Validation of an Ontological Medical Decision Support System for Patient Treatment Using a Repository of Patient Data: Insights into the Value of Machine Learning

ATIF KHAN, JOHN A. DOUCETTE, and ROBIN COHEN
University of Waterloo

Abstract

In this paper, we begin by presenting OMeD, a medical decision support system, and argue for its value over purely probabilistic approaches that reason about patients, for time-critical decision scenarios. We then progress to present Holmes: a Hybrid Ontological and Learning MEdical System which supports decision making about patient treatment. This system is introduced in order to cope with the case of missing data. We demonstrate its effectiveness by operating on an extensive set of real-world patient health data from the CDC, applied to the decision making scenario of administering sleeping pills. In particular, we clarify how the combination of semantic, ontological representations and probabilistic reasoning together enable the proposal of effective patient treatments. Our focus is thus on presenting an approach for interpreting medical data in the context of real-time decision making. This constitutes a comprehensive framework for the design of medical recommendation systems, for potential use by medical professionals and patients both, with the end result of personalized patient treatment. We conclude with a discussion of the value of our particular approach for such diverse considerations as: coping with misinformation provided by patients, performing effectively in time-critical environments where real-time decisions are necessary, and potential applications facilitating patient information gathering.

1 Introduction

Modern health information systems (HIS) make it possible to manage and process large amounts of medical data, allowing for informed medical decision making. The use of clinical decision support systems (CDSS) have been shown to improve practitioner performance [24] as well as medical workflows (such as drug dosing, preventive care etc.) [28, 34].

Today’s modern HIS are capable of storing, searching and processing heterogeneous datasets from diverse knowledge sources, such as electronic medi-
cal patient records, drug interaction registries and epidemiological databases. These systems are also capable of providing real time access to relevant medical information, thus opening up new possibilities for utilizing decision support systems in various medical settings. Access to current information from relevant sources can drastically increase the quality of medical decision making processes, improving the level of patient care and safety [36, 39]. There are also potential implications for secondary healthcare services, such as adherence to clinical guidelines [52]. Unlike the older CDSS, modern medical decision support systems should provide additional capabilities including:

1. compatibility with both structured and free-form medical data.
2. the ability to integrate information from different sources.
3. the ability to understand & translate information from heterogeneous systems.
4. tolerance for sparse patient data.
5. ability to explain their decisions and recommendations.

Despite the advantages hinted at by modern CDSS, integration with medical systems has been slow due to the proprietary nature of many health information systems. The exponential growth in the size of raw health related data\(^1\) has widened this integration gap, making timely information aggregation and retrieval a challenge. These problems are worsened by limited provisions for direct knowledge exchange between different health information systems. These factors mean that practitioners who wish to use medical decision support often have to query several different systems and then make sense of the results themselves. The decision maker’s ability to understand and process the data dictates the accuracy of the final decision, as opposed to the level of detail and usefulness of the relevant information. This severely limits the potential benefits of having large datasets and CDSS for medical decision making processes.

In our previous work ([35]) we presented an ontological decision support system focused on clinical decision support. Our approach utilized an ontological knowledge representation and semantic reasoner to answer user queries, using knowledge inference rules to discover the answer to a query. This semantic and logic based approach to decision making provided a means of producing accurate results, along with explanations expressed in first order logic for each system created result. From here, we progressed to present OMeD ([19]), a proof-of-concept implementation of the original framework, and contrasted it with various machine learning algorithms. Operating on simulated patient records, we were able to demonstrate that machine learning approaches

\(^1\)We use the term health related data to include all relevant datasets that might play a role in the medical decision making process. These datasets can differ widely in scope, from patient focused sets (such as electronic medical records) to population focused sets (such as public health data), and knowledge focused sets (such as drug-to-drug, drug-to-disease interaction registries).
frequently misidentify relationships between medical concepts, and that OMeD
greatly outperformed these methods in correctly responding to decision making
queries.

Despite the obvious strengths of a knowledge-based decision support system
(like OMeD), there is a fundamental underlying requirement that all information
(knowledge) must be complete and present before decision making inference
rules can be applied in response to a user query. Information gaps about patients
in the knowledge base can severely limit the performance and the usefulness of
such systems. This creates a challenge when these systems are deployed in real
world healthcare settings due to the fact that at any instance in time only a
subset of medical information is accessible for decision making. However, the
transparency and speed which accompany a knowledge-based approach are still
extremely valuable. This motivates the creation of a noise tolerant decision
support system, while maintaining these desirable properties.

In this paper, we present a Hybrid Ontological and Learning MEdical
System (Holmes) that extends OMeD and augments the purely ontological
model with probabilistic reasoning, using machine learning techniques. We be-
lieve this hybrid system better addresses the requirements of real-world settings,
where medical knowledge is always in a state of flux and is never observable in
its entirety. OMeD’s decision making capabilities are strongly rooted in first or-
der logic, and require complete information to find an answer for a user query.
Holmes’ hybrid design addresses this particular limitation by using machine
learning algorithms for probabilistic inference when patient data is missing. In
the event of missing information, these probabilistic models provide estimates of
the missing values for Holmes’ core semantic decision making process. Similarly,
the knowledge base in Holmes’ semantic components can be used to constrain
the machine learning models, reducing the difficulty of learning necessary pat-
tens.

It is important to note that machine learning methods are often inadequate
for complex medical decision support when used in isolation. Highly non-linear
decision rules may require a large amount of expert knowledge during the model
specification process. And whereas an ontological model can be easily extended
to express a new concept by building on older work, with machine learning meth-
ods an entirely new model must be constructed to allow for the new concept.
Further, the models produced by machine learning algorithms may be difficult
to translate into terms a computational layperson can easily understand. For
example, in medicine it is generally not sufficient to recommend a treatment
simply because the targeted patient is similar to others who received it within
a high dimensional feature space. Instead, medical personnel would prefer a
clearly stated medical reason for the suggested treatment. For these reasons, an
ontological approach is still quite desirable.

In summary, Holmes represents a framework designed to create a modern
medical decision support system that can provide accurate and reliable medical
decision support in real-time. The key contribution of our work is the
hybrid construction outlining exactly how to combine ontological and machine
learning techniques in a robust decision support system highly tolerant to absent
patient medical information.

Holmes is also designed to work with many different large datasets from distributed heterogeneous information sources, and to produce accurate, verifiable and interpretable decisions by consuming and processing all relevant information using machine automation. Finally, Holmes is designed to provide decision support with noisy, sparse, real-world datasets, unlike many preceding CDSS.

Similar to OMeD, Holmes utilizes semantic information representation and inference techniques. It uses ontologies for this purpose and is capable of working with multiple ontologies using static or dynamic mapping techniques [32, 37, 48]. This approach allows for aggregating information from different sources independent of local data formats. As long as the remote system can be queried and can return query results supported by an ontology, it can be used by Holmes. The ontology based approach also allows sharing of higher level concepts. For example, inference rules\(^2\) can be shared as long as they can be defined using an ontology. The use of an ontological data representation provides a detailed taxonomy, defining the concepts required to describe raw data as information. It also provides detailed relationships, explaining how the defined concepts relate to each other. In contrast, older expert systems built proprietary knowledge sets, making integration with real-world systems notoriously difficult.

In the sections that follow we describe the “Holmes” system in full and proceed to a validation of its units through a prototype implementation. In presenting Holmes we are able to a) demonstrate the application of a hybrid (semantic & machine learning) decision support system to large heterogeneous real-world datasets b) show that the use of machine learning allows such a system to offer greatly improved robustness when working with missing or incomplete pieces of data, as is often the case in real world medical settings c) demonstrate that, when working with complex prescription protocols, the explicitly coded semantic knowledge in this approach affords it a high degree of accuracy even when statistical methods find the problem difficult or intractable.

1.1 Motivational Example

The following example presents a scenario where Holmes might be useful, and provides an understanding of how the various higher-level components interact with each other.

Imagine a simple scenario where a healthcare provider has to make decisions regarding prescribing sleeping medications to different patients. For simplicity, let us only consider one dimension of the drug prescription protocol – drug-to-drug interactions. That is, if a patient happens to be currently taking other medications, then a sleeping drug can only be prescribed if it is safe to consume in conjunction with existing drugs.

We will later see that by introducing various other medical factors, even this simple scenario turns into a complex decision making process - potentially

\(^2\)Similar to RDF, RuleML (http://ruleml.org/) provides a mechanism to define knowledge inference rules in a standardized format.
requiring the healthcare provider to have knowledge of many different pieces of information. Furthermore, when non-medical factors like knowledge or temporal constraints are introduced, decision making can degrade further, loosing accuracy and impacting patient care. Holmes addresses these challenges as described in detail in the next section.

In order to facilitate this decision making process, Holmes will allow the healthcare provider to make the following patient centric query: “Can patient Bob be given sleeping pill Eszopiclone?” The system identifies the relevant concepts (such as: patient, Bob, sleeping pill, Eszopiclone), and the required information (Bob’s electronic medical records and drug-to-drug interaction rules) for answering the user query. Holmes will identify the key pieces of information and will reason about their relationships using inference rules. For example, Bob’s medical records may identify existing medications that (according to drug-to-drug interaction rules) cannot be administered in conjunction with Eszopiclone. Based on the identified information and automated reasoning strategies, the system will then produce a (semantic) result for the query with two important components:

1. The answer to the user query.
2. An explanation for the answer.

The healthcare provider can then decide whether or not to make use of the provided answer, based on the correctness of the generated explanation. For example, the system might respond by telling the user that Bob cannot be given Eszopiclone. The provided explanation might read:

1. Bob’s medical records state that he consumes an average of 14 units of alcohol per week.
2. Eszopiclone has an adverse interaction with alcohol.
3. Bob should not be given Eszopiclone.

