standard, the system performed with an average sensitivity of 82% and an average specificity of 93%.

In this paper, we describe the design and function of MediClass, a general-purpose knowledge-based system created for detecting clinical events in any EMR. Such a system shows promise for addressing a wide variety of concerns in health services research and clinical operations including care quality, disease surveillance, and adverse event detection.

Background

The MediClass system was built from open source components. It uses three distinct informatics technologies: (1) HL7’s CDA for representing the clinical encounter including both structured (coded) and unstructured (free-text) data elements, (2) natural language processing (NLP) techniques for parsing and assigning structured semantic representations to text segments within the CDA; and (3) knowledge-based systems for processing semantic representations addressing specific subdomains of medicine and clinical care and for defining logical classifications over the semantic contents of a clinical encounter.

The CDA provides a structured, standard format for representing all the data associated with a clinical encounter. The CDA uses XML and a domain-specific schema, yielding a representation that is simultaneously human readable (aiding in portability across institutions) and machine readable (making computer applications interoperable). MediClass employs a customized version of the CDA that simplifies classification processing while permitting ongoing compatibility with this emerging standard for medical record data.

MediClass also builds on a foundation of research projects that have pioneered NLP techniques for automatically coding or indexing clinical and medical content. Natural language processing uses a computational model of human language that includes knowledge about the structure (syntax), meaning (semantics), and contextual use (pragmatics) of words, phrases, and larger discourse units. Whereas generic NLP is still an unsolved problem, substantial success has been achieved by applications in constrained domains (e.g., detecting tuberculosis and pneumonia, coding stroke indications, and asthma management). These applications succeed because of the ability to create domain-specific models of language use within well-defined task contexts. For example, the model can include information about words that are commonly used, how these words combine to form systematically understandable speech units, and how surrounding features or communication purposes affect the discourse interpretation. Reflecting the three levels of linguistic knowledge, NLP systems use aspects of three distinct components to develop their understanding of a text document. These components include (1) syntactic processors, or parsers, that are concerned with the structural properties of sentences and their constituents; (2) semantic processors that employ a domain-specific lexicon to compute logical forms representing meanings of words and phrases; (3) contextual processors that use knowledge about specific use situations in an attempt to modify or contextualize these meanings according to domain-specific tasks and purposes.

Many systems have employed NLP methods that map clinical text to formally represented concepts within a standardized knowledge base, such as the Unified Medical Language System (UMLS). In addition, some systems have used processors or methods to apply constraints for selecting among representations produced by NLP analysis. For example, some researchers have developed applications that use word tokens (or phrases) extracted from free text portions of the electronic record as inputs to statistics-based classifiers. A significant limitation of statistics-based methods is the requirement for a relatively large number of both positive and negative examples from the clinical notes that have been categorized by human experts. In addition, in a recent head-to-head comparison in one particular domain, chest radiograph reports, Wilcox and Hripcsak found that the use of expert knowledge was more important to overall system performance and more cost-effective than any of the five methods derived from three different machine-learning algorithm classes (i.e., rule-based: decision trees and rule induction, instance-based: nearest neighbor and decision tables, and probabilistic: naïve-Bayes) that they chose for comparison.

MediClass combines NLP techniques with knowledge-based systems technology. Rather than language processing resulting directly in proposition-level representations of raw text inputs, MediClass uses the many atomic concepts identified in the free text together with coded encounter data as inputs to a knowledge-based classifier. This classifier uses a traditional rule base and forward-chaining rules engine to determine which concepts and their arrangements are relevant to specific event-detection problems.

Several UMLS-based frameworks discussed in the literature have heavily influenced the approach that we took in designing the concept identification component of MediClass. IndexFinder of Zou et al. used an efficient mapping of combinations of normalized tokens to the normalized string forms of UMLS concepts, enabling memory-resident access to the UMLS and therefore fast concept identification processing. The MetaMap approach of Aronson et al. to UMLS concept identification makes extensive use of the UMLS Specialist Lexicon tools, enabling robust and flexible lexical processing. Concept identification in MetaMap uses a novel “goodness score,” which weights the relative contributions from the subprocesses involved in parsing the source text. This process yields a measure of confidence in the concepts identified by the system.

Our work also builds on the seminal work of Sager et al. and the Linguistic String Project (LSP) developed at New York
University in the 1960s. Briefly, the LSP’s goal was to develop a generic parser for a broad segment of the English language along with a programming language for generation of natural-language grammars. An important component of this work was identification of the need for “sublanguage grammars” that placed domain-specific constraints on the representation of allowable expressions. From this work, they also developed methods to identify specific data structures that corresponded with important statement types found in clinical narrative texts by examining word-class co-occurrence relationships. For example, they found that words of the semantic class “symptom” often occurred with words from the class “body-part” (e.g., stomachache). These data structures could then be mapped directly to relational database systems, which allowed the clinical concepts and their relationships, identified from the narrative text, to be stored and queried using existing database tools. In this way they were able to ask and answer various types of questions regarding the information contained in the free-text narratives quite successfully.

