WatsonPaths: Scenario-Based Question Answering and Inference over Unstructured Information

Adam Lally, Sugato Bagchi, Michael A. Barborak, David W. Buchanan, Jennifer Chu-Carroll, David A. Ferrucci, Michael R. Glass, Aditya Kalyanpur, Erik T. Mueller, J. William Murdock, Siddharth Patwardhan, John M. Prager

■ *We present WatsonPaths, a novel system that can answer scenario-based questions. These include medical questions that present a patient summary and ask for the most likely diagnosis or most appropriate treatment. WatsonPaths builds on the IBM Watson question-answering system. WatsonPaths breaks down the input scenario into individual pieces of information, asks relevant subquestions of Watson to conclude new information, and represents these results in a graphic model. Probabilistic inference is performed over the graph to conclude the answer. On a set of medical test preparation questions, WatsonPaths shows a significant improvement in accuracy over multiple baselines.*

IBM Watson is a question-answering system that takes natural language questions as input and produces precise answers along with accurate confidences as output (Ferrucci et al. 2010). In 2011, in a modified version of the quiz show *Jeopardy!,* Watson defeated two of the best human players.

Jeopardy! questions are usually factoid questions — the answer and supporting evidence are usually stated explicitly in some document in the corpus. While in practice we may retrieve multiple redundant documents, in principle the answer could be expressed succinctly in one. The main challenges for a factoid question-answering system are retrieving the correct document, and then extracting the correct answer from the document. At the core of Watson's question answering is a suite of algorithms that match passages containing candidate answers to the original question. These algorithms have been described in a series of articles (Chu-Carroll et al. 2012; Ferrucci 2012; Gondek et al. 2012; Lally et al. 2012; McCord, Murdock, and Boguraev 2012; Murdock et al. 2012a; 2012b). In some important applications, however, questions do not have this "factoid" character. Consider the

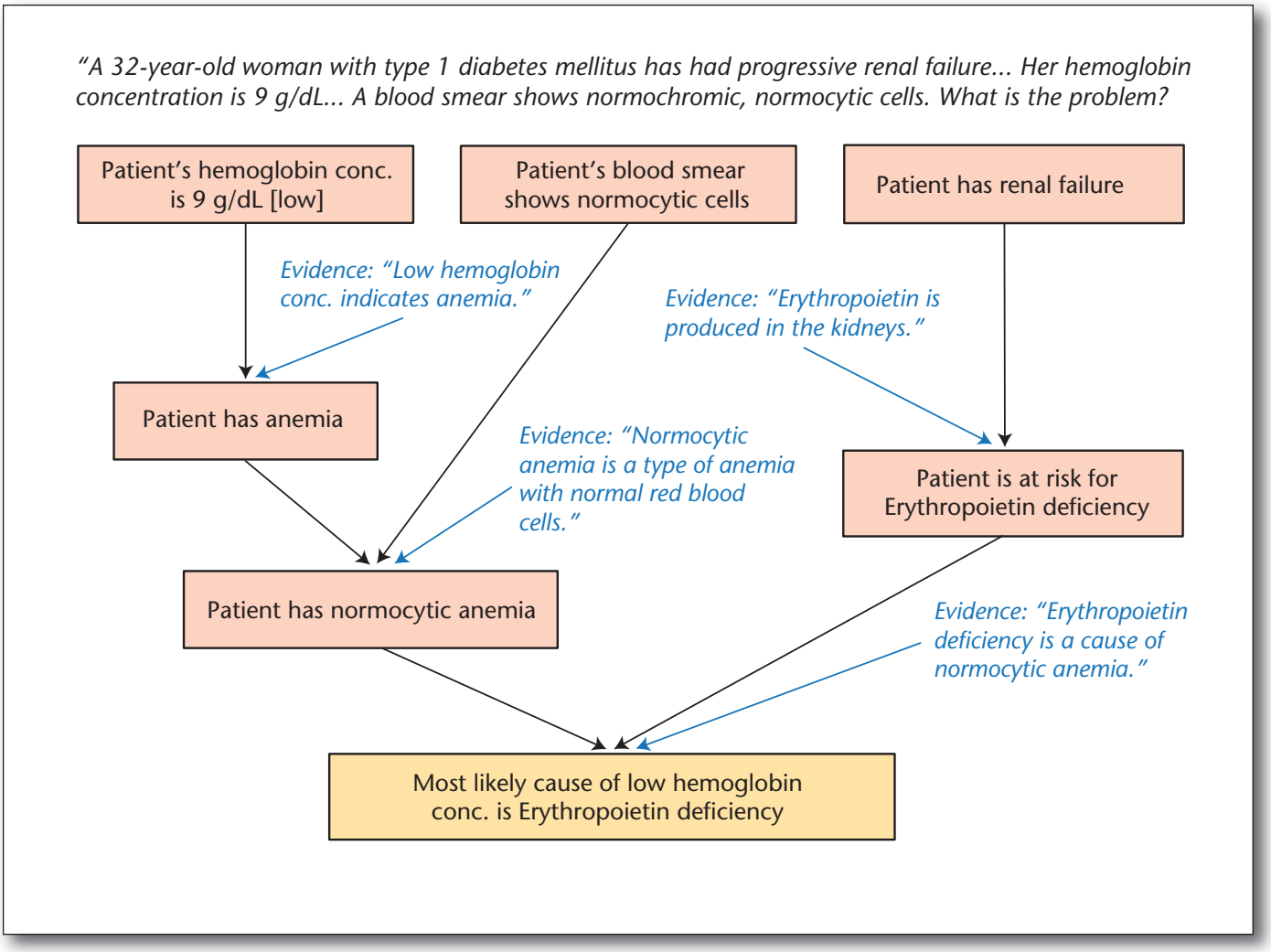

Figure 1. A Simple Diagnosis Graph for a Patient with Erythropoietin Deficiency.

following questions, one from medicine and one from taxation:

> A 32-year-old woman with type 1 diabetes mellitus has had progressive renal failure. Her hemoglobin concentration is 9 g/dL. A blood smear shows normochromic, normocytic cells. What is the problem?

> I inherited real-estate from a relative who died 5 years ago via a trust that was created before his death. The property was sold this year after dissolution of the trust, and the money was put in a Roth-IRA. Which tax form(s) do I need to file?

We will call these types of questions *scenario-based questions*. In these types of questions, it is not generally the case that the answer and supporting evidence can be contained in one document. Rather, for many scenario-based questions, information from multiple documents and other sources must generally be retrieved and then integrated to answer the questions properly. Furthermore, we must often apply general knowledge to a specific case, as in a medical scenario about a patient.

Before beginning work on automated scenario-based question answering, we investigated how humans solve such questions. We asked domain experts to describe their approach to solving a set of scenario-based questions in the medical domain. An example is shown in figure 1. Many drew a graph of initial signs and symptoms leading to their most likely possible causes and connecting them to a final conclusion. This motivated us to look into graph-based methods as a way of answering scenario-based questions automatically.

In this article, we describe WatsonPaths, a system that builds on Watson to answer scenario-based questions. The core idea is to break the question down into parts, over which we can ask and answer factoid subquestions using Watson, then integrate these answers into a graphic model that can be used to

answer the larger scenario-based question. We show that WatsonPaths not only outperforms a baseline system that uses simple information retrieval, but also outperforms its own subcomponent, Watson, in answering a set of scenario-based questions from the medical domain.

WatsonPaths Medical Use Case

Although WatsonPaths is intended as a domain-general technology for scenario-based question answering, we decided to start by focusing our attention on the medical domain. We focused on the problem of patient scenario analysis, where the goal is typically a diagnosis or a treatment recommendation.