Let us now further assume that Bob has never been diagnosed with depression before. Therefore, his medical records do not contain factual evidence of depression. In this case, the semantic reasoning component of Holmes will not consider depression in its analysis – as the semantic reasoner produces results purely on the basis of facts. However, there might be other facts (such as recent weight gain, change in employment status, drop in income level, lack of healthcare coverage, etc.) that, when viewed together, can indicate a disposition to depression. Therefore, even in the absence of an explicit fact stating whether or not Bob has depression, Holmes’ machine learning capabilities will be put to use. The machine learning component allows discovery of latent associations present in the raw data, which are then presented to the semantic system for consideration. In this instance, the machine learning component has a model of what both depressed and non-depressed patient records look like. Bob’s record is put into the model. Patients like Bob were found to be depressed 75% of the
time, so the machine learning component informs the semantic component that there is a 75% chance that Bob is depressed. The semantic component can then incorporate this information into its reasoning.

The remainder of the paper is organized as follows: § 2 walks the reader through all major components of the high level framework architecture of Holmes. In § 3, we explore how a real world dataset is processed to be used by Holmes to (i) create a semantic patient knowledge base and (ii) to create machine learning models from the raw data. § 4 provides a detailed construction of experiments carried out against real-world datasets to validate the proposed framework. We discuss the results and our findings in § 5. In § 6, we contrast Holmes with other clinical decision support systems. § 7 expands on natural extensions of the work presented in this paper. Finally § 8, summarizes the main findings of this paper and concludes with a summary of our contribution.

2 Proposed Framework

In this section, we describe the architecture of Holmes and its main components. This architecture can be adapted for a diverse range of medical decision support systems. The Holmes architecture is an extension of the basic OMeD framework suggested initially by [35] and then extended by [19]. The major enhancement made in this work is the integration of the ontological decision support system with machine learning techniques to enhance the decision support when only partial information is present. Holmes also enhances the knowledge integration components of the previous framework (OMeD), in an effort to make the system better suited for real-time decision support. We will discuss these changes in more detail in subsequent sections. Figure 1 depicts the Holmes architecture. The major components of Holmes are as follows:

2.1 Communication Interface Module:

The communication interface module allows users to pose high level composite queries. In the context of our motivational example, the user query “Can patient Bob be given Eszopiclone?” is a complex query that involves generating and satisfying additional sub-queries against different health information systems. The communication interface module requires a complex query and an optional problem context as its input, and produces the query result along with an explanation for the generated result as its output. The optional problem context is utilized for providing additional constraints that might be relevant for fulfillment of the query.

The communication interface module is designed to support two distinct types of communication interfaces. The first is to facilitate human users. The queries and their results may require additional pre & post processing to translate the results into a format understandable by human users. A user can post

---

3By composite query we refer to complex queries that require additional sub-queries to be performed to find the required answer.
his query in a *constrained language* (such as English with constrained vocabulary) and can expect the answer back in the same format. The above mentioned query (regarding patient Bob and sleeping medication Eszopiclone) is an example of a constrained language query. The second type of supported communication is for computational agents, facilitating automation. For example, a computational agent may expect the communication (request and response) to be in a machine processable format.

Let us take a closer look at the inner workings of the communication interface module in the context of our motivational example, where a healthcare provider (such as a nurse) needs to administer a sleeping pill (Eszopiclone) to patient Bob. Let us further assume that Bob is an elderly patient in a home-care setting with a history of alcohol abuse, recently placed under the care of the nurse. Using an on-line medical portal the nurse will launch a query asking whether Bob can be given the sleeping drug Eszopiclone. This query has the following characteristics:

1. Both the subject (Bob) and the object (Eszopiclone) components of the query are globally and uniquely identifiable in the system. The composite query will contain additional parameters for this purpose. For example Bob’s healthcare identifier token can be used to uniquely identify him to the decision support system.
2. The treating healthcare provider is not assumed to possess any special knowledge about either the patient or the treatment itself.

To initiate the query, the nurse enters Bob’s patient identifier information as the subject, selects the drug Eszopiclone as the object (perhaps form a drop down list of all sleeping pills), and selects the action ”can be given” from a list of possible actions (again from a drop down list).

Upon receiving this user query, the communication interface module would proceed to translate it into its semantic (system) query. There are three main components to the semantic query: 1. the object – patient Bob, 2. the subject – drug Eszopiclone, and 3. the verb – can be given.

Each user query is translated into a semantic query. A semantic query is simply defined as a query that can be consumed by a semantic reasoning engine, and is ontology and inference rules aware. For our work, we have used Notation 3 [4, 5] based queries for semantic reasoners and knowledge stores. In the semantic query format, the patient Bob is represented by a unique instance of the Patient concept, and the sleeping medication by an instance of the Drug concept. Once the subject and the object concepts have been identified, the verb/action can then be described by a semantic property that may exist between the object and the subject nodes. The semantic query is then passed along to the query execution engine for processing. Upon completion, the communication interface module will translate the answer tokens (result & semantic proof) into an appropriate representation. The triple form of semantic knowledge representation, which we utilize, has a natural transformation into human language, in that every triple is comprised of a subject, object and verb which can be mapped into simple constrained sentences. Although such simple mappings may not always yield grammatically valid sentences, they should be interpretable.

Such request–response based interfaces are common in knowledge-based systems. For example, [7] describes a similar web-based communication interface for a social semantic patient support application, where patients can form support groups by launching queries against a semantic engine to discover other patients with similar conditions and symptoms.

2.2 Knowledge Management Component – Data Representation

The knowledge management component is a high level component that provides an abstraction on top of local and distributed knowledge stores. Its main responsibility is to aggregate knowledge from diverse sources (that may be relevant in answering a user’s query) locally. The aggregated information is stored in an ontological format.

We use [29]’s generic definition of an ontology as below:

**Definition 1 (Ontology) [29]:** Let $V$ be the set of structured vocabulary, and $A_e$ axioms about $V$, which are formulated in formal language $L$. An ontology
is a sign-system $\text{Out} = L, V, A_x$, where: the symbols of $V$ denote categories, and relations between categories or between their instances; and $L$ is a formal language associated to a vocabulary $V$ and used to declare a set of $L(V) = A_x$, which are usually a declarative formulae. It is also assumed that: $V \subseteq V_1 \Rightarrow L(V) \subseteq L(V_1)$, and $L(L(V) = L(V)$.

Before we proceed further, it is important to state the following assumptions which underly the knowledge management component:

1. The knowledge management component does not aim to define or generate a full problem-specific ontology. The main motivation behind using an ontological knowledge representation is to collect information from heterogeneous sources into a locally defined format, which is suitable for automated query processing.

2. Any ontology that is capable of describing the relevant concepts required for querying the system can be used in the knowledge management component.

3. In the absence of a global domain ontology, the problem may require (static or dynamic) mapping of ontological concepts to and from a local ontology. Ontology mapping has been well studied [21, 32, 14]. However, in the implementation presented in § 3, we aggregate information from three sources using a single ontology. Therefore, we do not require concept mapping.

4. We assume that the ontology (knowledge representation and knowledge inference both) is complete: experts have provided full information. In order to deal with information uncertainty, fuzzy-logic based techniques can be applied to extend an existing core ontology [55] (when the ontology itself is incomplete). Note that this is a different “missing information” issue; we are focused on coping with missing patient data.

The framework provides a plugin based extension mechanism for provisioning and integrating external data sources at run-time. The plugin bridge makes it possible to submit a query and to receive the query results in an ontology based format (irrespective of the native data format of the external data source).

It is important to highlight that our current design diverges from the original one [35, 19], as it does not require pre-processing of non-semantic data-stores into their ontology-based equivalent representation. Holmes requires that if a non-semantic data-store is queried, then the results of the query be translated into an ontology based representation.

\footnote{For future work, it would be interesting to explore how to consider this issue as well within our framework.}
2.3 Knowledge Management Component – Inference Rules

Inference rules describe relationships which exist between different facts in a knowledge-base. Multiple facts can be used in conjunction to assert a rule. Generally a rule will have the form:

\[ \{ f_1, f_2, f_3, \ldots, f_n \} \rightarrow \text{rule} \]

Where \( f_1, \ldots, f_n \) (body) and \( \text{rule} \) (head) are atomic facts in the knowledge-base. Holmes utilizes N3 rule expression syntax, which allows us to create generic rules using variables. For example, we can say that any patient who is over the age of 65 is an elderly patient as follows:

\[
\begin{align*}
\{ & \ ?P \ : \ \text{Patient}. \\
& \ ?P \ : \ \text{hasAge} \ ?AGE. \\
& \ ?AGE \ : \ \text{greaterThan} \ 65, \\
\} & \Rightarrow \{ \ ?P \ : \ \text{hasCondition} : \ \text{Elderly} \ \}.
\end{align*}
\]

N3 Rule Where \( ?P \) is a variable and can be substituted by any patient instance during the rule evaluation.