The research of Friedman and colleagues with the MedLee system has also influenced our design. Recent versions of MedLee use XML representations of encounter data and perform automated clinical document markup. MedLee also uses a complex knowledge representation with the UMLS as its core ontology. This representation is augmented with concept modifiers that capture information relevant to specific concerns of clinical interpretation tasks, such as the date or severity of a finding. MedLee includes domain-specific knowledge for reconciling equivalent or ambiguous representations and mapping to codes following linguistic processing. However, the system does not incorporate a classification engine for more complex and flexible processing of the knowledge representations produced by NLP techniques. Processors have been developed to add this functionality to the MedLee system. Nielson and Wilcox discuss a tool for visually formulating “rules” by selecting and grouping the observations that are identified in output produced by MedLee. This tool would allow someone to develop rules that can then be used to define a classification scheme for application to other records parsed by MedLee.

Finally, Haug and colleagues have developed NLP systems that combine syntactic and semantic processing techniques to extract coded data items from free-text reports for use in decision-support applications. Briefly, their systems use complex event-definition knowledge structures to define important clinical events (e.g., new clinical findings) coupled with syntactic analyses using augmented transition networks to identify a grammatical role for each word (i.e., noun, verb) that is consistent with its nearest neighbors. These individual words are then bundled and reclassified using higher level syntactic categories (i.e., noun phrase, prepositional phrase). Finally, these syntactically categorized words and phrases are grouped into semantically meaningful propositions using a Bayesian network. Haug and colleagues have developed Bayesian networks to represent various pathophysiologic interpretations of disease states along with networks that can interpret text describing placement and type of hardware typically encountered in radiologic examinations.

**System Design**

MediClass provides automated coding and classification of diverse medical record data aggregated at the encounter level. The system enhances the functionality of EMRs by automatically identifying clinical events of interest to stakeholders, as defined by specific knowledge coded into MediClass. In designing this system, we aimed to address the following goals.

**Design Goals**

**Process Encounter-Level Data of Any EMR**

We wanted the system to be capable of working with any EMR that captures patient- and encounter-specific data. The system needed to accommodate EMR implementations that enforce strict coding of inputs as well as implementations that simply create an electronic “envelope” to hold dictated or typed clinical notes. Furthermore, our system could not interfere with the transactional system that supports clinical workflow. To design such a system, we separated the goal of producing structured data in the EMR (i.e., coding) from the goal of effective capture of information into the EMR (i.e., data entry). The latter may best be served by various kinds of interface devices, including ones that support natural language inputs. Our system should augment a medical record by providing a mechanism to automatically code data regardless of input device.

**Identify Medical Concepts in Both Free-Text and Coded Data**

We wanted a system able to map the contents of both coded and uncoded data into a common set of abstract medical concepts or a knowledge representation so that the entire encounter (as captured in the medical record) could be subjected to a uniform analysis.

**Generate Classifications Based on the Medical Concepts Identified**

The system needed to be capable of analyzing identified medical concepts to generate a set of higher level classifications of the medical record.

**Process the Language Peculiarities of Clinical Notes**

The NLP methods used to identify the medical concepts in the free-text portions of the medical record needed to accommodate the highly truncated and poorly formed language constructs of clinical notes typed by busy clinicians. These notes convey information with the extensive use of domain- and organization-specific knowledge and shorthand and minimal use of standard grammatical constructs.

**Provide Scalable and Explicit Knowledge Representation within System**

Although the target implementation addresses one specific care quality assessment problem (detection of smoking cessation care activities), the system needed to easily accommodate new domains or types of “clinical events.” Therefore, the system needed to leverage existing knowledge sources yet easily accommodate additional knowledge specification—knowledge that we could acquire from textbooks or domain experts. Furthermore, the system needed to be able to make explicit the links between knowledge formally encoded into the system and the classification results produced by the system.
Achieve Moderate Processing Throughput on a Readily Available Personal Computer (PC)

For our first implementation target for MediClass, the system needed to be capable of processing thousands of patient encounters on a standard desktop PC (1 GHz processor, 1GB RAM, 15GB IDE hard disk) within several hours. Meeting these performance metrics would require special attention to algorithmic efficiency in system implementation.

Locally Control Implementations

Our target implementation of MediClass would involve four different health care organizations. Because of the complications entailed by sharing protected health information across organizational boundaries, the system would have to be deployed at each site to be run locally. However, to serve the science goals for the project, the system must use common definitions of clinical events.

High-Level Architecture

MediClass is designed and built around three distinct “layers” that together form a process stack for sequentially analyzing medical encounter records, one at a time (Fig. 1). We briefly outline the function of each layer below and then describe the three layers in more detail in the following sections. The three layers of the MediClass system perform the following functions.

System Integration: The first layer in the architecture implements data-level integration between clinical information systems and the MediClass system. Specifically, the System Integration layer extracts encounter-based EMR data from clinical information systems or an offline clinical data repository via a standard format (an XML document conforming to a specialization of the HL7 CDA). This layer then loads those data into the MediClass system for further processing.