To explore this kind of problem solving, we obtained a set of medical test preparation questions. These are multiple-choice medical questions based on an unstructured or semistructured natural language description of a patient. Although WatsonPaths is not restricted to multiple-choice questions, we saw multiple-choice questions as a good starting point for development. Many of these questions involve diagnosis, either as the entire question, as in the previous medical example, or as an intermediate step, as in the following example:

> A 63-year old patient is sent to the neurologist with a clinical picture of resting tremor that began 2 years ago. At first it was only on the left hand, but now it compromises the whole arm. At physical exam, the patient has an unexpressive face and difficulty in walking, and a continuous movement of the tip of the first digit over the tip of the second digit of the left hand is seen at rest. What part of his nervous system is most likely affected?

For this question, it is useful to diagnose that the patient has Parkinson's disease before determining which part of his nervous system is most likely affected. These multistep inferences are a natural fit for the graphs that WatsonPaths constructs. In this example, the diagnosis is the missing link on the way to the final answer.

Scenario-Based Question Answering

In scenario-based question answering, the system receives a scenario description that ends with a punch line question. For instance, the punch line question in the Parkinson's example is "What part of his nervous system is most likely affected?" Instead of treating the entire scenario as one monolithic question as would Watson, WatsonPaths explores multiple facts in the scenario in parallel and reasons with the results of its exploration as a whole to arrive at the most likely conclusion regarding the punch line question. The architecture of WatsonPaths is shown in figure 2. In this section, we briefly outline each step, while the bulk of the rest of the article goes into more detail on important steps.

Scenario Analysis

The first step in the pipeline is scenario analysis, where we identify factors in the input scenario that may be of importance. In the medical domain, the factors may include demographics ("32-year old woman"), preexisting conditions ("type 1 diabetes mellitus"), signs and symptoms ("progressive renal failure"), and test results ("hemoglobin concentration is 9 g/dL," "normochromic cells," "normocytic cells"). The extracted factors become nodes in a graph structure called the assertion graph, on which the remaining steps of the process will operate.

Node Prioritization

The next step is node prioritization, where we decide which nodes in the graph are most important for solving the problem. In a small scenario like this example, we may be able to explore everything, but in general this will not be the case. Factors that affect the priority of a node may include the system's confidence in the node assertion or the system's estimation of how fruitful it would be to expand a node. For example, normal test results and demographic information are generally less useful for starting a diagnosis than symptoms and abnormal test results.

Relation Generation

The relation-generation step builds the assertion graph. We do this primarily by asking Watson questions about the factors. In medicine we want to know the causes of the findings and abnormal test results that are consistent with the patient's demographic information and normal test results. Given the scenario in the Introduction, we could ask, "What does type 1 diabetes mellitus cause?" We use a medical ontology to guide the process of formulating subquestions to ask Watson. Relevant factors may also be combined to form a single, more targeted question. Because in this step we want to emphasize recall, we take several of Watson's highly ranked answers. The exact number of answers taken, or the confidence threshold, are parameters that must be tuned. Given a set of answers, we add them to the graph as nodes, with edges from nodes that were used in questions to nodes that were answers. The edge is labeled with the predicate used to formulate the question (like *causes* or *indicates*), and the strength of the edge is initially set to Watson's confidence in the answer.

Although Watson is the primary way we add edges to the graph, WatsonPaths allows for any number of relation generator components to post edges to the graph. For instance, we apply term matchers to pairs of nodes, and post a relation between nodes that match.

Belief Computation

Once the assertion graph has been expanded in this way, we recompute the confidences of nodes in the graph based on new information. We do this using

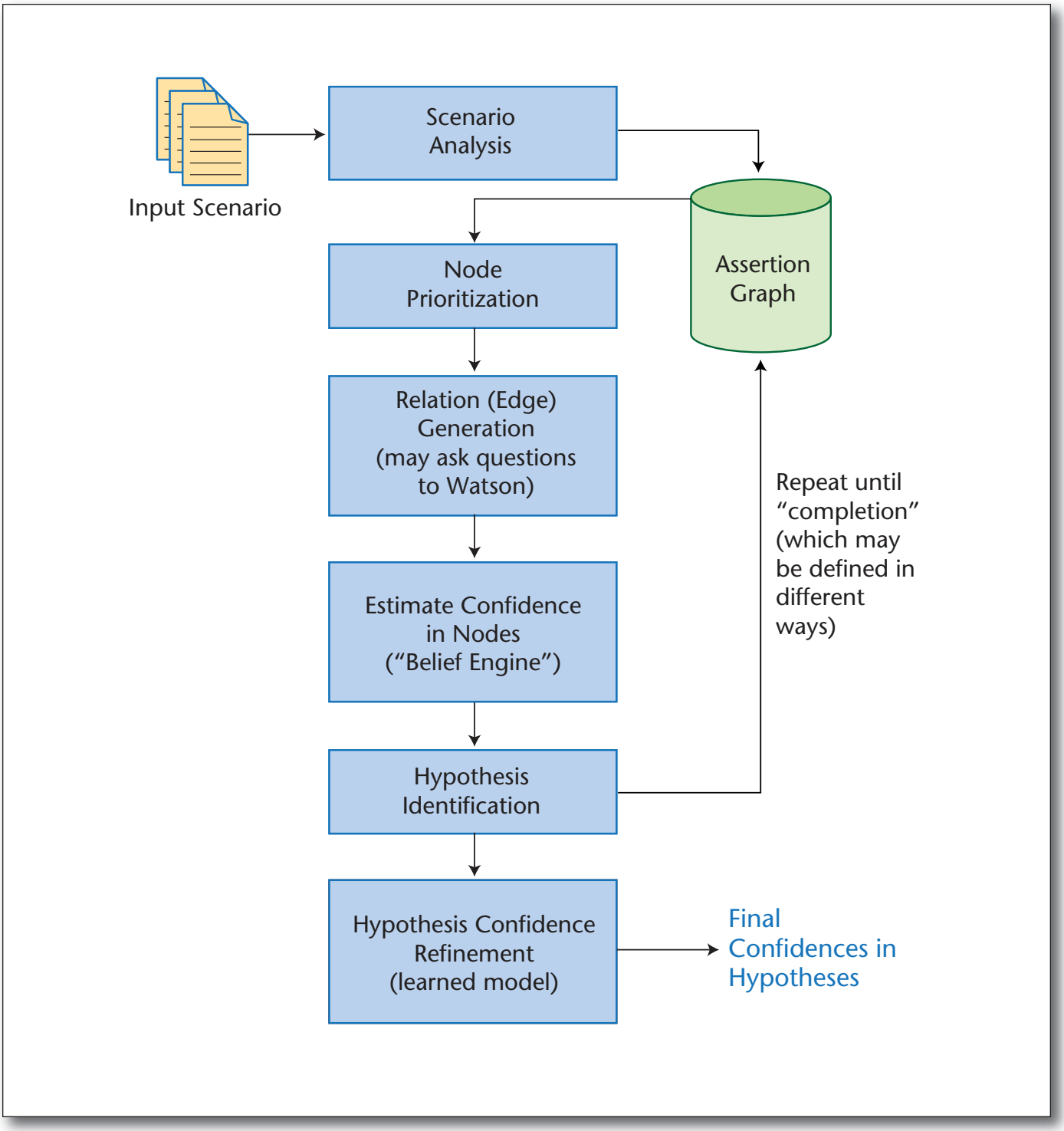

Figure 2. Scenario-Based Question-Answering Architecture.

probabilistic inference systems that take a holistic view of the assertion graph and try to reconcile the results of multiple paths of exploration.

Hypothesis Identification

As figure 2 shows, the process can go through multiple iterations, during which the nodes that were the answers to the previous round of questions can be used to ask the next round of questions, producing more nodes and edges in the graph.

After each iteration we may do hypothesis identification, where some nodes in the graph are identi-

fied as potential final answers to the punch line question (for example, the most likely diagnoses of a patient's problem). In some situations hypotheses may be provided up front — a physician may have a list of competing diagnoses and want to explore the evidence for each. But in general the system needs to identify these. Hypothesis nodes may be treated differently in later iterations. For instance, we may attempt to do backward chaining from the hypotheses, asking Watson what things, if they were true of the patient, would support or refute a hypothesis. The process may terminate after a fixed number of iterations or based on some other criterion like confidence in the hypotheses.