In the example above, the base knowledge set does not have to explicitly define all patients that are elderly. As long as the patient’s age has been defined, using the above rule we can infer whether a patient is an elderly person. Therefore, inference rules define implicit knowledge. All inference rules used by our proposed framework are capable of working with ontological concepts and can be processed by semantic reasoners.

2.4 Query Execution Engine

At a high level, the query execution engine is responsible for generating answer tokens for the user query. It takes two input parameters (i) user query and (ii) a query context (which can be used to provide additional meta-information about the context regarding the user query). As discussed earlier, the user query can be a complex composite query or it can a simple atomic query.

The main decision making part of the query execution engine utilizes a semantic reasoner. For each user query, the semantic reasoner is responsible for generating two answer tokens (i) a result and (ii) a first order logic proof explaining how the result was obtained (usually in the form of a path traversed to reach the answer from the knowledge graph).

Based on the user query and the query context, a local query specific knowledge set is built using the knowledge management component. That is a composite query is decomposed into atomic sub-queries which are executed against the corresponding data-sets. The results are aggregated in a local semantic knowledge base. Relevant inference rules are also identified at this stage, which provide the required inference and decision making capabilities. The semantic reasoner is then responsible for processing the user query over the knowledge base, using the inference rules to produce the result and the proof (see Figure 2).
In the event that the semantic reasoner cannot determine an answer for a given query, we are faced with two possibilities: (i) unknown result: due to lack of information/facts\(^5\) required by the inference rules, or (ii) negative result: where given a complete knowledge-base, the conditions satisfying the inference rules were not met. An unknown or undefined result appears when a reasoner does not have all the information required to determine the answer to a query. In contrast, a negative result appears when a reasoner has all the information it needs to answer a query, but cannot find a line of reasoning which allows it to produce an answer. For example, suppose that a user asks whether Bob can be given sleeping pills. Consider the following three cases:

- If we know everything about Bob, and there is no evidence that Bob cannot be given sleeping pills, the reasoner returns negative result.

- If we do not know whether or not Bob is depressed, we cannot rule out the possibility that Bob cannot be given sleeping pills even if there is no evidence suggesting that he cannot be given them. The semantic reasoner returns unknown.

- If there are other reasons why Bob should not be given sleeping pills, we do not need to know whether or not Bob is depressed. For example, if Bob is an alcoholic, then we can return a proof that he cannot be given sleeping pills which does not involve knowledge of his depression status. In this case, the semantic reasoner returns an answer to the query and a proof, even if Bob’s depression status is unknown.

This particular distinction is of great interest as it motivates the use of our proposed hybrid model. That is when a semantic reasoner produces an unknown result (due to missing facts), we turn to machine learning inference models to provide probabilistic facts. From these models, we can guess (with some confidence\(^6\)) what the missing values might be like for a particular query.

---

\(^{5}\)This is closely related to the open world assumption of the semantic reasoning where missing information should lead to an unknown result rather than a negative result.

\(^{6}\)Confidence is estimated by treating each predicted value as an independent prediction. More complex Bayesian confidence estimates could also be used, but are challenging to train. We discuss this possibility briefly as a potential avenue for future work in § 7.
This data is inserted into the query specific knowledge base and the semantic reasoner is asked to re-evaluate the query (see Algorithm 1).

3 Proof-of-Concept Implementation Using Real World Datasets

In this section, we will discuss the validation strategy for Holmes architecture using real-world health related datasets. For this validation process, we constructed a proof-of-concept implementation of Holmes, with the goal of studying the following core components: the knowledge management component (see § 2.2) and the query execution engine (see § 2.4). A simplified implementation of the communication interface module utilized Notation 3 [5, 4] to accept user queries and produce result tokens.

A simplified ontology (discussed in § 3.3) was created to describe the relevant information, and to create the knowledge inference rules using N3 representation. We used the Euler Proof Mechanism: EulerSharp[51] as the knowledge inference engine. EulerSharp is a backward chaining inference engine, and provides the capability to distinguish between the unknown answer and negative answer scenarios. In order to build machine learning based inference models, we used Weka [25] to experiment with various popular machine learning algorithms (discussed in detail in § 3.6).

Our experiments were based on real world data drawn from the Center for Disease Control (CDC) Behavioral Risk Factor Surveillance System (BRFSS) telephone survey for 2010. The BRFSS is the world’s largest ongoing health survey, and produces a publicly available dataset each year, which contains results of hundreds of thousands of interviews. The results of each interview are stored as several hundred pieces of demographic and health data [12]. Each piece of data is stored as a numeric attribute or feature of a particular interview record. For example, interview records will all have a feature called ‘Age’ which denotes the respondent’s age. A complete record contains basic demographic information about the respondent (age, race, sex, geographic location, etc.), and then information about a wide range of common medical conditions (e.g. cancer, asthma, mental illness, diabetes), as well as any behavioural risk factors (e.g. alcohol consumption, drug use, sleep deprivation). For each of these conditions and factors, the BRFSS data consists of an initial question (e.g. “Have you ever been told you have asthma by a medical professional?”), and a series of followup questions if applicable (e.g. “Do you still have asthma?”, “How many days have you had trouble breathing in the past month?”, etc.). All possible responses are encoded numerically. The dataset is a representative sample of the health state of residents of the United States, including information about heart disease, diabetes, cancer, depression, and many other diseases.

For our validation strategy, we utilized the following three real-world datasets, and followed a simple line of inquiry - “Which patients can and cannot be given various sleeping pills”.
1. The Center for Disease Control (CDC) Behavioral Risk Factor Surveillan
cystem (BRFSS) telephone survey for 2010. We interpreted the results of
the survey as medical records for hypothetical patients. See below (§ 3.2)
for details on how the raw data was processed and used.

2. A Drug interaction registry [20] to identify which drugs are safe to be
prescribed together.

3. The Mayo Clinic sleep medication prescription protocol [49]. Using the
BRFSS dataset based patient records, the Mayo Clinic sleeping pill pre-
scription protocol served as a model for the creation of knowledge inference
rules.

The BRFSS dataset also matched the following goals identified for our frame-
work validation strategy:

1. The dataset should be representative of real-world data and available with-
out any restriction from a trusted source.

2. The dataset should contain a large number of features to fully explore the
potential of machine learning algorithms. The selected BRFSS dataset
contains 400+ features.

3. The dataset should contain enough information to build a reasonable ont-
ology.

4. The dataset should contain a very high number of patient records, allowing
for decision making scenarios involving sparse data.

We will now describe the construction of our validation strategy in detail.

3.1 Line of Inquiry

We used the real world BRFSS based medical records as a basis for our vali-
dation experiments. In the experiments (discussed in greater detail in the next
section), several systems are asked to determine whether given patients should
be prescribed sleeping pills. The systems involved include OMeD, Holmes, and
a selection of machine learning algorithms, representing semantic approaches
alone, semantic approaches augmented with machine learning, and machine
learning approaches alone. Sleeping pills are prescribed as a treatment for in-
somnia - a condition in which a patient has difficulty falling or staying asleep
[49]. Insomnia is an interesting choice for our line of inquiry, because it is caused
by a wide variety of factors, and because chronic insomnia is often indicative of
an underlying problem which should be addressed instead of merely prescribing
pills [49] [56]. In addition, sleeping pills may interact with existing medical con-
ditions. For example, patients with kidney or liver conditions may not be able
to safely metabolize certain types of sleeping pills [49].
3.2 Ontological Knowledge Representation & Reasoning

In order to provide a working understanding of the (BRFSS) data in the context of our line of inquiry, we created a simplified semantic model focused on insomnia treatment. The model utilizes an ontology to define the core concepts and to describe the organization of these concepts. Figure 3 provides a graphical representation of our constructed ontology.

Our ontological model contains four pivotal concepts described as follows:

1. **Patient** is a central concept that helps describe the various characteristics of individuals found in the BRFSS dataset.
   - We utilize the *SasVariable* concept to capture atomic data values for individuals from the raw data-set. The *Patient* and the *SasVariable* concepts can then be associated using a *patientRecord* relationship to define a complete collection of all BRFSS data for each individual, generating simplified individual patient records. The raw values can be further mapped onto other more meaningful concepts. (In the next section, we will see an example of mapping a raw *SasVariable* value of a datum to either the *Male* or the *Female* concepts).
   - The two *Patient-to-Drug* associations i) *isPrescribed* and ii) *isTaking* describe the drugs that a patient is assumed to be taking. Using these two relationships, we can define all drugs that an individual is currently using.
   - An individual may be suffering from many *diseases*, or may be experiencing various medical and/or non-medical *conditions*. We utilize the *hasDisease* and the *hasCondition* relationships respectively to model these relationships. Again the raw data from the BRFSS dataset is used to map conditions and diseases to higher semantic concepts (such as *Asthma* and *Pregnancy*).

2. **Drug** is another high level concept that represents the different medications used in our line of inquiry.
   - We utilize a hierarchical *subsumption* relationship to describe the “type” of a drug. Although there are more elegant ways to describe the “type” relationship, our simple subsumption based approach is justified based on the fact that we only consider two categories of drugs i) pain medication using the *PainMedication* concept, and ii) sleeping medication using the *SleepingPill* concept.
   - In our system, each individual drug is then simply an instantiation of the respective drug type concept. For example, *Eszopiclone* represents a unique instance of a drug used as a sleeping medication.