Concept Identification: This layer identifies the abstract medical concepts contained in both the free-text and coded portions of the CDA representation of a patient encounter. Medical concepts are drawn from a version of the UMLS Metathesaurus that has been customized to support medical concepts specific to the domain of interest. Concept identification for CDA sections containing free text is performed using NLP algorithms (described in detail below), which build on the logic and data contained in the UMLS SPECIALIST Lexicon and Metathesaurus. Concept identification for coded data, where the code belongs to a controlled vocabulary mapped to the UMLS, is performed directly using associated source code-to-concept mapping. Identified concepts are said to be “instantiated” when additional context captured by concept modifier logic is attached to the concept.

Classification: This final layer transforms the set of instantiated concepts produced by the previous layer into a set of higher-level classifications. The MediClass system implements classification via a rule-based classification engine that takes the set of instantiated concepts as input and generates a set of terminal states that represent the medical classifications (“clinical events”) of interest.

System Integration Design

A key feature of the System Integration layer is the HL7 CDA for comprehensively representing encounter-based clinical data. Few EMRs and none of the four systems involved in our first target implementation currently produce encounter

---

**Figure 1.** The MediClass Architecture. As described in the text, the architecture is composed of three distinct layers that are sequentially involved in processing each clinical encounter. CDA = Clinical Document Architecture; EMR = electronic medical record; UMLS = Unified Medical Language System.
data in CDA format. However, our design commitment to the CDA ensures future compatibility of MediClass with this emerging standard while permitting the concise data mapping required for the four different target EMR Data Warehouses in our implementation. Figure 2 shows a simplified schematic of the flow of processing required to prepare EMR data for MediClass via the CDA representation of a clinical encounter.

This design can be broken down into three major steps:

1. Extraction of patient encounter data from the EMR Data Warehouse by the EMR Adapter component into a set of XML documents.
2. Transformation of the EMR XML documents into an HL7 CDA XML document via an XSLT Engine—a technology for isomorphic mapping of XML documents that conform to one data definition into documents that conform to a second data definition.
3. Parsing of the HL7 CDA XML document to produce Java objects within the MediClass system representing the CDA document components. If the EMR system produces encounter data directly in a standard CDA format, then only this step is necessary.

Concept Identification

The second layer in the MediClass architecture is responsible for determining the semantic contents of a clinical encounter. This layer's primary function is to transform the record of a patient encounter structured as an HL7 CDA document into a knowledge representation in which the many different medical concepts of the encounter are identified. The general form of this knowledge representation is a collection of instantiated medical concepts drawn from a standardized medical ontology. In a knowledge-based system such as MediClass, an ontology is the universe of all possible abstract concepts and relationships among those concepts. We use the UMLS Metathesaurus as the core of our ontology. In particular, medical concepts in this ontology are identified by Concept Unique Identifiers (CUIs) from the UMLS, which link together synonymous medical terms. MediClass also makes use of UMLS grouping of concepts by semantic type; however, the many other semantic relationships asserted over concepts within the UMLS are not currently used by MediClass. In the UMLS, terms are provided by over 100 different source vocabularies (including ICD9, MeSH, SNOMED, and others), which are part of the Metathesaurus. As part of the knowledge engineering work for a particular application of MediClass, we can add our own concepts as well as new terms for existing concepts into this database. Insertion of a new term requires identification of the concept, while insertion of a new concept requires identification of the semantic type. Identifying the relevant concept or semantic type is accomplished by searching the UMLS using the appropriate terms. Later additions to the UMLS could be necessary if rules that use specific concepts are not able to produce the desired classifications for an application. The three applications of MediClass reported on below (detection of immunization adverse events, subclassification of diabetic retinopathy, and assessment of smoking cessation care delivery) required adding 0, 3, and 23 concepts and 50, 35, and 255 terms, respectively, to the UMLS.

The MediClass concept identification process generates a knowledge representation comprising a set of “instantiated concepts,” as discussed below in detail. In addition to CUI information, which identifies the abstract concept from the ontology, instantiated concepts encode location information within the CDA representation and the status of any “modifiers” that may apply to the concept as a result of local context. Modifiers can provide contextual information such as quantity (for abstract concepts that take integer or real values, such as a specific laboratory result), quality (for abstract concepts that take on discrete states along some scale or dimension, such as the severity of a symptom), and truth status (a special type of quality signifying the absence/presence of the concept, as asserted by the author of the text, such as a negative finding).

The transformation of raw encounter data into a knowledge representation in this layer of the architecture entails two distinct processes:

1. Free-text concept identification: Identification of the UMLS concepts represented by the terms contained in segments of natural language text.
2. Coded-data concept identification: Identification of the UMLS concepts represented by codes from standardized medical coding systems available within the UMLS.

![Figure 2](image-url) The System Integration Layer. The System Integration layer of the MediClass architecture aggregates and transforms encounter data from a clinical information science warehouse into Java objects within MediClass. This process is mediated by the HL7 Clinical Document Architecture (CDA) for representing the encounter as a single XML document. EMR = electronic medical record.
The knowledge representation produced by this layer of the architecture is then embedded within the CDA document and the marked-up CDA is ready for processing by the final Classification layer of MediClass.