While hypothesis identification is part of WatsonPaths, it is not described in detail in this article. In the system that generates the results we present in this article, no hypothesis identification is necessary because the multiple-choice answers are provided. That system always does one iteration of expansion, both forward from the identified factors and backward from the hypotheses, before stopping.

Hypothesis Confidence Refinement

As described so far, WatsonPath's confidence in each hypothesis depends on the strengths of the edges leading to it, and since our primary relation (edge) generator is Watson, the hypothesis confidence depends heavily on the confidence of Watson's answers. Having good answer confidence depends on having a representative set of question/answer pairs with which to train Watson. The following question arises: What can we do if we do not have a representative set of question/answer pairs, but we do have training examples for entire scenarios (for example, correct diagnoses associated with patient scenarios)? To leverage the available scenario-level ground truth, we have built machine-learning techniques to learn a refinement of Watson's confidence estimation that produces better results when applied to the entire scenario. We describe our techniques in the Learning over Assertion Graphs section.

Assertion Graphs

The core data structure used by WatsonPaths is the assertion graph. Figure 3 explains this data structure, along with the visualization that we commonly use for it. Assertion graphs are defined as follows.

A *statement* is something that can be true or false (though its state may not be known). Often we deal with unstructured statements, which are natural language expressions like "A 63-year-old patient is sent to the neurologist with a clinical picture of resting tremor that began 2 years ago." WatsonPaths also allows for statements that are structured expressions, namely, a predicate and arguments. Not all natural language expressions can have a truth value. For instance, the string "patient" cannot be true or false; thus it does not fit into the semantics of an assertion graph. WatsonPaths is charitable in interpreting strings as if they had a truth value. For instance, the default semantics of the string "low hemoglobin" is the same as "patient has low hemoglobin."

A *relation* is a named association between statements. Technically, relations are themselves statements, and have a truth value. The relation has a head, a tail, and predicate; for instance in medicine we may say that "Parkinsons causes resting tremor" or "Parkinson's matches Parkinsonism." Typically we are concerned with relations that may provide evidence for the truth of one statement given another. Although some relations may have special meanings in the probabilistic inference systems, a common semantics for a relation is indicative in the following way: *"A* indicates *B"* means that the truth of *A* provides an independent reason to believe that *B* is true.

An *assertion* is a claim that some agent makes about the truth of a statement (including a relation). The assertion records the name of the agent and a confidence value. Assertions may also record provenance information that explains how the agent came to its conclusion. For the Watson question-answering agent, this includes natural language passages that provide evidence for the answer.

In the assertion graph, each node represents exactly one statement, and each edge represents exactly one relation. Nodes and edges may have multiple assertions attached to them, one for each agent that has asserted that node or edge to be true.

We often visualize assertion graphs by using a node's border width to represent the confidence of the node, an edge's width to represent the confidence of the edge, and an edge's gray level as the amount of "belief flow" along that edge. Belief flow is described later, but essentially it is how much the value of the head influences the value of the tail. This depends mostly on the confidences of the assertions on the edge.

Scenario Analysis

The goal of scenario analysis is to identify information in the natural language narrative of the problem scenario that is potentially relevant to solving the problem. When human experts read the problem narrative, they are trained to extract concepts that match a set of semantic types relevant for solving the problem. In the medical domain, doctors and nurses identify semantic types like chief complaints, past medical history, demographics, family and social history, physical examination findings, labs, and current medications (Bowen 2006). Experts also generalize from specific observations in a particular problem instance to more general terms used in the domain corpus. An important aspect of this information extraction is to identify the semantic qualifiers associated with the clinical observations (Chang, Bor-

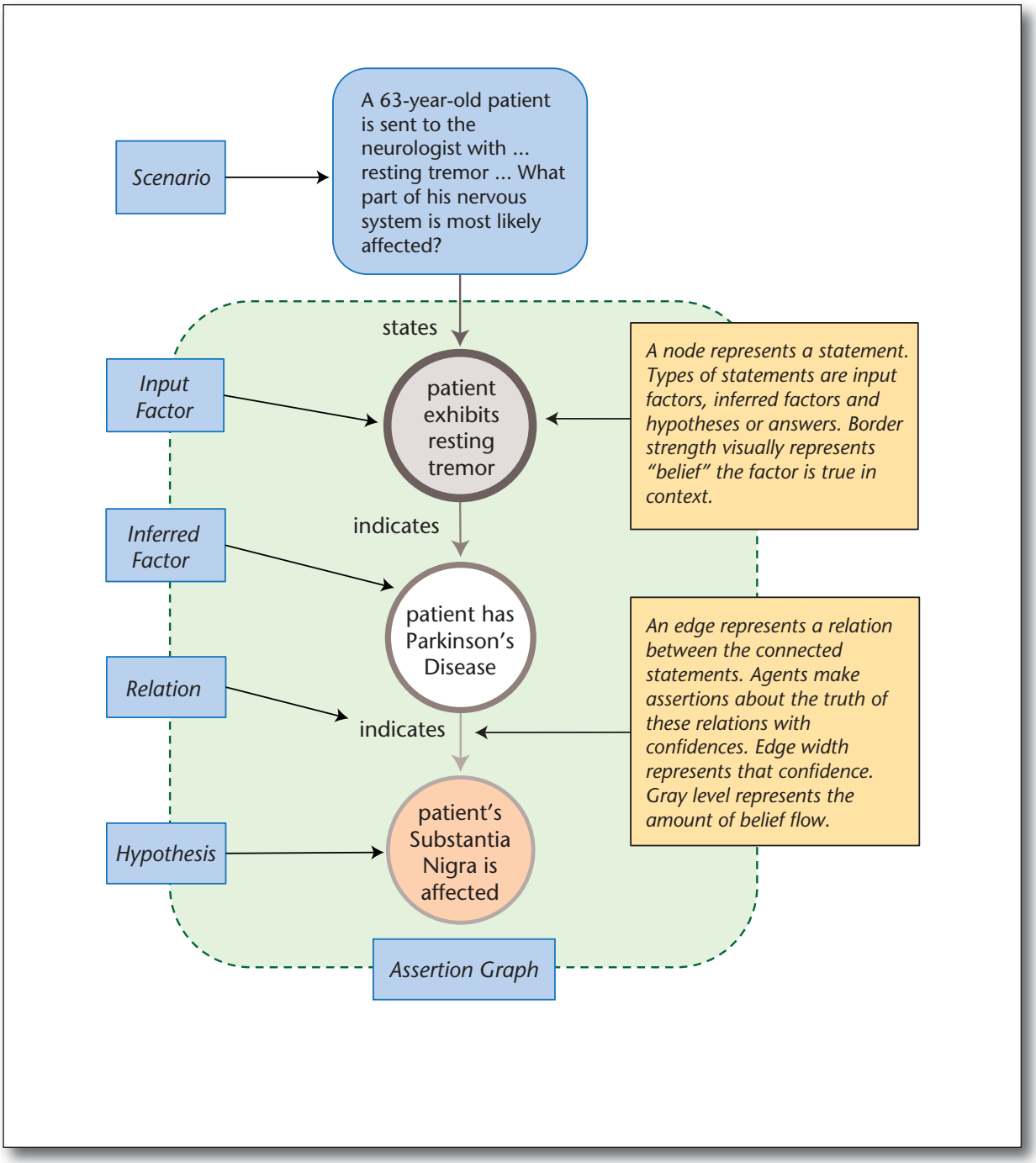

Figure 3. Visualization of an Assertion Graph.

By convention, input factors are placed at the top and hypotheses at the bottom with levels of inference factors in between.

dage, and Connell 1998). These qualifiers could be temporal (for example, "pain started two days ago"), spatial ("pain in the epigastric region"), or other associations ("pain after eating fatty foods"). Implicit in this task is the human's ability to extract concepts and their associated qualifiers from the natural language narrative. For example, the above qualifiers might have to be extracted from the sentence "The patient reports pain, which started two days ago, in the epigastric region especially after eating fatty foods."