---

7It is important to reiterate that this simple ontological model is constructed mainly for the purpose of validating Holmes. For real world deployments, a much more elaborate problem-specific (domain or local) ontological model may be constructed.
The hasContraindication relationship describes three important associations. The first is a drug-to-drug relationship, representing drugs that may not be taken together due to the risk of adverse drug-to-drug interactions. For example, a certain sleeping pill may not be taken along with a certain pain medication. Similarly, we can define a second drug-to-disease association between the Drug and the Disease concepts. That is to say that some drugs cannot be given in the presence of certain diseases. Lastly, we can define the same type of relationship between Drug and Condition concepts, indicating drugs that can not be given under certain medical and non-medical conditions. Figure 6 presents some examples of the hasContraindication associations between Drug, Disease and Condition concepts.

3. A Disease concept represents various medical diseases under investigation. Each unique medical disease is represented using an “is a” concept hierarchy relationship. We have purposely kept this hierarchy flat for ease of use.

4. The Condition concept is utilized to capture things that may not be classified as a disease, such as old age, pregnancy, and so on. Similar to the Disease concept hierarchy, each unique medical and non-medical condition is also represented using an “is a” concept hierarchy relationship.

Knowledge inference rules were created to detect the latent information links between these concepts. As discussed above, the Mayo Clinic sleeping pill prescription protocol defined the medical rules for prescribing various sleep medications. These guidelines were translated into semantic inference rules to support the targeted decision making process.

3.3 Ontology-based Knowledge Representation of BRFSS

The simplified ontology defines the key concepts such as patient, drug, disease, condition and so on. The information from the raw BRFSS dataset was then converted into individuals (belonging to various classes) using the ontological concepts and their data attributes. The concepts helped define the target universe-of-discourse, and allowed us to manage information accordingly. As mentioned earlier, we utilized W3C’s resource description framework (RDF) [57] data representation. We selected RDF’s Notation 3 [4] data representation to create the knowledge store and to write our inference rules in.

Using the Patient class, the raw data rows (from the BRFSS dataset) were transformed into individuals (patients) and their health information (medical records). The BRFSS dataset codebook (provided with the dataset) categorized 403 different attributes for each patient. The codebook also defined the semantics of the different values each attribute could be assigned. For example, the
Sex attribute could be male or female. Figure 4 displays the raw N3 facts regarding a particular patient instance (patient224348) mined from the BRFSS dataset. Figure 5 depicts how the knowledge mining process takes place by applying inference rules to the ontology based data. For example, the patient has a relationship hasSasVariableSEX with a attribute SEX\(^9\). From the raw BRFSS dataset, we record this value as 2. The translation of the raw value ‘2’ into a semantic class Female is defined by the inference rule hasSex in accordance with the BRFSS data codebook. For the semantic model, we transformed the insomnia related attributes into semantic medical records for all patients.

Table 1 provides a summary of the main rules utilized for this information transformation. For a detailed listing of these rules, please refer to appendix-B.

The above mentioned rules provide a sophisticated mapping framework for translating raw data into Patient records/facts (defined using hasSasVariableXXX relationships) into richer semantic concepts and relationships.

### 3.4 Additional Data Sources

It is important to realize that a medical decision support system like Holmes should provide the ability to work with multiple supporting datasets. In order to demonstrate this, we chose to model a subset of a drug-to-drug interaction registry [20]. In light of our validation strategy, we limited the drug-to-drug interactions to just two groups of drugs, pain medications and sleeping pills.

The BRFSS dataset does not contain information about whether or not a patient has been prescribed pain medication, so we augmented the data with this information. Pain medications were prescribed to patients who reported experiencing pain in the BRFSS survey. A selected few pain medications were instantiated in the knowledge base along with their drug-to-drug interactions (see Figure 6 for an example of how the various drug related associations were mapped). We assigned all pain medications in roughly equal proportions, using a uniform random number generator to decide which drug a particular patient should be assigned.

### 3.5 Inference-based Decision Making for Prescribing Sleep Medication

In the previous two sections we demonstrated how to represent raw BRFSS data as individuals (patients) with attributes (medical records). We then explained how to apply simple inference rules to map raw patient information to more complex concepts, and finally, we discussed how to integrate an external drug-to-drug knowledge store with our patient information knowledge base.

In this section, we will demonstrate that this information can be mined for medical decision support activities. Recall that the main question we would like to ask the system is whether a patient can or cannot be given a sleeping

\(^9\)The SasVariable prefix is an artifact of the conversion process for the BRFSS data and can be ignored.
medication based on the patient’s medical records. We define the following three simple cannotBeGiven rules in order to facilitate this decision making process (refer to appendix-C for the actual rules in N3 format).

- If a patient is taking an existing drug (D1), and D1 is contraindicated by another drug D2, then drug D2 should not be prescribed to the patient.
- If a patient has a condition that is contraindicated by a drug, then the patient should not be given the drug.
- If a patient has a disease and the drug is contraindicated in the presence of the disease, then the drug cannot be given to patient.

We can now launch the following query against our knowledge base. This query asks what drugs cannot be given to which patients.

\[ \text{:WHO :cannotBeGiven :DRUG.} \]

We utilize a semantic reasoner (Euler [51] to traverse the various knowledge graphs and to find an answer to the user query. The semantic reasoner provides us with two important artifacts (i) an answer to the query (:Patient224348 :cannotBeGiven :Eszopiclone) and (ii) a justification (knowledge graph) as to how the answer was realized.

\[
\{\text{:Patient224348 a :Patient} \text{ e : evidence <file:knowledge-base.n3#.1>}. \}
\{\text{:Patient224348 a :Patient} \text{ e : evidence <file:knowledge-base.n3#.1>}. \}
\{\text{:Patient224348 hasSasVariableAGE 66} \text{ e : evidence <file:knowledge-base#.1>}. \}
\]
\[
\{\text{:Patient224348 hasAge 66} \text{ e : evidence <file:rules.n3#.5>}. \}
\{\text{:Patient224348 hasCondition :Elderly} \text{ e : evidence <file:rules.n3#.94>}. \}
\{\text{:Patient224348 cannotBeGiven :Eszopiclone} \text{ e : evidence <file:drug_rules.n3#.30>}. \}
\]
\[
\{\text{:Patient224348 a :Patient} \text{ e : evidence <file:knowledge-base#.1>}. \}
\{\text{:Patient224348 a :Patient} \text{ e : evidence <file:knowledge-base#.1>}. \}
\{\text{:Patient224348 hasSasVariableAGE 66} \text{ e : evidence <file:knowledge-base#.1>}. \}
\]
\[
\{\text{:Patient224348 hasAge 66} \text{ e : evidence <file:rules.n3#.5>}. \}
\{\text{:Patient224348 hasCondition :Elderly} \text{ e : evidence <file:rules.n3#.94>}. \}
\{\text{:Patient224348 cannotBeGiven :Estazolam} \text{ e : evidence <file:drug_rules.n3#.30>}. \}
\]
\[
\{\text{:Patient224348 a :Patient} \text{ e : evidence <file:knowledge-base#.1>}. \}
\{\text{:Patient224348 a :Patient} \text{ e : evidence <file:knowledge-base#.1>}. \}
\{\text{:Patient224348 hasSasVariableSLEPSNOR 1} \text{ e : evidence <file:knowledge-base#.5>}. \}
\]
\[
\{\text{:Patient224348 hasDisease :SleepApnea} \text{ e : evidence <file:rules.n3#.32>}. \}
\{\text{:Patient224348 cannotBeGiven :Ramelteon} \text{ e : evidence <file:drug_rules.n3#.41>}. \}
\]

For this particular example, the semantic reasoner has found evidence that Patient224348 cannot be given Eszopiclone or Estazolam (based on patient’s age), and Ramelteon (based on the fact that the patient suffers from sleep apnea).
3.6 Dealing with Missing Data - Machine Learning based Inference Models

There may be missing values in our patient data set. For example, a patient may fail to report her age. Or a patient may not report that he is a smoker. It is often important to ascertain this missing data in order to properly determine whether or not to prescribe a drug. In cases where an inference rule regarding the prescription of a drug requires knowing a feature value that is missing, we employ machine learning methods to predict the value of that feature for this patient, based on what other patients in the data set with similar features have had recorded for this value. For example, suppose it is important to know a patient’s age, in order to resolve whether to prescribe the sleeping medication Estazolam. We reduce our search to a classification of the patient’s age as either elderly or not (i.e. > 65 or not).

This motivates the construction of the machine learning component for Holmes. Construction begins by obtaining a selection of numerically represented records. In this case, since the records all come from a common numeric source and are augmented with only one extra variable (the pain medications discussed above), this is quite straightforward. The resulting dataset is called, by convention, the training set. The training set will have some of the records from the main dataset (but does not need all of them), and each record will have all the attributes present in the main set. If a record does not have a value for a particular attribute in the main set, the value is recorded as ‘?’ in the training set. For example, a responder to the BRFSS who refused to report their age would have a value of ‘?’ for the Age attribute in the training set.

For each attribute in the training set, a predictive model is constructed using a machine learning technique (any technique will do - some will perform better than others on any given dataset). The resulting models are sometimes called classifiers. The classifier for a particular attribute will accept as input an arbitrary record in the same format as the training set. If the attribute this classifier is intended to predict is set to ‘?’ in the record, the classifier predicts what the value really is, based on the other attributes in the record. For example, if Bob has many of the symptoms of depression recorded, but no record of whether or not he has depression, the classifier for depression would likely predict that Bob is depressed.