The MediClass system executes free-text processing on all segments of character data contained within “<content>” elements in the input HL7 CDA document using a four-stage process. First, the entire text is parsed for sentence boundaries and other special patterns of interest (e.g., datasystem-specific structured text). Second, each token in the text is subjected to lexical processing involving tokenization and word variant generation, which includes possible spelling correction. Third, concept identification proceeds by using an “input window” (of configurable width) that moves across the text to define candidate strings. Candidate strings are then compared against normalized string representations of UMLS Metathesaurus concepts to locate concept matches. Concept matches are scored for “goodness” according to the number of derivations and deletions that were applied to the original text segment to create the match. The system processes the local context information of each concept and attaches the relevant modifiers to produce the final concept instances.

Lexical Processing
The lexical processing stage takes as input a segment of English natural language text and outputs a two-dimensional array of tokens. The first dimension is defined by the normalized forms of the words most likely to hold semantic meaning, and the second dimension comprises the lexical variants of each word (e.g., spelling variant, inflectional variant, synonym, acronym/abbreviation, derivational variant). This high-level, logical architecture is illustrated in Figure 3.

Three main types of linguistic knowledge are involved in variant generation:
1. Morphologic knowledge (e.g., inflection based on grammatical category of the word)
2. Orthographic knowledge (e.g., spelling variation and spelling correction)
3. Semantic knowledge (e.g., synonymous terms and acronyms)

The results of lexical processing depicted in Figure 3 are encapsulated in a Java LexicalStructure object, which is added back into the CDA Java object model within the object that held the original source text block.

Concept Identification
The concept identification stage considers all candidate strings formed from word tokens within the “input window,” together with variants resulting from lexical processing of these words (as shown in Fig. 3). Concept matches are found by comparing candidate strings against all string representations of UMLS concepts. Candidate strings need not reflect the word ordering of the original text to create a match. For example, the original text segment “gave advice on smoking cessation” will match the UMLS concept called “Smoking cessation advice” that is identified by the Concept Unique Identifier (CUI) C0150352 in the UMLS Metathesaurus. The example string would also produce matches for the independent concepts of “advice” (C0150600) and “smoking” (C0030769). A configurable system parameter can restrict concept matches to only those candidate strings whose tokens occur in the same sentence in the text. Finally, matches are given a “goodness of fit” score that is computed from (1) the amount of separation in original text among word token variants involved in the match, and (2) the scores of variants involved in the match. Variants and their scores are produced by the “Generate Fruitful Variants” configuration of the Lexical Variant Generation (lvg) tool of the UMLS. Figure 4 shows an example of the entire process from clinical note text segment to identified and instantiated medical concepts.

Concept Instantiation
The concept instantiation process identifies modifiers for a concept given the local context. An example is negation of a concept, such as a negative finding in a clinical note. In MediClass, modifier logic is implemented by modifier detection modules (Java methods) that output modifier data into the representation of concept instances (see Fig. 4). A modifier detection module can be one of three possible types, based on the type of output it produces: Boolean, numeric, or string. Each detection module is run in the context of each identified concept and therefore has access to the concept’s semantic types as well as the entire parsed CDA. The module’s outputs provide contextual data about each concept called “modifiers” that are used by the Classification layer of MediClass, as discussed below. Modifier detection modules (and thus the set of possible modifiers) are incorporated into MediClass via a configuration file that defines their existence and availability to the program at system run time.

To date, we have implemented and experimented with negation, severity, and quantification modifier detection modules.

![Figure 3](image.png)

**Figure 3.** Logical architecture for the lexical processing of a text segment. The process produces a set of variants for each token identified in the text segment.
For example, our negation detection module implements logic for identifying the negation of concepts. It identifies members of a small set of negation terms and tokens and their syntactic arrangements within the sentence context of an identified concept. By looking forward (at succeeding tokens) and backward (at preceding tokens) in the context of the concept, the logic implements a finite state machine that can detect simple, but common, negation constructs of medical language used in clinical notes. The MediClass negation detection module outputs a "true" or "false" value as modifier for each concept considered. Although this logic is simplified given all the complexity of negation in medical language, our informal experiments have shown the method to perform fairly well. Our method is informed by the work of Mutalik and colleagues. In their study, they found that a small set of negation signals ("not," "no," "denies," "without," together with variants of these) accounted for over 92% of negations in discharge summaries and surgical notes. Even simple syntactic parsing that used these signals captured 67% of the negations. We believe our negation detection module implements a significant amount of the lexical specification and parse grammar used in the study of Mutalik and colleagues.

We anticipate that modifiers of interest will fall into two very general classes: quantification (e.g., capturing value of a laboratory result) and qualification (e.g., capturing degree or severity of a symptom or the positive/negative status of a finding). However, this classification of modifiers is purely speculative.