The computer system needs to perform a similar analysis of the narrative. We use the term *factor* to denote the potentially relevant observations along

with their associated semantic qualifiers. Reliably identifying and typing these factors, however, is a difficult task, because medical terms are far more complex than the kind of named entities typically studied in natural language processing. Our scenario analytics pipeline attempts to address this problem with the following major processing steps:

Step one: The analysis starts with syntactic parsing of the natural language. This creates a dependency tree of syntactically linked terms in a sentence and helps to associate terms that are distant from each other in the sentence.

Step two: The terms are mapped to a dictionary to identify concepts and their semantic types. For the medical domain, our dictionary is derived from the *Unified Medical Language System (UMLS) Metathesaurus* (National Library of Medicine 2009), Wikipedia redirects, and medical abbreviation resources. The concepts identified by the dictionary are then typed using the UMLS Semantic Network, which consists of a taxonomy of biological and clinical semantic types like Anatomy, SignOrSymptom, DiseaseOrSyndrome, and TherapeuticOrPreventativeProcedure. In addition to mapping the sequence of tokens in a sentence to the dictionary, the dependency parse is also used to map syntactically linked terms. For example "… stiffness and swelling in the arm and leg" can be mapped to the four separate concepts contained in that phrase.

Step three: The syntactic and semantic information identified above is used by a set of predefined rules to identify important relations. Negation is commonly used in clinical narratives and needs to be accurately identified. Rules based on parse features identify the negation trigger term and its scope in a sentence. Factors found within the negated scope can then be associated with a negated qualifier. Another example of rule-based annotation is lab value analysis. This associates a quantitative measurement to the substance measured and then looks up reference lab value ranges to make a clinical assessment. For example "hemoglobin concentration is 9 g/dL" is processed by rules to extract the value, unit, and substance and then assessed to be "low hemoglobin" by looking up a reference. Next, the clinical assessment is mapped by the dictionary to the corresponding clinical concept.

At this point, we should have all the information to identify factors and their semantic qualifiers. We have to contend, however, with language ambiguities, errors in parsing, a noisy and noncomprehensive dictionary, and a limited set of rules. If we were to rely solely on a rule-based system, then the resulting factor identification would suffer from a compounding of errors in these components. To address this issue, we employ machine-learning methods to learn clinical factors and their semantic qualifiers in the problem narrative. We obtained the ground truth by asking medical students to annotate clinical factor spans and their semantic types. They also annotated semantic qualifier spans and linked them to factors as attributive relations.

The machine-learning system comprises two sequential steps:

In step one, a conditional random field (CRF) model (Lafferty, McCallum, and Pereira 2001) learns the spans of text that should be marked as one of the following factor types: finding, disease, test, treatment, demographics, negation, or a semantic qualifier. Features used for training the CRF model are lexical (lemmas, morphological information, part-of-speech tags), semantic (UMLS semantic types and groups, demographic and lab value annotations), and parse-based (features associated with dependency links from a given token). A token window size of five (two tokens before and after) is used to associate features for a given token. A BIO tagging scheme is used by the CRF to identify entities in terms of their token spans and types.

In step two, a maximum entropy model then learns the relations between the entities identified by the CRF model. For each pair of entities in a sentence, this model uses lexical features (within and between entities), entity type, and other semantic features associated with both entities, and parse features in the dependency path linking them. The relations learned by this model are *negation* and *attributeOf* relations linking negation triggers and semantic qualifiers (respectively) to factors.

The combined entity and relation identification models have a precision of 71 percent and recall of 65 percent on a blind evaluation set of patient scenarios found in medical test preparation questions. We are currently exploring joint inference models and identification of relations that span multiple sentences using coreference resolution.

Relation Generation

The scenario analysis component described in the previous section extracts pertinent factors related to the patient from the scenario description. At this stage, the assertion graph consists of the full scenario, individual scenario sentences, and the extracted factors. An *indicates* relation is posted from a source node (for example, a scenario sentence node) to a target node whose assertion was derived from the assertion in the source node (for example, a factor extracted from that sentence). In addition, a set of hypotheses, if given, is posted as the goal nodes in the assertion graph.

The task of the relation generation component is to (1) expand the graph by inferring new facts from known facts in the graph and (2) identify relationships between nodes in the graph (like *matches* and *contraindicates*) to help with reasoning and confidence estimation. We begin by discussing how we infer new facts for graph expansion.

Expanding the Graph with Watson

In medical problem solving, experts reason with chief complaints, findings, medical history, demographic information, and so on, to identify the underlying causes for the patient's problems. Depending on the situation, they may then proceed to propose a test whose results will allow them to distinguish between multiple possible problem causes, or identify the best treatment for the identified cause, and so on.

Motivated by the medical problem-solving paradigm, WatsonPaths first attempts to make a diagnosis based on factors extracted from the scenario. The graph is expanded to include new assertions about the patient by asking questions of a version of the Watson question-answering system adapted for the medical domain (Ferrucci et al. 2013). WatsonPaths takes a two-pronged approach to medical problem solving by expanding the graph forward from the scenario in an attempt to make a diagnosis, and then linking high-confidence diagnoses with the hypotheses. The latter step is typically done by identifying an important relation expressed in the punch line question (for example, "What is the most appropriate treatment for this patient" or "What body part is most likely affected?"). This approach is a logical extension of the open-domain work of Prager, Chu-Carroll, and Czuba (2004), where in order to build a profile of an entity, questions were asked of properties of the entity and constraints between the answers were enforced to establish internal consistency.

The graph expansion process of WatsonPaths begins with automatically formulating questions related to high-confidence assertions, which in our graphs represent statements WatsonPaths believes to be true to a certain degree of confidence about the patient. These statements may be factors, as extracted and typed by our scenario analysis algorithm, or combinations of those factors.

To determine what kinds of questions to ask, WatsonPaths can use a domain model that tells us what relations form paths between the semantic type of a high-confidence node and the semantic type of a hypothesis like a diagnosis or treatment. For the medical domain, we created a model that we called the Emerald, which is shown in figure 4. (Notice the resemblence to an emerald.) The Emerald is a small model of entity types and relations that are crucial for diagnosis and for formulating next steps.

We select from the Emerald all relations that link the semantic type of a high-confidence source node to a semantic type of interest. The relations and the high-confidence nodes then form the basis of instantiating the target nodes, thereby expanding the assertion graph. To instantiate the target nodes, we issue WatsonPaths subquestions to Watson. All answers returned by Watson that score above a predetermined threshold are posted as target nodes in the inference graph. A relation edge is posted from the source node to each new target node where the confidence of the relation is Watson's confidence in the answer in the target node.

In addition to asking questions from scenario factors, WatsonPaths may also expand backwards from hypotheses. The premise for this approach is to explore how a hypothesis fits in with the rest of the inference graph. If one hypothesis is found to have a strong relationship with an existing node in the assertion graph, then our probabilistic inference mechanisms allow belief to flow from known factors to that hypothesis, thus increasing the system's confidence in that hypothesis.

Figure 5 illustrates the WatsonPaths graph expansion process. The top two rows of nodes and the edges between them show a subset of the WatsonPaths assertion graph after scenario analysis, with the second row of nodes representing some clinical factors extracted from the scenario sentences.

The graph expansion process identifies the most confident assertions in the graph, which include the four clinical factor nodes extracted from the scenario. These four nodes are all typed as findings, so they are aggregated into a single *finding* node for the purpose of graph expansion. For a *finding* node, the Emerald proposes a single *findingOf* relation that links it to a disease. This results in the formulation of the subquestion "What disease causes resting tremor that began 2 years ago, compromises the whole arm, unexpressive face, and difficulty in walking?" whose answers include Parkinson's disease, Huntington's disease, cerebellar disease, and so on. These answer nodes are added to the graph and some of them are shown in the third row of nodes in figure 5.