We note that since this system works by comparing Bob to patient records seen in the training set, the model may make mistakes by using correlations to make predictions. For example, lack of exercise might correlate with being overweight, but if Bob has recently recovered from cancer then it is unlikely that he is overweight, even if he is not exercising. This fact tends to make machine learning models unsuitable for direct use as medical decision support systems, as we demonstrate with experiments below. In direct use, classifiers are trained to predict whether or not a patient should be given sleeping pills, rather than just being used to fill in missing attributes. The sleeping pills model is highly non-linear, and many attributes in the dataset are weakly correlated with whether or not a patient should be given sleeping pills. Consequently,
the machine learning models we apply to this task tend to perform poorly by themselves.

4 Experiments

This section is divided into a discussion of the experiments and the data preprocessing, of the three experiments we conducted, and of the techniques we used to analyze our results.

We aim to show that the Holmes architecture is effective in supporting medical decision making for a range of possible scenarios. This is achieved by demonstrating Holmes in operation with the specific collection of data sets described earlier, namely the BRFSS dataset for patient information, Mayo Clinic sleep medication prescription protocol for generating inference rules, and an online drug-to-drug interaction registry for describing which drugs can be prescribed together.

For the validation of Holmes, we need to select a specific machine learning method to employ that is appropriate for these data sets. Our first experiment serves that purpose, by determining the effectiveness of each of a variety of machine learning methods; we also identify the appropriate size of training data for these machine learning methods to use.

In examining the machine learning methods, we also demonstrate that using machine learning alone for our medical decision making has important shortcomings. This is clarified first for the case where there is complete data (as part of our first experiment) and then in a second experiment for the case where there is missing data. We show that these methods are insensitive to the noise in the data but their performance is poor even in the absence of noise. This leads to our third experiment where the machine learning techniques are integrated into the Holmes framework. In the presence of noisy data, Holmes outperforms the purely semantic model, confirming its value.

To perform experiments 1 and 2, we augmented each record from the BRFSS dataset with a label indicating whether or not a person with the corresponding health profile should be given sleeping pills if asked, based on the predictions made by OMeD (i.e. the semantic model discussed in the previous section). The data are incomplete, so a small number of records could not be labeled by the model and were discarded. After this preprocessing, the labeled dataset contained just over 275,000 records. In experiments 1 and 2, we created randomized class-balanced training sets [60][18]. These are subsets of the data which contain equal numbers of patients who should and should not be given each type of sleeping pill. The technique is used for imbalanced data, such as the labeled BRFSS set, where one class (i.e. one value of the attribute we are building a model to predict) appears much more often than other values. In such sets, many machine learning algorithms produce degenerate classifiers, which do not look at any attributes at all, and instead label everything as being from the dominant class.
4.1 Experiment 1: Limited Data

It is well known that, as the number of features in a dataset grows relative to the number of data points, statistical techniques for fitting the data will tend to produce worse approximations of the correct model. In our previous research, we highlighted this fact in comparisons with semantic methods (e.g. [19]). In the first experiment below, we analyze the performance of several machine learning approaches as the number of patient records used in the training data varies. The purpose of this experiment is twofold. First, we expect to find machine learning methods wanting as predictors for whether or not patients should be given sleeping pills. Second, we hope to gain some calibration information for subsequent experiments. Specifically, we hope to learn which machine learning algorithms perform well on the BRFSS data, and how much data we need to use to train them.

In the experiment, we trained four machine learning algorithms on 50 different training sets constructed as described above. The algorithms built a model for each of the 7 different types of sleeping pills, and attempted to predict the values for all the records which they had not been trained on. We evaluated the classifiers with datasets containing 5,000 exemplars, and again with datasets containing 2,500 exemplars. A training set size larger than 5,000 points was not computationally feasible for use in later experiments, so larger values were not considered. For each size, we generated 50 training sets using different random seeds to select training records. We used an information gain based feature selection algorithm [63] to reduce each set to its 30 most informative attributes. We then trained a model using each algorithm on each data set, and evaluated each resulting model across all data not included in the model’s training partition.

We used four algorithms based around decision trees, with increasing degrees of complexity. Our first method was the infamous decision stump, an incredibly simple classification method pioneered by Holte, which nonetheless has been demonstrated [27] to perform well in practice. Decision stump picks the single attribute in the dataset to base all its predictions upon which maximizes the accuracy of the resulting model - a single layer decision tree. For example, we might expect number of hours slept per night to be the single most highly correlated attribute with respect to whether or not a patient should be prescribed sleeping pills, so the decision stump would base its predictions solely on this attribute. Decision Stump represents a baseline level of performance and an indication of the inherent difficulty of a problem. The second method we used was the class C4.5 decision tree algorithm. This widely used algorithm is capable of generating efficient and powerful classifiers, even for non-linear or partially missing data [46]. In particular, we used a java-based implementation of C4.5, release 8, called J48. The final two methods are actually meta-classification algorithms, which combine the results of a large number of simple classifiers to produce a label. Bootstrap Aggregation (Bagging) trains many separate classifiers on bootstrapped samples of the provided dataset (i.e. subsamples drawn with replacement). When labeling training data, the classifiers each produce
their preferred label, which is used as a “vote” to determine the final label of the meta-classifier [8]. Boosting is the process of building a meta-classifier by iteratively adding a new weak classifier which focuses on the points in the data on which the model currently performs poorly. Often the resulting meta-classifier can substantially outperform a single strong classifier trained directly from the data. We used the popular AdaBoost boosting algorithm in this work [22].

We used the implementations of these algorithms found in the Weka Machine Learning Toolkit [25]. We used REP trees (A decision tree algorithm similar to C4.5, but which foregoes repeated sorting of attributes to improve speed) as the underlying classifier for bagging, and decision stump as the underlying classifier for boosting. In both cases, these are the default classifiers suggested by the meta-classification algorithms.

4.2 Experiment 2: Noisy Data

Experiment 2 examined the application of both OMeD (Holmes without machine learning) and the machine learning techniques to records where important data are missing. For example, a patient’s diabetic status may be unknown if no tests for that status have been administered. In these situations, semantic techniques may perform poorly because they are based upon formal reasoning which cannot contend with flawed data. The machine learning techniques may also perform less well than would with correct data, but are expected to cope better because of their ability to produce a statistically accurate result based on correlations. For example, Ambien should not be given to patients with a history of respiratory problems [49], but in the presence of noisy data machine learning techniques may use other features like a patient’s smoking habits when that data are missing. Although smoking may eventually cause respiratory problems, being a smoker does not directly imply that a patient presently has a history of respiratory problems, and so this correlation cannot be encoded nicely as a semantic rule.

In the experiment, we ensure the existence of ground truth for the missing data by using the following procedure. We first “fuzz” the data by introducing random omissions from the patient records. A parameter $\epsilon$ determines the probability that the value of an attribute important to the knowledge-based decision making process has been altered. Altered attributes are changed to “?”. $\epsilon$ is varied in the experiment from an average of $\frac{1}{10}$ errors per record to an average of 6 errors per record. After fuzzing, we drew 50 random subsamples of 5000 exemplars from the fuzzed BRFSS dataset and assigned them the values for whether or not they should be given sleeping pills based on the corresponding labels in the noise-free dataset. We then applied the feature selection algorithm mentioned in experiment 1 to produce a dataset small enough to process efficiently. This simulates a situation wherein some of the patient records have known outcomes though the data are noisy, as might be the case when training on historical data. We selected AdaBoost from the learning algorithms used in experiment 1 because of its comparatively good performance on the noise-free data, and trained a model for each of the subsets. Each model was then evalu-
ated on those exemplars which were not part of its original dataset. The OMeD was also provided with the complete noised dataset and asked to predict which sleeping pills (if any) each of the patients should be given. The results produced by OMeD appear with the results of experiment 3 for ease of comparison.

4.3 Experiment 3: Augmentation Techniques

Experiment three is the validation of the primary contributions made by this paper, and examines a method of combining machine learning techniques with OMeD to facilitate automated decision support when the available pool of training exemplars is small and the data are noisy. The method involves using OMeD to perform reasoning, and then using simple classifiers to predict the values of any important attributes which are missing, using a small subset of training data and the other attributes in the dataset. This technique will perform identically to OMeD when all pertinent features are present and correct in the data, but might perform better when the data are noisy since Holmes will be able to report a decision, and, based on the classifier confidence associated with the predicted values of the missing attributes, a probability of its decision being valid. We used AdaBoost as the machine learning algorithm for generating the predictive models used by Holmes (see Algorithm 1).

For each of the noisy datasets in experiment 2, we generated a training dataset for each of the attributes used in the semantic model, comprised of all the patient records where a value for that attribute was present. Thus, for example, a classifier was trained to predict whether or not a patient was over 65 based on all the other attributes in the data, and a separate classifier was trained to predict whether or not a patient had asthma, also based on all the other attributes in the data. When the semantic model required a missing value to be filled, it simply asked the classifier corresponding to the missing attribute to make a prediction based on the patient information which was available. The model returned both a predicted value and an associated confidence which roughly corresponds to the probability that the classifier’s prediction is correct. When issuing its decision, the semantic model can compute an estimate of the probability that its reasoning is sound by treating each prediction as an independent event and multiplying the associated probabilities.