Within the MediClass framework for implementing modifiers, modifier detection modules may be general to all concepts, specific to classes of concepts or to individual concepts. We anticipate that some modifiers will be application specific and thus represent one place where procedural code can be easily modified to extend the system to address new problem domains. Investigations of modifiers, as well as development and validation of modifier detection modules, are ongoing activities of the research team.

Once outputs produced by modifier detection modules are stored with the respective concept instances, the entire set of instances constitutes the knowledge representation of the text segment.

Identification of the UMLS concept associated with a CDA-coded entry element is implemented by MediClass as follows:

- If the coding system is one of the source vocabularies supported by the UMLS Metathesaurus, the UMLS concept for the code is simply retrieved from the appropriate UMLS database table.
- If the coding system is not supported by the UMLS Metathesaurus, the coded entry name (e.g., the string name of a medication or medical condition) is mapped to UMLS concepts using the free-text concept identification algorithm discussed above.

Figure 4. Natural language processing in MediClass. The schematic shows how MediClass produces a set of instantiated concepts from a text segment. First, lexical variants are combined in all possible ways to produce "candidate strings." These are then compared against the string representations of concepts in the Unified Medical Language System Metathesaurus to identify concept matches. Additional context about the match (including the results of procedural "modifier logic") is stored with the final concept instances that constitute the knowledge representation of the text segment.

<table>
<thead>
<tr>
<th>Concept</th>
<th>Identification And Instantiation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continue, C0750536, MatchScore, [context info]</td>
<td></td>
</tr>
<tr>
<td>Smoking, C0037369, MatchScore, [context info]</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Input Window</th>
</tr>
</thead>
<tbody>
<tr>
<td>patient (n) continue smoke (v)</td>
</tr>
<tr>
<td>patient (adj) continuous smoky (adj)</td>
</tr>
<tr>
<td>patientce continually smoker</td>
</tr>
<tr>
<td>patiently</td>
</tr>
</tbody>
</table>

Lexical Processing

patient continues to smoke 1/2ppd. not ready to quit.
Classification

As discussed in the previous section, the Concept Identification layer of MediClass produces a knowledge representation of the encounter consisting of many instantiated concepts; typically hundreds of concept instances are produced for an average encounter. The Classification layer of MediClass then uses domain-specific rules to determine whether the concepts of interest for a specific clinical event detection problem are present and appropriately arranged in the data. Programming the Classification layer of MediClass entails specifying rules defining the classes of interest in order to define all and only the relevant encounters belonging to each class.

Classification is performed by a rules engine executing a set of forward-chaining logical rules over the set of concept instances produced during Concept Identification. These concept instances may have been derived from either coded or free-text data elements of the originating CDA document, and rules may be written to be sensitive or indifferent to those origins in the source data. The MediClass classification engine was designed to support extensive handling of the rich knowledge representation produced by the Concept Identification layer. Each rule specifies a set of concept instances by means of logical constraints and produces as output either a new concept (called an “intermediate concept”) or a classification decision (called a “terminal state”). Intermediate concepts can be used by other rules and are the basis for forward chaining by the rules engine. The terminal states of a set of rules represent the final classifications that MediClass can generate.

The Concept Identification layer provides the originating knowledge representation, which is then loaded into a working memory to initialize the classification engine. The engine operates by iterating through all rules that have not yet “fired” and determining if the contents of working memory apply to any of these rules. Each rule specifies constraints that are tested against the knowledge representation resident in current working memory. A rule applies or “fires” when its constraints are met; it then produces either an intermediate concept or a terminal state and adds this to working memory. When the classification engine passes through all the rules without any rules firing, then it halts and all terminal states produced up to that point define the classifications of the current encounter. Figure 5 shows a simplified example of how the classification engine determines that the clinical event “Ask”—one of the guideline recommended 5 A’s of smoking cessation care—occurred during the hypothetical encounter represented by the simple clinical note shown earlier, namely the text segment “patient continues to smoke 1/2ppd. not ready to quit.”

---

**Figure 5.** The MediClass classification process. The rules engine applies rules to the concept instances located in “working memory.” If a rule “fires,” it produces a “TerminalState” or an “IntermediateConcept” (IC) and is removed from further consideration. The rules engine halts when there are no rules left to apply that can fire. The set of terminal states produced up until the engine halts are the final classifications produced by the system.
Rules employ “constraints” to specify criteria that determine, in conjunction with the contents of working memory, whether a rule fires or fails to fire. Two types of constraints may be coded into rules by the rule author: concept-level constraints and rule-level constraints.

Concept-level constraints determine which individual concept instances in working memory apply to the rule. For instance, concepts may be required (with the AND keyword), optional (with the OR keyword), or excluded (with the NOT keyword) in the rule specification. In addition, modifier information stored with the concept instance can be evaluated by concept-level constraints to determine applicability of the rule. For example, the Boolean value of a concept’s negation modifier may be tested by a concept-level constraint to determine whether the rule may fire based on the concept’s negation status. Similarly, a concept instance with a quantification modifier may be tested by a concept-level constraint written into the rule to determine whether the rule may fire. For example, the modifier value attached to the concept “packs_per_day” (generated by numeric output from the quantification modifier detection module) constitutes a concept instance that can be tested with a concept-level constraint such as “packs_per_day[value] > 1.”