In the reverse direction, WatsonPaths explores relationships between hypotheses to nodes in the existing graph based on the punch line question in the scenario, which in this case is "What part of his nervous system is mostly likely affected?" Assuming each hypothesis to be true, the system formulates subquestions to link it to the assertion graph. Consider the hypothesis, substantia nigra. WatsonPaths can ask "In what disease is *substantia nigra* most likely affected?" A subset of the answers to this question, including Parkinson's disease and diffuse Lewy body disease are shown in the fourth row of nodes in figure 5.

Matching Graph Nodes

When a new node is added to the WatsonPaths assertion graph, we compare the assertion in the new node to those in existing nodes to ensure that equivalence relations between nodes are properly identified. This is done by comparing the statements in those assertions: for unstructured statements, whether the statements are lexically equivalent, and for structured statements, whether the predicates and their arguments are the same. A more complex operation is to identify when nodes contain assertions that may be equivalent to the new assertion.

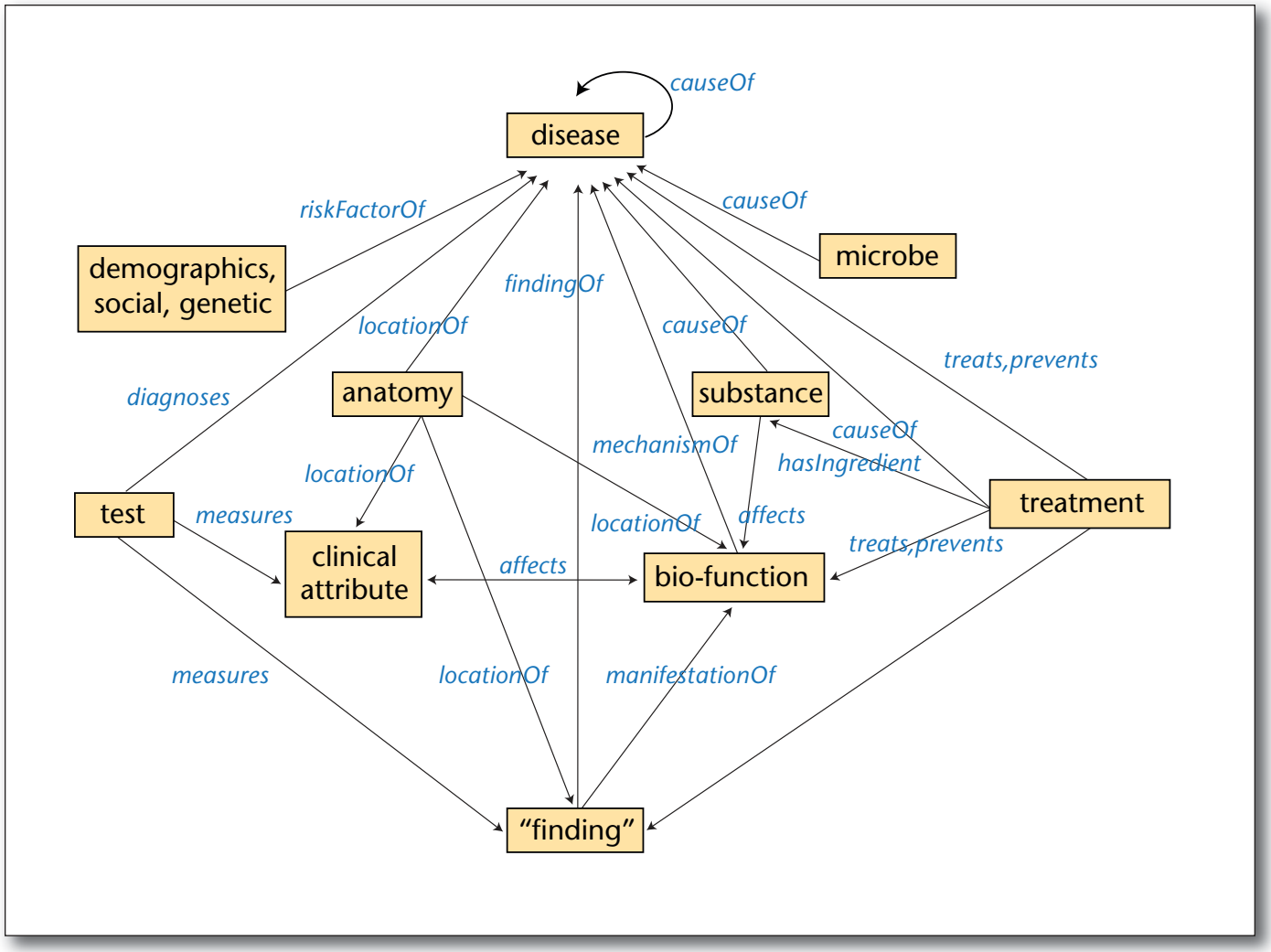

Figure 4. The Emerald.

We employ an aggregate of term matchers (Murdock et al. 2012a) to match pairs of assertions. Each term matcher posts a confidence value on the degree of match between two assertions based on its own resource for determining equivalence. For example, a WordNet-based term matcher considers terms in the same synset to be equivalent, and a Wikipedia-redirect-based term matcher considers terms with a redirect link between them in Wikipedia to be a match. The dotted line between *Parkinson disease* and *Parkinson's disease* in figure 5 is posted by the UMLS-based term matcher, which considers variants for the same concept to be equivalent.

Confidence and Belief

Once the assertion graph is constructed, and some questions and answers are posted, there remains the problem of confidence estimation. We develop multiple models of inference to address this step.

Belief Engine

One approach to the problem of inferring the correct hypothesis from the assertion graph is probabilistic inference over a graphic model (Pearl 1988). We refer to the component that does this as the belief engine.

Although the primary goal of the belief engine is to infer confidences in hypotheses, it also has the secondary goal to infer belief in unknown nodes that are not hypotheses. These intermediate nodes may be important intermediate steps toward an answer; by assigning high confidences to them in the main loop, we know to assign them high priority for subquestion asking. Therefore, the belief engine needs to assign a confidence to each node, not just hypotheses.

To execute the belief engine, we first make a working copy of the assertion graph that we call the inference graph. A separate graph is used so that we can make changes without losing information that might