4.4 Analysis

We labeled patients who should be given sleeping pills “positive” exemplars, and those who should not, “negative” exemplars. When a method assigned the correct label, it produced a “true positive” (tp) or “true negative” (tn). Otherwise, it produced a “false positive” (fp) or “false negative” (fn). We evaluated our results in terms of specificity and sensitivity, and ‘balanced accuracy’, which is a simple average of the two [10]:

\[ balAcc = \frac{Spec + Sens}{2} \]
Specificity is often used as a medical diagnostic measure, formally defined as

\[ Spec = \frac{tn}{tn + fp} \]

or the proportion of those labeled as healthy who are actually healthy. Sensitivity is the other side of the coin, defined as

\[ Sens = \frac{tp}{tp + fn} \]

or the proportion of all those who are not healthy and are labeled by the classifier as unhealthy. Intuitively, a system with high sensitivity will report the presence of an ailment if it is present. A system with high specificity will report that the ailment is not present if it is not. A theoretically perfect classifier is one which is both perfectly specific and perfectly sensitive. In practice, there often is a substantial trade off between these two measures.

5 Results

Throughout this section, statistical significance was determined according to the following procedure. First, all data under consideration was examined to determine whether it conformed well to the assumptions required by parametric tests of significance. If the data agreed with these assumptions, we used pairwise t-tests to perform the comparisons. If the data were inconsistent with these assumptions we used pairwise Wilcoxon Rank Sum tests instead. All p-values were adjusted for multiple hypothesis testing, where appropriate, using Holm's method. Analysis was conducted using R [47] and figures were generated using the vioplot package [2] 10

5.1 Experiment 1

The results from experiment 1 are summarized in figures 7 to 10, which depict the distribution of balanced accuracy on test data for each of the four classifiers we evaluated under training data sets containing 5,000 and 2,500 exemplars respectively. Pairwise student t-tests found a significant decline in mean test performance for AdaBoost and J48 when the training set size was reduced, but Bagging and Decision Stump were unaffected. This is borne out by an examination of the training and test performances, which suggest that Bagging is overfitting the data, meaning that further data is not expected to be useful. Decision stump builds a model based on only one attribute, so it is not expected to need much data in any case. Overall, AdaBoost has the best test performance, significantly outperforming all other methods at both training set sizes. Consequently, we selected AdaBoost for use in future experiments. This

---

10 Violin plots depict a classic box-and-whisker plot with a rotated kernel density plot (smoothed histogram) overlaid. These comparatively new plots provide an easily readable summary of the shape of a distribution as well rank statistics.
also required us to use the larger training set size, due to AdaBoost’s demonstrated sensitivity to this parameter. We note, however, that the magnitude of the improvement received by AdaBoost when the data is doubled is actually very small, with a 95% confidence interval of (0.0038 0.0059).

From these results, we can conclude that the problem posed represents a substantial challenge for machine learning techniques, as evidenced by their very low rates of test accuracy. A possible reason for this is that the rules which generated the problem (i.e. the rules in the semantic model) contain many non-linear relationships. For example, many of the semantic rules take the form of step functions combined with conjunctive operators, which may not be at all obvious when attempting to build the rules from the data with a machine learning algorithm.

We also conclude that, due to the observed sensitivity to dataset size, we will have to use larger training sets wherever computationally feasible. We observed approximately quadratic run time increases with dataset size, making this a significant issue.

5.2 Experiment 2

The results for experiment 2 are summarized in figures 11 and 12, which depict the distribution of balanced accuracy on training and test data respectively for the AdaBoost classifier selected on the basis of performance in experiment 1. The distributions are approximately normal and t-tests were used in evaluation. We found no statistically significant differences in performance as $\epsilon$ increased, and also found no significant decline in performance between training and test evaluations, although the mean decline was around 0.01. This suggests that AdaBoost is not overfitting (i.e. fitting models to noise), but is instead picking out useful correlations between attributes to build its models. In spite of this, the complex nature of the problem, and to a lesser extent, the limitations on available computational power, prevent good models from being generated. This is visible in the graphs as a very small drop in the y-axis of the test performance plot relative to the training performance plot. A larger drop would suggest overfitting.

5.3 Experiment 3

The results of our third experiment (see § 4.3 for details on the experimental setup) are summarized in Tables 2 and 3. The tables summarize the results corresponding to the highest four levels of injected noise ($\epsilon$). The table headings are described in section 4.4, except $tpr$, $fpr$, $tnr$, and $fnr$, which are the true positive, false positive, true negative and false negative rates respectively. They are obtained by normalizing the number of records in each category by the total number of records with a particular label. For example, $tpr = \frac{\#tp}{\#p}$, i.e. the fraction of all positive exemplars which a method labels as positive. $fpr = \frac{\#fp}{\#n}$
As seen in table 2, Holmes does experience some degradation in its performance as noise increases, but this degradation is small. An increase of 5 percentage points in the noise factor causes a decrease in performance of less than 1 percentage point. In contrast, OMeD experiences a decrease in performance of around 4 percentage points as noise increases. Both semantic methods outperform AdaBoost (the pure machine learning based method whose results are shown in experiment 2 above). It appears that Holmes has greater difficulty controlling for false negatives than for false positives, but this bias could be counteracted (or increased) in practice if desired by adjusting the parameters used to train the probabilistic models underlying Holmes.

From these results, we conclude that Holmes substantially outperforms both OMeD and our leading non-semantic algorithm (AdaBoost) in making predictions under noise, justifying Holmes as a decision support system for real world applications.

6 Related Work

In medicine, decision support systems are utilized in different areas and in different capacities [3]. It has been shown that decision support systems help provide an enhanced level of patient care [24, 28, 34]. Several dimensions of a clinical decision support system are important to consider, including timing of the decision, active or passive role of the system, and ease of use [44]. In addition, [3] provide us with two major classifications of clinical decision support systems, knowledge-based and nonknowledge-based. Holmes represents a decision support system that can be used either in an active or a passive capacity for various stages of a decision making process.

As noted above, clinical decision support systems are either knowledge-based or nonknowledge-based. Each classification has its own set of strengths and weaknesses. For example, the knowledge-based systems translate domain expert knowledge into inference rules for decision making. In the presence of complete information, this process is quite robust and powerful. However, in the absence of information, the effectiveness of rule-based inference degrades quite rapidly [3]. Furthermore, the rule-sets are difficult to manage and evolve, and require input from domain experts [38]. In contrast, nonknowledge-based systems generally apply machine learning algorithms to provide the inference capability that does not depend on explicitly defined rules, and is capable of working with incomplete datasets [44]. However, the inference models generated by these systems require time-consuming training and are scenario specific. Also, the models do not provide a mechanism to validate decisions made by the system [3]. Holmes provides a very specific construction for combining knowledge-based and nonknowledge-based approaches in a single system design. The hybrid design merges the two classifications, offsetting the respective weaknesses of each and building on their strengths.

Given the large volume of available patient information (of different modalities), decision support systems should not be limited to a single knowledge base.
In our design, the use of ontology based knowledge representation aims to provide a mechanism to integrate information from various heterogeneous knowledge repositories. Furthermore, [1] highlight the fact that medical decisions should be grounded in both evidence and expert experience. Providing consistency across both these dimensions is not an easy task to solve. Holmes primarily utilizes semantic reasoning to provide evidence for each system made decision. This is done using first order logic proof mechanisms that can be easily validated.

The interest in using machine learning techniques for clinical decision making is on the rise [26, 43]. For example, [64] explored the use of machine learning algorithms in a geriatric patient rehabilitation setting. They provided a comparative evaluation of two machine learning techniques against the existing decision making process (using only a clinical assessment protocol-CAP). Their results demonstrated a definite advantage of using machine learning algorithms. However, they noted that the machine learning techniques (alone) produced more false positives and false negatives. Furthermore, the machine learning results were “less readily interpretable” [64].

[23] presented a different approach where a knowledge-based expert system was created to provide case-based reasoning capabilities. They transformed raw patient data into patient cases, and then provided inference rules to perform “near-match” search queries. Their particular construction is different from Holmes as they only utilized statistical analysis (of the raw data) to determine weights for ranking the results.

7 Future Work

Holmes provides a combination of machine learning based classifiers with the semantic reasoners. However, as mentioned above, the framework itself does not place any restrictions on which machine learning inference models can be used. A possible extension to this work can explore the use of other machine learning based inference techniques. For example, association mining can be used to explore latent relationships between different features of the datasets. As it is not feasible to explicitly define a rule for every possible relationship that may exist, in the absence of a specific inference rule, probabilistic inference rules can be utilized.

Our patient (BRFSS) dataset is multi-dimensional in the sense that it provides many features across many different aspects of patient health. Our choice of this dataset was motivated by the fact that medical records (for an individual patient) will contain information reflecting various aspects of patient health. Although this dataset is reflective of patient medical records, this assumption might be too strong for decision support systems for a particular modality. Therefore, the use of patient information limited to a single modality might produce more accurate machine learning inference models for a decision support system for that modality.

In our construction, the machine learning models were created using pre-
processing. As medical information is temporal (in the sense that older information is not replaced by new information, rather new information is aggregated on top of the old information), it would be interesting to consider machine-learning models that can deal with incremental addition of new patient information. As these models are time consuming to build, an incremental approach to update pre-built machine learning models could offer significant advantages.