In addition to concept-level constraints of a rule, which determine the suitability of a specific set of concept instances in working memory, rule-level constraints express additional requirements that must be satisfied across this set for the rule to fire. These include constraints specifying (1) section: the concept instances must be within specific sections of the original CDA document for the rule to fire; (2) ordering: the concept instances must be in a specific order in the original document for the rule to fire; (3) proximity: for the rule to fire, the tokens that generate concept instances must be within some threshold distance of each other (where distance is given in terms of character separation); (4) sentence separation: this constraint is similar to the proximity constraint but specifies distance in terms of sentence separation rather than character separation.

A rule fires when the contents of working memory meet the constraints specified by the rule. When no rule will fire given the contents of working memory, then the classification engine halts. All terminal states produced during execution of the classification engine and all rule firings that lead to these terminal states are stored with the encounter information allowing complete reconstruction of the behavior of the classification engine. The set of rules used by the MediClass system for a given detection problem is specified and maintained in a single XML file, which simplifies authoring and maintenance of classification rules. The rule called “SmokingStatus” depicted in Figure 5, is shown in its XML format in Figure 6.

**Status Report**

The MediClass system has now been successfully deployed at four health maintenance organizations (HMOs) across the country. A local analyst at each site with only basic information technology experience can run the MediClass system in a secure local environment, using a standard desktop PC installed with open source infrastructure components. As a result, sensitive medical record information never leaves the organization, yet care activity assessments (encounter classifications) made by the system can be aggregated and meaningfully compared across all four sites. Each installation employs a custom “EMR Adapter” to pull together disparate data from a clinical data repository into CDA documents that comprehensively represent each encounter. As the CDA evolves into a practical standard, we anticipate that CDA representations of the encounter will be made directly available from within vendors’ EMR solutions. An installed MediClass system processes CDA documents at a rate of 10,000 encounters per 24 hours on a single basic desktop PC. The MediClass system employs a vast, publicly available medical knowledge base (the UMLS Metathesaurus), and it allows for modular extension of this knowledge base by adding custom concepts and rules that are unique to a specific clinical event detection problem. The system employs NLP techniques allowing automated classification using both the coded and uncoded or free-text portions of the EMR. Furthermore, these NLP techniques are well suited for the terse, ungrammatical, and poorly structured representations produced by busy clinicians who type their own clinical notes in order to efficiently capture many and diverse aspects of the primary care encounter.

We have applied MediClass to assess delivery of the 5 A’s of smoking cessation7–9; to detect immunization adverse events37,38; and to classify subtypes of nonproliferative diabetic retinopathy as recorded in the EMR. When configured

```
<SmokingCessRules.rule name="SmokingStatus" terminalState="Ask" section="progress_note" proximate="true">
   <AND>
      <concept name="SmokingIndicator" CUI="IC1"/>
      <OR>
         <concept name="Continued" CUI="C0750536"/>
         <concept name="Former" CUI="C0750523"/>
         <concept name="History" CUI="C0199182"/>
      </OR>
   </AND>
</SmokingCessRules.rule>
```

**Figure 6.** An example rule called “SmokingStatus” shown in XML format. In order for this rule to fire, there must be a “SmokingIndicator” (produced as an output of another rule because it is identified as an “IntermediateConcept” or “IC”) and one of the concepts “Continued,” “Former,” or “History” present in the progress note and proximal to one another.
to detect immunization adverse events, MediClass had sensitivity and specificity of 75% and 97%, respectively, in a 248-record training set, with a 25% prevalence of immunization adverse events. When configured to identify moderate or severe nonproliferative retinopathy in ophthalmology and optometry notes, MediClass demonstrated 81% sensitivity and 89% specificity in a training set of 115 patient records identified with visit diagnosis codes of type 2 diabetes and retinopathy. For each of these clinical event detection applications, the MediClass architecture did not change. Each application required defining the new classification rules and the addition of application-specific terms and concepts to the UMLS, as described above.

Our target implementation of MediClass was for an NCI-funded study called HMO Interventions in Tobacco2 (HIT2), involving detection of smoking cessation care delivery within primary care encounters. The HIT2 study involves four different HMOs and four distinct EMR data systems. The study aims to use MediClass’ assessments of encounters to generate feedback reports to clinicians on their use of the 5 A’s during care for patients who are smokers. The first 18 months of the project involved designing and implementing the MediClass system; developing the custom concepts and classification rules for detecting the 5 A’s with MediClass; validating the system through comparison to trained (human) medical record abstractors; building at each site the “EMR Adapter,” which produces EMR data in the standard CDA format used by MediClass; installing the system at each site; and training an analyst at each site how to run the system.