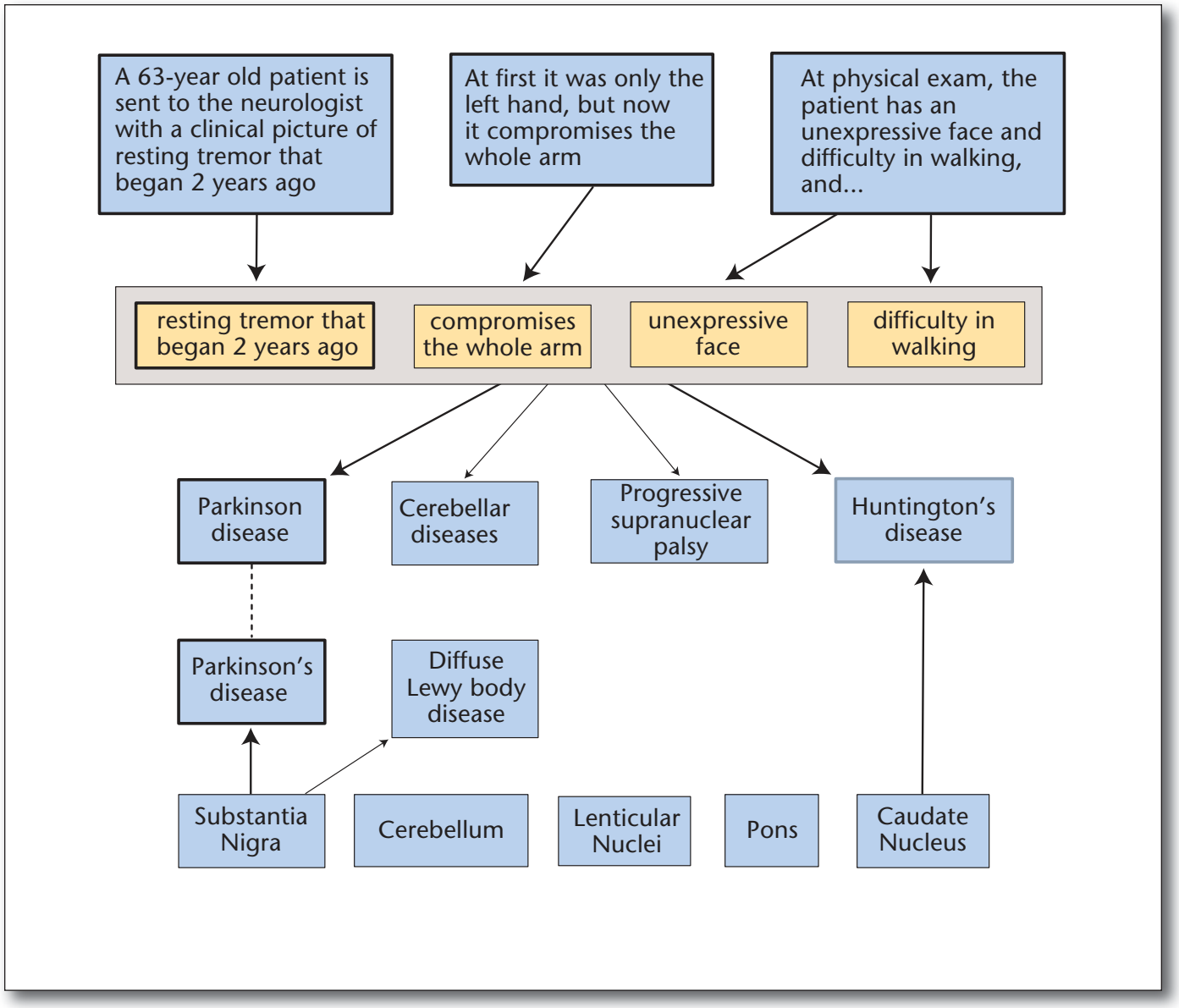

Figure 5. WatsonPaths Graph Expansion Process.

be useful in later steps of inference. For instance, we might choose to merge nodes or reorient edges. Once the inference graph has been built, we run a probabilistic inference engine over the graph to generate new confidences. Each node represents an assertion, so it can be in one of two states: true or false ("on" or "off"). Thus a graph with k nodes can be in 2^k possible states. The inference graph specifies the likelihoods of each of these states. The belief engine uses these likelihoods to calculate the marginal probability, for each node, of it being in the true state. This marginal probability is treated as a confidence. Finally, we read confidences and other data from the inference graph back into the assertion graph.

There are some challenges in applying probabilistic inference to an assertion graph. Most tools in the inference literature were designed to solve a different problem, which we will call the classical inference problem. In this problem, we are given a training set and a test set that can be seen as samples from a common joint distribution. The task is to construct a model that captures the training set (for instance, by maximizing the likelihood of the training set), and then apply the model to predict unknown values in the test set. Arguably the greatest problem in the classical inference task is that the structure of the graphic model is underdetermined; a large space of possible structures needs to be explored. Once a structure

is found, adjusting the strengths is relatively easier, because we know that samples from the training set are sampled from a consistent joint distribution.

In WatsonPaths, we face a different set of problems. The challenge is not to construct a model from training data, but to use a very noisy, already constructed model to do inference. Training data in the classical sense is absent or very sparse; all we have are correct answers to some scenario-level questions. An advantage is that a graph structure is given. A disadvantage is that the graph is noisy. Furthermore, it is not known that the confidences on the edges necessarily correspond to the optimal edge strengths. (In the next section, we address the problem of learning edge strengths.) Thus we have the problem of selecting a semantics — a way to convert the assertion graph into a graph over which we can do optimal probabilistic inference to meet our goals.

After much experimentation, the primary semantics used by the belief engine is the indicative semantics: If there is a directed relation from node A to node B with strength x, then A provides an independent reason to believe that B is true with probability x. Some edges are classified as contraindicative; for these edges, A provides an independent reason to believe that B is false with probability x. The independence means that multiple parents R can easily be combined using a noisy-OR function. For instance, if the node *resting tremor* indicates Parkinson disease with strength 0.8, and the node *difficulty in walking* indicates Parkinson disease with power 0.4, then the probability of Parkinson disease will be $(1 - (1 - 0.8)(1 - 0.4)) = 0.88$. If so, then the edge with strength 0.9 to Parkinson's disease will fire with probability $0.88 \cdot 0.9 = 0.792$. In this way, probabilities can often multiply down simple chains. Inference must be more sophisticated to handle the graphs we see in practice, but the intuition is the same.

As an example of a problem that requires additional sophistication, the assertion graphs in WatsonPaths contain many directed and undirected cycles, which are not allowed in many inference algorithms. We have developed a novel inference algorithm to deal with the directed cycles, which shows an improvement over existing methods. Another example is an "exactly one" constraint that can be optionally added to multiple-choice questions. This constraint assigns a higher likelihood to assignments in which exactly one multiple-choice answer is true. Because of these kinds of constraints, we cannot simply calculate the probabilities in a feed-forward manner. To perform inference we use Metropolis-Hastings sampling over a factor graph representation of the inference graph. This has the advantage of being a very general approach — the inference engine can easily be adapted to a new semantics — and also allows an arbitrary level of precision given enough processing time.

Formally, the indicative semantics instantiates a version of a noisy-logical Bayesian network (Yuille and Lu 2007). The strength of each edge can be interpreted as an indicative power, a concept related to causal power (Cheng 1997), with the difference that we are semantically agnostic as to the true direction of the causal relation.

In experiments, the indicative semantics performs at least as well as other semantics. We outline a few simple alternatives here. One of the first semantics we tried was undirected pairwise Markov random fields. In this semantics, we take the maximum confidence connecting two nodes to be their compatibility: how likely they are to take the same truth value. This performed poorly in practice. We hypothesize that this is because important information is contained in the direction of the edges that Watson returns: asking about A and getting B as an answer is different from asking about B and getting A as an answer. An undirected model loses this information.

The indicative semantics is a default, basic semantics. The ability to reason over arbitrary relations makes the indicative semantics robust, but it is easy to construct examples in which the indicative semantics is not strictly correct. For instance, "fever is a finding of Lyme disease" may be correctly true with high confidence, but this does not mean that fever provides an independent reason to believe that Lyme disease is present, with high probability. Fever is caused by many things, each of which could explain it. We are currently working on adding a causal semantics in which we use a noisy-logical Bayesian network, but belief flows from causes to effects, rather than from factors to hypotheses. Edges are oriented according to the types of the nodes: Diseases cause findings but not vice versa. Currently this does not lead to detectable improvement in accuracy. We expect that we need to improve the precision of the rest of the system before it will show impact.

Learning over Assertion Graphs

The belief engine described in the previous section uses confidences directly as strengths. These confidences include the output of multiple relation generators (for instance, matching algorithms) but most importantly the confidences of Watson's subquestion answering. For instance, if we ask a question about resting tremor, and get Parkinson's disease as an answer with 73 per cent confidence, then the belief engine will treat this as a directed relation from resting tremor to parkinson's disease with an indicative power of 0.73. When multiple relations exist in one direction, the belief engine takes the maximum of all the confidences in that direction. Watson combines subquestion features using a model that was trained on medical trivia questions (see the evaluation section for details). This raises the question of whether we can also use the training set of scenario-based

questions to better learn how to map features more optimally to edge strengths. This section describes our attempts to do so.

There are two main ways that we use the scenario-based training set. First, we create a set of closed-form inference methods, most of which are approximations of methods described in the previous section, but are more straightforward to optimize. We use these methods to create equations that express candidate confidences in terms of the model, and then optimize the model to maximize certain objectives such as accuracy. Second, we combine all the inference methods, including the belief engine, into an ensemble and use the scenario-based train set to optimize the weights of the ensemble.