Holmes’ hybrid design places semantic (knowledge-based) inference at the heart of the decision making process. However, there may be other cases where rule-based inference may not be suitable due to uncertainties in the decision making process. For example, decision making in internal medicine is quite challenging due to many uncertainties that exist in the diagnosis. For such scenarios, it may be better to use the machine learning inference models as primary models (backed up by rule-based inference) for the decision making process.

We envision that the Holmes architecture can be instantiated for various other medical scenarios as well (such as emergency medicine, first response care, etc.). In the example of an emergency response team (ERT) member needing to stabilize a patient in a finite amount of time, there are two major hurdles that determine the quality of the final decision. The first challenge is a knowledge problem reflecting the expertise and the training of the ERT person. The allocated amount of finite time for decision making represents the second temporal problem. Holmes provides an answer for both these challenges. By using many knowledge-sets and inference rules, we get a decision that is not dependent on locally available expertise. It would be interesting to see how our approach can be extended to work with other similar situations.

Finally, it might be interesting to examine a more complex approach to multi-labeled classification when estimating the confidence associated with a Holmes query. One principled approach could be the use of multi-labeled maximum entropy models, which can estimate the joint probability of several attributes adopting particular values [65]. Bayesian approaches may also be considered.

8 Conclusion

In this paper we presented Holmes, a hybrid framework for automated clinical decision support systems. We demonstrated that the performance of knowledge-based decision making systems (like OMeD) degrades rapidly in the absence of complete information. The Holmes hybrid framework combined a knowledge-based approach with machine learning to produce a noise tolerant semantic decision support system. Being able to cope with missing information is indeed a critical concern for current medical decision making systems due to the fact that patients are known to misreport or to fail to report their current conditions. That we are able to provide a platform that addresses this particular concern is thus significant.

In order to validate the proposed hybrid design, we provided a specialized construction where the core reasoning process utilized knowledge-based decision
making, and the machine learning models provided predictions using problem specific classifiers. It is important to note that our framework is capable of using other data mining techniques as well, and is not limited to the use of classifiers only. For example, association mining could be used.

In Holmes, we presented a design enhancement (over OMeD’s framework) eliminating the need for entire datasets to be pre-processed. As a result, Holmes is capable of working with many different datasets, and it allows for knowledge exchange to take place in real-time. This is especially important for scenarios such as emergency medical assistance, discussed at greater length as a valuable setting for medical decision making in our previous work [19, 35].

The ability to engage a wide array of information sources leads to improved decision making. Holmes hybrid approach was validated against real world datasets, where the CDC-BRFSS data was used as the patient data, and the Mayo Clinic sleep medication prescription guidelines were used to create decision making rules along with a drug-to-drug interaction registry. Our validation strategy involved asking what sleep medications can be given to which patients. For this, we first established a base-line using purely knowledge-based decision making, and then injected various levels of noise (by removing parts of the patient records) to show the degradation in the decision making capabilities of the knowledge-based only (OMeD) systems. Then using Holmes hybrid approach to decision making, we displayed the effectiveness of using machine learning models in conjunction with OMeD.

Although our line of inquiry was focused on a simple scenario, we have demonstrated the usefulness of our hybrid model. We believe that our proposed approach is very much applicable for other decision making scenarios where constraints (on time, information and knowledge) might limit the usefulness of traditional decision making processes. As such, we are offering a detailed framework for medical decision making that has valuable potential for addressing central challenges that are at issue for current personalized treatment recommendation systems.
ALGORITHM 1: Holmes Decision Making Algorithm

Input: query: An initial user query in a machine processable format.
Input: KB: A knowledge base containing all known facts required by the query.
Input: rules: A set of inference rules defining implicit knowledge for KB.
Input: reasoner: A semantic reasoning engine
Input: recommender: A set of predictive models for each type of fact in KB generated using machine learning over KB.

Output: result - answer to the user query or negative
Output: proof - a first order logic proof describing how result was obtained
Output: conf - confidence in the result obtained.

// Attempt to answer the query directly.
SET response[r, p] = reasoner.doProof(query, KB, rules);
SET ResultFound = response[r, p] not empty;
// If we were successful, return the result.
if ResultFound then
  SET result = response.r;
  SET proof = response.p;
  SET conf = 1.0;
  RETURN (result, proof, conf);
end

// Otherwise, determine why the result failed
SET NoResult = reasoner.inspectForFalseModel(proof);
SET UnknownResult = reasoner.inspectForCounterModel(proof);
// If we got a negative result, just return that.
if NoResult and not UnknownResult then
  SET result = null;
  SET proof = null;
  SET conf = 1.0;
  RETURN (result, proof, conf);
end

// If a response was not found because vital facts are missing from the knowledge base
if UnknownResult then
  LET r be the subset of vital facts in KB which reasoner is missing.
  LET conf = 1
  // Use the predictive models to estimate the values of the missing attributes
  for each attribute a in from r: do
    LET {guess, confidence} = recommender.predict(a given KB)
    // Estimate the probability that all these guesses are correct
    SET conf = conf * confidence;
    SET r[a] = guess;
  end
  // Perform reasoning again, with the guesses filled in.
  SET response[r, p] = reasoner.doProof(query, r, rules);
  // Return the result of the second reasoning attempt.
  SET result = response.r;
  SET proof = response.p;
  RETURN (result, proof, conf);
end
Figure 3: Ontological Model for Insomnia: There are four main concepts: i) patient, ii) drug, iii) disease and iv) condition. The model represents relationships important to our line of inquiry; such as patient-to-drug, patient-to-disease, drug-to-drug, drug-to-disease types of associations. These core concepts and their relationships are used to define knowledge inference rules for decision making.
Figure 4: Semantic N3 based representation of raw patient information

Figure 5: Ontology graph based representation of raw patient information along with inference based knowledge translation.
Table 1: Knowledge Inference & Translation Rules

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hasAge</td>
<td>Translates the attribute AGE into a semantic data attribute for the patient class.</td>
</tr>
<tr>
<td>hasSex</td>
<td>Translates the attribute SEX into a semantic Patient hasSex [Male, Female] relationship</td>
</tr>
<tr>
<td>hasDisease X</td>
<td>Discovers if a patient has disease instance relationship based on the value of the attribute and the clinical diagnoses rules. These rules help identify things like if a patient has sleep apnea, or a liver disease or a lung disease etc.</td>
</tr>
<tr>
<td>hasCondition X</td>
<td>Discovers if a patient instance has a relationship with a condition instance based on the value of the appropriate attribute. These rules help identify if a patient is pregnant, or is breast feeding, or is an elderly person.</td>
</tr>
</tbody>
</table>

Figure 6: hasContraindication and prescribedFor relationships (with drugs, conditions, and diseases) for Eszopiclone & Ramelteon are displayed.

Table 2: Results for Holmes in Experiment 3

<table>
<thead>
<tr>
<th>ϵ</th>
<th>tpr</th>
<th>fpr</th>
<th>tnr</th>
<th>fnr</th>
<th>Spec</th>
<th>Sens</th>
<th>balAcc</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00 (no noise)</td>
<td>100%</td>
<td>0%</td>
<td>100%</td>
<td>0%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>0.15</td>
<td>96.38%</td>
<td>1.42%</td>
<td>98.58%</td>
<td>3.62%</td>
<td>98.58%</td>
<td>96.38%</td>
<td>97.48%</td>
</tr>
<tr>
<td>0.20</td>
<td>95.07%</td>
<td>1.54%</td>
<td>98.46%</td>
<td>4.93%</td>
<td>98.46%</td>
<td>95.07%</td>
<td>96.77%</td>
</tr>
<tr>
<td>0.25</td>
<td>93.82%</td>
<td>1.63%</td>
<td>98.37%</td>
<td>6.18%</td>
<td>98.37%</td>
<td>93.82%</td>
<td>96.10%</td>
</tr>
<tr>
<td>0.30</td>
<td>92.62%</td>
<td>1.75%</td>
<td>98.25%</td>
<td>7.38%</td>
<td>98.25%</td>
<td>92.62%</td>
<td>95.43%</td>
</tr>
</tbody>
</table>
Training Performance with 2500 Training Exemplars

Figure 7: Violin plots showing the distribution of balanced accuracy (y-axis) on training data for the four different machine learning models (x-axis) trained on 2,500 exemplars.

Test Performance with 2500 Training Exemplars

Figure 8: Violin plots showing the distribution of balanced accuracy (y-axis) on test data for the four different machine learning models (x-axis) trained on 2,500 exemplars.
Figure 9: Violin plots showing the distribution of balanced accuracy (y-axis) on training data for the four different machine learning models (x-axis) trained on 5,000 exemplars.

Figure 10: Violin plots showing the distribution of balanced accuracy (y-axis) on test data for the four different machine learning models (x-axis) trained on 5,000 exemplars.
Figure 11: Violin plots showing the distribution of balanced accuracy over training data for classifiers generated by AdaBoost in experiment 2. The x-axis shows various noising rates applied to the data while the y-axis shows performance. The distributions are all statistically identical, demonstrating that the classification algorithm is insensitive to this parameter.