As described above, MediClass leverages a vast knowledge base of medical concepts but also requires encoding specific concepts and rules into the system for specific clinical event detection problems. We consider this process to be a knowledge-engineering task: in this case, defining and encoding the 5 A’s of smoking cessation into the system. We met weekly with investigators on the research team, which included both clinicians and tobacco researchers from each study site, to conduct this knowledge-engineering work. We started with the consensus 5 A’s guidelines to define the prescribed care delivery activities.24 Because the research team was interested in distinguishing among four different types of “Assist” (handing the patient a brochure, counseling him or her directly, referring him or her to a class or establishing a quit date, and pharmacotherapy assistance), we had the program and the experts code the records for “8 A’s” (Ask, Advise, Assess, Assist 1–4, and Arrange). We also pulled sample clinical notes from the EMRs at each site to get general and site-specific examples of smoking cessation care activities that are recorded in the medical record.

Once we had encoded the relevant rules and concepts into MediClass, we ran the system on a small training set (144) of sample text segments (smoking cessation content excerpted from clinical notes at each health plan) that had been manually coded for the 8 A’s by the research team. We detected mistakes made by the program, then modified the system knowledge and evaluated again. After several iterations, we determined that the system was performing on a par with our project team experts. Table 1 shows the interrater agreements among the five experts and MediClass in coding the entire training set. Each text segment in the training set was, on average, 93 characters and agreement between each pair of coders was measured with kappa by considering agreement on 1,152 binary decisions (yes/no on eight categories for 144 segments). Table 2 shows some example clinical note segments from the training set, and the classifications produced by the experts and by MediClass for these examples. The average prevalence of positive findings was 28% across all experts, which, although this low prevalence may have produced a downward effect on the magnitude of kappa, was adequate to justify use of the kappa statistic.36 The final column of Table 1 clearly shows that MediClass performed similarly to our experts as measured by mean agreement with the other human coders. Finally, we ran several large validation studies involving hundreds of real primary care encounters of known smokers at each of the four sites. Four trained abstractors provided the gold standard for these studies. We report the details of these studies elsewhere.9 Overall, the validation study results were consistent with the accuracy obtained and reported here for the training set. Given the much lower cost (and reduced tedium) of using MediClass rather than humans for this classification, we believe the system to be a success. The HIT2 project team recently began a randomized, controlled trial across these four HMOs to test the effects of feeding back smoking cessation care reports (based on the results of MediClass processing) to participating primary care clinicians.

### Discussion

MediClass uses a knowledge-based system encoded with domain-specific knowledge (i.e., rules specific to an event-detection problem) as a classifier that processes the outputs produced by NLP techniques. We have designed the MediClass system with the goal of addressing any clinical event detection problem, using the data from any EMR system. Each clinical event detection problem or domain requires modeling the knowledge used by those who record the events in the medical record, and each implementation of the system may require handling of some specific content that is derived from the particular EMR or clinical practice. MediClass was designed to enable modeling of all aspects of care delivery, not just the diagnosis-related concerns of clinical reasoning. This design focus has a number of consequences for the built system including (1) a classifier that processes the entire medical record, including both the coded and free-text portions; (2) use of only “weak” NLP methods to enable classification of text that is rich in semantic content but phrased in minimalist linguistic constructions; for example, we wanted to be able to classify the report of a conversation between clinician and patient; and (3) “strong” knowledge-based methods for applying both semantic and syntactic constraints to a rich knowledge representation of the encounter.

<table>
<thead>
<tr>
<th>Set (n=144)</th>
<th>C1</th>
<th>C2</th>
<th>C3</th>
<th>C4</th>
<th>C5</th>
<th>Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>0.57</td>
<td>0.73</td>
<td>0.63</td>
<td>0.73</td>
<td>0.67</td>
<td></td>
</tr>
<tr>
<td>C2</td>
<td>0.57</td>
<td>0.47</td>
<td>0.43</td>
<td>0.52</td>
<td>0.47</td>
<td></td>
</tr>
<tr>
<td>C3</td>
<td>0.73</td>
<td>0.47</td>
<td>0.63</td>
<td>0.67</td>
<td>0.59</td>
<td></td>
</tr>
<tr>
<td>C4</td>
<td>0.63</td>
<td>0.43</td>
<td>0.63</td>
<td>0.59</td>
<td>0.55</td>
<td></td>
</tr>
<tr>
<td>C5</td>
<td>0.73</td>
<td>0.52</td>
<td>0.67</td>
<td>0.59</td>
<td>0.59</td>
<td></td>
</tr>
<tr>
<td>MC</td>
<td>0.59</td>
<td>0.47</td>
<td>0.58</td>
<td>0.63</td>
<td>0.58</td>
<td>0.57</td>
</tr>
</tbody>
</table>

MC = MediClass.
The NLP methods that we use are “weak” because they do not explicitly employ syntactic constraints in text parsing and produce a noisy knowledge representation that includes many spurious concepts and propositions. Many NLP systems rely heavily on syntactic constraints in attempts to generate clean proposition-level representations; however, these systems are typically restricted to processing the noun phrases of well-formed sentences. Although dictated clinical notes are often made grammatical by the transcriptionist, notes typed by the clinician are often not well-formed grammatical constructions. Furthermore, much of the meaning that is relevant to general care delivery processes will be found in linguistic units other than well-formed noun phrases. This is certainly the case for what gets recorded in clinical notes about smoking cessation care, which includes reference to what is said, desired, and intended. Finally, our design leverages large and fast memory and processing cycles. Therefore, the system tolerates many spurious concepts in a preliminary knowledge representation generated by weak NLP methods because the classifier’s rules add problem-specific constraints to filter out the inappropriate concepts.