Closed-Form Inference Methods

This section describes a series of closed-form inference methods. Note that none of these methods by themselves have been shown to give higher accuracy than the belief engine method described earlier. However, they are more amenable to optimization, and the ensemble of methods may perform better.

The *noisy-OR* model is most similar to the indicative semantics used in the belief engine, with some differences. While the belief engine allows a graph with directed cycles, the noisy-OR model requires a directed acyclic graph (DAG). As mentioned above, the assertion graph is not, in general, free of cycles. Additionally, the assertion graph contains matching relations, which are undirected. To form a DAG, the nodes in the assertion graph are first clustered by these matching relations, and then cycles are broken by applying heuristics to reorient edges to point from factors to hypotheses. Confidence is computed in a feed-forward manner. The confidence in factors extracted by scenario analysis is 1.0. For all other nodes the confidence is defined recursively in terms of the confidences of the parents and the confidence of the edges produced by the question-answering system. This generates an equation for each candidate, expressing its confidence interms of the parameters.

The *edge type* variant of the noisy-OR model considers the type of the edge when propagating confidence from parents to children. The strength of the edge according to the question-answering model is multiplied by a per-edge-type learned weight, then a sigmoid function is applied. In this model, different types of subquestions may have different influence on confidences, even when the question-answering model produces similar features for them.

The *matching model* estimates the confidence in a hypothesis according to how well each factor in the scenario, plus the answers to forward questions asked about it, match against either the hypothesis or the answers to the backward questions asked from it. We estimate this degree of match using the term matchers described earlier in the Matching Graph Nodes section.

The *feature addition model* uses the same DAG as the noisy-OR model, but confidence in the intermediate nodes is computed by adding the feature values for the questions that lead to it and then applying the logistic model to the resulting vector. An effect is that the confidence for a node does not increase monotonically with the number of parents. Instead, if features that are negatively associated with correctness are present in one edge, it can lower the confidence of the node below the confidence given by another edge.

The *causal model* attempts to capture causal semantics by expressing the confidence for each candidate as the product over every clinical factor of the probability that either the diagnosis could explain the factor (as estimated from Watson/question-answering features), or the factor "leaked" — it is an unexplained observation or is not actually relevant.

In the closed-form inference systems described, there is no constraint that the answer confidences sum to one. We implement a final stage where features based on the raw confidence from the inference model are transformed into a proper probability distribution over the candidate answers.

Direct Learning

The methods described in the previous section permit expressing the confidence in the correct answer as a closed-form expression. Summing the log of the confidence in the correct hypothesis across the training set *T,* we construct a learning problem with log-likelihood in the correct final answer as our objective function. The result is a function that is nonconvex, and in some cases (due to max) not differentiable in the parameters.

To limit overfitting and encourage a sparse, interpretable parameter weighting we use L1-regularization. The absolute value of all learned weights is subtracted from the objective function.

To learn the parameters for the inference models we apply a "black-box" optimization method: greedy-stochastic local search. Learning explores the parameter space, tending to search in regions of high value while never becoming stuck in a local maximum.

We also experimented with the Nelder-Mead simplex method (Nelder and Mead 1965) and the multi-directional search method of Torczon (1989) but found weaker performance from these methods.

Ensemble Learning

We have multiple inference methods, each approaching the problem of combining the subquestion confidences from a different intuition and formalizing it in a different way. To combine all these different approaches we train an ensemble.

This is a final, convex, confidence estimation over the multiple-choice answers using the predictions of the inference models as features.

The ensemble learning uses the same training set that the individual closed-form inference models use. To avoid giving excess weight to inference models that have overfit the training set, we use a common technique from stacking ensembles (Breiman 1996). The training set is split into five folds, each leaving out 20 percent of the training data, as though for cross validation. Each closed-form inference model is trained on each fold. When the ensemble gathers an inference model's confidence as a feature for an instance, the inference model uses the learned parameters from the fold that excludes that instance. In this way, each inference model's performance is testlike, and the ensemble model does not overly trust overfit models.

The ensemble is a binary logistic regression per answer hypothesis using three features from each inference model. The features used are: the probability of the hypothesis, the logit of the probability, and the rank of the answer among the multiple-choice answers. Using the logit of the probability ensures that selecting a single inference model is in the ensemble's hypothesis space, achieved by simply setting the weight for that model's logit feature to one and all other weights to zero.

Each closed-form inference model is also trained on the full training set. These versions are applied at test time to generate the features for the ensemble.

Evaluation

For the automatic evaluation of WatsonPaths, we used a set of medical test preparation questions from Exam Master and McGraw-Hill, which are analogous to the examples we have used throughout this article. These questions consist of a paragraph-sized natural language scenario description of a patient case, optionally accompanied by a semistructured tabular structure. The paragraph description typically ends with a punch line question and a set of multiple-choice answers (average 5.2 answer choices per question). We excluded from consideration questions that require image analysis or whose answers are not text segments.

The punch line questions may simply be seeking the most likely disease that caused the patient's symptoms (for example, "What is the most likely diagnosis in this patient?"), in which case the question is classified as a diagnosis question. The diagnosis question set reported in this evaluation was identified by independent annotators. Nondiagnosis punch line questions may include appropriate treatments, the organism causing the disease, and so on (for example, "What is the most appropriate treatment?" and "Which organism is the most likely cause of his meningitis?" respectively). We observed that most questions that did not directly ask for a diagnosis nonetheless required a diagnosis as an intermediate step. For this reason, we decided that focusing initially on diagnosis questions was a good strategy.

We split our data set of 2190 questions into a training set of 1000 questions, a development set of 690 questions, and a blind test set of 500 questions. The development set was used to iteratively drive the development of the scenario analysis, relation generation, and belief engine components, and for parameter tuning. The training set was used to build models used by the learning component.

As noted earlier, our learning process requires subquestion training data to consolidate groups of question-answering features into smaller, more manageable sets of features. We do not have robust and comprehensive ground truth for a sufficiently large set of our automatically generated subquestions. Instead, we use a preexisting set of simple factoid medical questions as subquestion training data: the Doctor's Dilemma (DD) question set.[1] DD is an established benchmark used to assess performance in factoid medical question answering. We use 1039 DD questions (with a known answer key) as our subquestion training data. Although the Doctor's Dilemma questions do have some basic similarity to the subquestions we ask in assertion graphs, there are some important differences: (1) In an assertion graph subquestion, there is usually one known entity and one relation that is being asked about. For DD, the question may constrain the answer by multiple entities and relations. (2) An assertion graph subquestion like "What causes hypertension?" has many correct answers, whereas DD questions have a single best answer. (3) There may be a mismatch between how confidence for DD is trained and how subquestion confidence is used in an inference method. The DD confidence model is trained to maximize log-likelihood on a correct/incorrect binary classification task. In contrast, many probabilistic inference methods use confidence as something like strength of indication or relevance.

For all these reasons, DD data might seem poorly suited to training a complete model for judging edge-strength for subquestion edges in WatsonPaths. In practice, however, we have found that DD data is useful as subquestion training data because it is easier to obtain than subquestion ground truth, and so far shows improved performance over the limited subquestion ground truth we have constructed.[2] In our hybrid learning approach, we use 1039 DD questions for consolidating question-answering features and then use the smaller, consolidated set of features as inputs to the inference models that are trained on the 1000 medical test preparation questions.

Metrics and Baseline

As a baseline, we attempted to answer questions from our tests set using a simple information-retrieval strategy. It used as much as possible the same corpus and starting point used by WatsonPaths. In this base-

		Full	Diagnosis
Accuracy	Baseline	30.6%	41.0%
	Watson	42.0%	53.8%
	WatsonPaths	**48.0%**	**64.1%**
Confidence Weighted Score	Baseline	42.9%	52.1%
	Watson	59.8%	75.3%
	WatsonPaths	**67.5%**	81.8%

Table 1. WatsonPaths Performance Results.

line, we took each sentence in the question and generated an Indri query by removing stop words. We then ran this query over our medical corpus, returning a set of 100 passages. (We also tried different numbers of passages on our development set; 100 passages appeared to show the best results.) The score for each candidate was simply the number of times that the candidate text appeared in any of these passages. Confidence in each answer was generated by normalizing the scores. For instance, if answer A appeared 4 times in the passages, and answer B appeared 1 time, the confidence in answer A would be 80 per cent.