Table 3: Results for OMeD in Experiment 3

<table>
<thead>
<tr>
<th>$\epsilon$</th>
<th>tpr</th>
<th>fpr</th>
<th>tnr</th>
<th>fnr</th>
<th>Spec</th>
<th>Sens</th>
<th>balAcc</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00 (no noise)</td>
<td>100%</td>
<td>0%</td>
<td>100%</td>
<td>0%</td>
<td>100%</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>0.15</td>
<td>88.69%</td>
<td>10.49%</td>
<td>89.51%</td>
<td>11.31%</td>
<td>89.51%</td>
<td>88.69%</td>
<td>89.10%</td>
</tr>
<tr>
<td>0.20</td>
<td>84.65%</td>
<td>14.38%</td>
<td>85.62%</td>
<td>15.35%</td>
<td>85.62%</td>
<td>84.65%</td>
<td>85.14%</td>
</tr>
<tr>
<td>0.25</td>
<td>80.54%</td>
<td>18.27%</td>
<td>81.73%</td>
<td>19.46%</td>
<td>81.73%</td>
<td>80.54%</td>
<td>81.14%</td>
</tr>
<tr>
<td>0.30</td>
<td>76.56%</td>
<td>22.29%</td>
<td>77.71%</td>
<td>23.44%</td>
<td>77.71%</td>
<td>76.56%</td>
<td>77.14%</td>
</tr>
</tbody>
</table>
Figure 12: Violin plots showing the distribution of balanced accuracy over training data for classifiers generated by AdaBoost in experiment 2. The x-axis shows various noising rates applied to the data while the y-axis shows performance. The distributions are all statistically identical, demonstrating that the classification algorithm is insensitive to this parameter.
References


40


A Mayo Clinic – Sleeping Pills Prescription Protocol

We modelled the following subset of sleeping pills for our investigation. For each medication, we utilized the Mayo Clinic sleeping pill prescription protocol as follows:

Table 4: Drugs and Safety

<table>
<thead>
<tr>
<th>Sleeping Pill</th>
<th>Prescription Protocol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Eszopiclone (Lunesta)</td>
<td>Have a history of drug or alcohol abuse, depression, lung disease, or a condition that affects metabolism.</td>
</tr>
<tr>
<td>Ramelteon (Rozerem)</td>
<td>Are pregnant or breast-feeding. Have a history of kidney or respiratory problems, sleep apnea, or depression. Have a liver disease.</td>
</tr>
<tr>
<td>Triazolam (Halcion)</td>
<td>Are pregnant or breast-feeding. Have a history of drug abuse, depression or respiratory conditions.</td>
</tr>
<tr>
<td>Zaleplon (Sonata)</td>
<td>Have severe liver problems. Are pregnant or breast feeding. Have a history of depression, liver or kidney disease, or respiratory conditions.</td>
</tr>
<tr>
<td>Zolpidem (Ambien, Edluar)</td>
<td>Have a history of depression, liver or kidney disease, or respiratory conditions.</td>
</tr>
<tr>
<td>Estazolam</td>
<td>Are pregnant, breast-feeding, or are an older adult.</td>
</tr>
<tr>
<td>Eszopiclone (Lunesta)</td>
<td>Have a history of drug or alcohol abuse, depression, lung disease, or a condition that affects metabolism.</td>
</tr>
<tr>
<td>Temazepam (Restoril)</td>
<td>Have a history of severe depression, substance abuse, lung disease, or kidney or liver problems. Are pregnant or breast-feeding.</td>
</tr>
<tr>
<td>Zolpidem (Ambien CR)</td>
<td>Have a history of depression, liver or kidney disease, or respiratory conditions. Are pregnant or breast-feeding.</td>
</tr>
</tbody>
</table>
# Simple rules for gender and age
{ ?P : hasSasVariableAGE ?VAR. } ℜ { ?P : hasAge ?VAR. }
{ ?P : hasSasVariableSEX 2. } ℜ { ?P : hasSex : Female. }
{ ?P : hasSasVariableSEX 1. } ℜ { ?P : hasSex : Male. }

# RULE: hasDisease : Insomnia
# DESCRIPTION: A patient has insomnia if
# 1. Patient gets less than 4 hours of sleep in a day
# 2. Patient gets less than 15 hours of sleep in a month

# RULE: hasDisease : SleepApnea
# DESCRIPTION: A patient has sleep apnea if he/she snores
{ ?P a : Patient. ?P : hasSasVariableSLEPSNOR 1. } ℜ { ?P : hasDisease : SleepApnea. }

# RULE: hasDisease : Asthma
# DESCRIPTION: A patient has asthma if hasSasVariableASTHMA2=1
# or hasSasVariableASTHNOW=1
{ ?P a : Patient. ?P : hasSasVariableASTHMA2 1. } ℜ { ?P : hasDisease : Asthma. }
{ ?P a : Patient. ?P : hasSasVariableASTHNOW 1. } ℜ { ?P : hasDisease : Asthma. }

# Asthma is a RespiratoryCondition condition
{ ?P : hasDisease : Asthma. } ℜ { ?P : hasCondition : RespiratoryCondition. }

# RULE: hasDisease : LungDisease
# DESCRIPTION: A patient has lung disease if been told
{ ?P a : Patient. ?P : hasSasVariableDRHPAD1 1. } ℜ { ?P : hasDisease : LungDisease. }

# has lung cancer
{ ?P a : Patient. ?P : hasSasVariableCNCRTYPE 23. } ℜ { ?P : hasDisease : LungDisease. }

# RULE: hasDisease : KidneyDisease
# DESCRIPTION: A patient has Kidney disease if been told
{ ?P a : Patient. ?P : hasSasVariableDRHPAD1 1. } ℜ { ?P : hasDisease : KidneyDisease. }

# has Kidney cancer
{ ?P a : Patient. ?P : hasSasVariableCNCRTYPE 25. } ℜ { ?P : hasDisease : KidneyDisease. }
# RULE: hasDisease : Liver Disease
# DESCRIPTION: A patient has Liver disease if he has liver cancer
{ ?P a : Patient. ?P : hasSasVariableCNCRTYPE 11. }
} => { ?P : hasDisease : Liver Disease }.

# RULE: a : Elderly
# DESCRIPTION: A patient if over 65 is considered to be an elderly person
{ ?P a : Patient. ?P : hasAge ?AGE.
  ?AGE math : greaterThan 65. }
} => { ?P : hasCondition : Elderly }.

# RULE: hasCondition : Pregnancy
# DESCRIPTION: A patient is pregnant if the patient is female and has
# hasSasVariablePREGNANT=1
{ ?P a : Patient. ?P : hasSex : Female. ?P : hasSasVariablePREGNANT 1. }
} => { ?P : hasCondition : Pregnancy }.

# RULE: hasCondition : BreastFeeding
# DESCRIPTION: A patient is breastfeeding if Female and hasSasVariableNOBCUSE=13
{ ?P a : Patient. ?P : hasSex : Female. ?P : hasSasVariableNOBCUSE 13. }
} => { ?P : hasCondition : BreastFeeding }.

# RULE: hasCondition : Alcohol Abuse
# DESCRIPTION: A hasSasVariable_RFDRHV3=2
{ ?P a : Patient. ?P : hasSasVariable_RFDRHV3 2. }
} => { ?P : hasCondition : Alcohol Abuse }.

# RULE: hasDisease : Depression
{ ?P a : Patient.
  ?P : hasSasVariableMENTHLTH ?VAR.
  ?VAR math : lessThan 31. 
  ?VAR math : greaterThan 13. }
} => { ?P : hasDisease : Depression }.

{ ?P a : Patient.
  ?P : hasSasVariableADDOWN ?VAR.
  ?VAR math : lessThan 15. 
  ?VAR math : greaterThan 6. }
} => { ?P : hasDisease : Depression }.

{ ?P a : Patient.
  ?P : hasSasVariableADPLEASR ?VAR.
  ?VAR math : lessThan 15. 
  ?VAR math : greaterThan 6. }
} => { ?P : hasDisease : Depression }.

{ ?P a : Patient.
  ?P : hasSasVariableADFAIL ?VAR.
  ?VAR math : lessThan 15. 
  ?VAR math : greaterThan 6. }
} => { ?P : hasDisease : Depression }.

{ ?P a : Patient.
  ?P : hasSasVariableADANXEV 1. }
\{ a : Patient.
\} \implies \{ ?P : hasSasVariableADDEPEV 1.
\} \implies \{ ?P : hasDisease : Depression \}.  

C Drug Prescription Rules

# RULE: Generic drug interaction rule
# If a patient is taking an existing drug (D1) and D1 has contraindication with another drug D2
# then drug d2 should not be prescribed to the patient
#
{ ?P a:Patient.
 ?D1 a:Drug.
 ?D2 a:Drug.
 ?D1 :hasContraindication ?D2.
} => { ?P :cannotBeGiven ?D2 }.

{ ?P a:Patient.
 ?D1 a:Drug.
 ?D2 a:Drug.
} => { ?P :cannotBeGiven ?D2 }.

# RULE: Generic drug interaction rule
# If a patient has a condition that has a contraindication
# with a drug, then the patient should not be given the drug.
#
{ ?P a:Patient.
 ?D a:Drug.
 ?P :hasCondition ?MAL.
 ?D :hasContraindication ?MAL.
} => { ?P :cannotBeGiven ?D }.

# RULE: cannotBeGiven
# If a patient has a disease and the drug has a contraindication
# with the disease, then the drug can not be given to patient.
#
{ ?P a:Patient.
 ?D a:Drug.
 ?P :hasDisease ?DIS.
 ?D :hasContraindication ?DIS.
} => { ?P :cannotBeGiven ?D }.