The system has been applied to three distinct problems representing patient safety (detection of possible vaccination reactions), disease surveillance (subclassification of diabetic retinopathy), and care quality (assessment of smoking cessation care delivery). Several limitations of these evaluations should be noted. In only the smoking cessation case has a true evaluation study been conducted that uses a gold standard created from a true “test set” of records. The evaluation reported in this paper addresses only the system’s performance using clinical notes preselected for their smoking cessation content and used in development and training of the system for the smoking cessation application. Furthermore, as shown in Tables 1 and 2, the problem of interpreting clinical notes for smoking cessation content is difficult and creates coding-agreement problems even among human experts. Task difficulty, possibly in combination with relatively low prevalence of events in the data, created lower magnitudes of the system’s knowledge was described above (see the Concept Identification section).

Our design of the system requires an effort to model each detection problem of interest. This effort can range from relatively easy to very hard, depending on the event-detection problem being addressed. During the course of development of the target implementation addressing smoking cessation, we were surprised and often frustrated by how long it took to codify the knowledge necessary to classify encounters according to the 5 A’s guidelines. Such knowledge (although effortlessly coined in shorthand phrases like “the 5 A’s”) is poorly specified and often disconnected from the realities of documentation in the medical record. One part of clinical practice involves recording what happens in the encounter within the medical record. Capturing conversations with the patient and expressing intents and plans for action is complicated, and interpreting what gets captured is difficult, for both humans and machines. Our experiences on the HIT2 project promise to add insight into this important aspect of clinical practice.

Our design attempts to minimize the specialized treatment of data from each EMR and health plan. However, some aspects of an EMR implementation or of a health plan’s practices must be specially treated. For instance, each health plan has unique content that is relevant to smoking cessation care delivery. A health plan’s unique smoking cessation treatment program (e.g., “Quitline”), treatment names, or counseling services must all get added into the ontology as terms that represent the appropriate medical concepts. This knowledge engineering task entails elaborating the abstract concepts (used by the rules developed to define the classification) with specific text identifiers of programs, brochures, counseling, therapy, or local processes used by clinicians. This task of updating of the system’s knowledge was described above (see the Concept Identification section).

We were also surprised to find that for some EMR implementations, large amounts of structured text routinely get “dumped” into the clinical notes, which can make them difficult to read and often difficult to automatically process. This text may get inserted in response to user macros or typing shortcuts or simply as part of the clinical workflow associated with the encounter. Since this text is not really natural language (i.e., it is often canned text rather than authored narrative), it often created problems with the rules that we had
crafted to correctly classify 5 A’s events from more naturally recorded narrative. To accommodate these cases, we added a preprocessing step to the Lexical Processor that employs simple string pattern replacement techniques for these rare text cases as the first step in the standard flow of MediClass text processing. In the study that uses the system to measure smoking cessation care delivery in four health plans, we are using just nine text-replacement rules to handle these problems.

Finally, we discovered that the simple model of language use underlying our system’s clinical event detection is not currently well suited for handling historical data that often make their way into the encounter representation. To better handle this type of data will require elaboration of a mechanism for temporally contextualizing concepts as “historical statements.” We envision two distinct but interdependent aspects to a mechanism that could address historical statements in the MediClass framework. The first aspect involves modeling the historical nature of concepts using modifiers. It may be possible to have a modifier detection module code the historical meanings of statements such as “Hx of smoking” or “Prior reactions noted” with a modifier that properly creates a temporal frame of reference in the knowledge representation. The second aspect of a mechanism to address historical statements requires supporting more complex reasoning or inferences that use this knowledge representation. This role is played by rules in MediClass, which mediate between identified concepts and the classes representing clinical events of interest. Processing historical statements in support of complex reasoning about the encounter is an important area of future work.

Supporting patient-specific clinical communications and enabling robust care process controls are two promises of the modern EMR. The care organization strives to increase coded data entry to the system because this enhances care process control. However, coded data entry can be cumbersome for clinicians, can interfere with clinical work, and may reduce the total amount of relevant clinical information that is captured in the EMR. MediClass is a knowledge-based system that processes both free-text and coded data to detect clinical events in the medical record. It is an example of technology that strives to optimize what is needed for both clinical practice and process control by coding medical record contents regardless of input method (e.g., dictation, structured template, code pick lists, or typed narrative). MediClass could improve health services research and clinical operations in important areas such as care quality assessment, disease surveillance, and adverse event detection. It is the goal of future work to determine how well each of these areas can be addressed with this technology.

References