We also evaluated the performance of the Watson question-answering system adapted for the medical domain (Ferrucci et al. 2013). We ran this factoid-based pipeline on our scenario-based questions in order to evaluate the value added by our scenario-based approach. Watson takes the entire scenario as input and evaluates each multiple-choice answer based on its likelihood of being the correct answer to the punch line question. This one-shot approach to answering medical scenario questions contrasts with the WatsonPaths approach of decomposing the scenario, asking questions of atomic factors, and performing probabilistic inference over the resulting graphic model. Note that Watson is the same system that WatsonPaths uses as a subcomponent. It has been developed and improved along with WatsonPaths.

We tuned various parameters in the WatsonPaths system on the development set to balance speed and performance. The system performs one iteration each of forward and backward relation generation. The minimum confidence threshold for expanding a node is 0.25, and the maximum number of nodes expanded per iteration is 40. In the relation generation component, the Watson medical question-answering system returns all answers with a confidence of above 0.01.

We evaluate system performance both on the full test set as well as on the diagnosis subset only. The reason for evaluating the diagnosis subset separately is because most questions that do not directly seek a diagnosis in the punch line depend on a correct diagnosis along the way. Thus progress on the diagnosis subset may be a step toward better performance on multistep questions. We use the full 1000 questions in the training set to learn the models for both the baseline system and the WatsonPaths system. As noted earlier, Doctor's Dilemma training data is used to consolidate question-answering features in the WatsonPaths system. In the Watson system that was not part of WatsonPaths, we did not use Doctor's Dilemma training data for any purpose.

Results

Table 1 shows the results of our evaluation on a set of 500 blind questions of which a subset of 156 questions were identified as diagnosis questions by annotators.

We report results on our blind evaluation data, using two metrics. *Accuracy* simply measures the percentage of questions for which a system ranks the correct answer in top position. *Confidence weighted score* is a metric that takes into account both the accuracy of the system and its confidence in producing the top answer (Voorhees 2003). We sort all <question, top answer> pairs in an evaluation set in decreasing order of the system's confidence in the top answer and compute the confidence weighted score as

$$\text{CWS} = \frac{1}{n}\sum_{i=1}^{n}\frac{\text{number correct in first } i \text{ ranks}}{i}$$

where n is the number of questions in the evaluation set. This metric rewards systems for more accurately assigning high confidences to correct answers, an important consideration for real-world question-answering and medical diagnosis systems.

Because most questions have five or six multiple-choice answers, chance performance on our test set was approximately 19.8 per cent.

Results show that in terms of accuracy, WatsonPaths outperforms both the baseline system and Watson on both the full set and the diagnosis subset. We used a significance level of $p < 0.05$. In terms of confidence weighted score, WatsonPaths significantly outperforms the baseline system on both sets, and significantly outperforms Watson on the full set. For the diagnosis subset, the difference between Watson and WatsonPaths on confidence weighted score was not statistically significant, despite a 6+ percent score increase. This is likely due to the small diagnosis subset, which contains only 156 questions.

Overall, these results suggest that WatsonPaths adds significant value to scenario-based question answering, over and above a simple information-retrieval baseline, and also over and above a factoid-type question-answering approach. This is true even when comparing WatsonPaths to the same Watson system that was developed as a subcomponent for WatsonPaths. These results suggest that the graphic

model, subquestion strategy, and probabilistic inference engines, such as the ones used in WatsonPaths, can add significant value to scenario-based question answering.

Discussion

WatsonPaths has some key features that drive the performance improvement over Watson. The first and most important is that WatsonPaths has the ability to engage in inference. Watson does well on short diagnostic questions where one phrase is strongly associated with the correct diagnosis. WatsonPaths does better than Watson when there is another factor that rules out a diagnosis that would otherwise be likely, or a second symptom that is not explained by that diagnosis. Holistically performing inference over the information contained in the question is often necessary to answer such questions well.

Development of the inference capabilities of WatsonPaths has been driven by empirical results: We continued to add sophistication to inference as long as we detected a statistically significant improvement in accuracy in the development set. The framework supports many more kinds of inference. Some types of inference (for instance, causal inference) already have implementations, but have not yet shown a statistically significant impact. For some of these types of inference, we suspect that improvements in the underlying Watson question answering will be necessary before this impact will emerge. For some other kinds of inference, such as reasoning about events in time, we are not convinced that such inference will be necessary to do well on the United States Medical Licensing Examination (USMLE) test set. Finally, some kinds of inference are not well supported by WatsonPaths. For instance, as we mentioned, statements in WatsonPaths must be either true or false. Thus explicit reasoning about entities and events, and the relations between them, would require a major extension to WatsonPaths. Overall, because the primary impact of WatsonPaths appears to be its more advanced inference, and further advancing inference depends on the quality of Watson's results, we believe that the biggest gains in the performance of WatsonPaths will come from improvements in Watson.

Another factor contributing to the impact of WatsonPaths, is that WatsonPaths seems less likely than Watson to get overwhelmed by irrelevant details in long questions. While Watson tries to weight the relevance of various phrases in the question, its baseline assumption is that all the text is potentially important. Thus irrelevant text can water down the score of a candidate hypothesis that would otherwise get a high score. In WatsonPaths, we ask many subquestions using different ways of breaking down the scenario. For instance, we ask questions about sentences, factors, and combinations of factors. This increases the chances that some set of words will produce a strong inference chain that connects to a hypothesis. In contrast, irrelevant text will be unlikely to produce inference chains to the hypotheses. This property is important as many real-world applications are not as concise as trivia questions. For instance, medical records often contain large amounts of detail, much of which is irrelevant to a particular question.

Related Work

Clinical decision support systems (CDSSs) have had a long history of development starting from the early days of artificial intelligence. These systems use a variety of knowledge representations, reasoning processes, system architectures, scope of medical domain, and types of decision (Musen, Middleton, and Greenes 2014). Although several studies have reported on the success of CDSS implementations in improving clinical outcomes (Kawamoto et al. 2005; Roshanov et al. 2013), widespread adoption and routine use is still lacking (Osheroff et al. 2007).

The pioneering Leeds abdominal pain system (De Dombal et al. 1972) used structured knowledge in the form of conditional probabilities for diseases and their symptoms. Its success at using Bayesian reasoning was comparable to experienced clinicians at the Leeds hospital where it was developed. But it did not adapt successfully to other hospitals or regions, indicating the brittleness of some systems when they are separated from their original developers. A recent systemic review of 162 CDSS implementations shows that success at clinical trials is significantly associated with systems that were evaluated by their own developers (Roshanov et al. 2013). MYCIN (Shortliffe 1976) was another early system that used structured representation in the form of production rules. Its scope was limited to the treatment of infectious diseases and, as with other systems with structured knowledge bases, required expert humans to develop and maintain these production rules. This manual process can prove to be infeasible in many medical specialties where active research produces new diagnosis and treatment guidelines and phases out older ones. Many CDSS implementations mitigate this limitation by focusing their manual decision logic development effort on clinical guidelines for specific diseases or treatments, for example, hypertension management (Goldstein et al. 2001). But such systems lack the ability to handle patient comorbidities and concurrent treatment plans (Sittig et al. 2008). Another notable system that used structured knowledge was Internist-1. The knowledge base contained disease-to-finding mappings represented as conditional probabilities (of disease given finding, and of finding given disease) mapped to a 1–5 scale. Despite initial success as a diagnostic tool, its design as an expert consultant was not considered to meet the information needs of most physicians. Eventually, its

underlying knowledge base helped its evolution into an electronic reference that can provide physicians with customized information (Miller et al. 1986). A similar system, DXplain (Barnett et al. 1987), continues to be commercially successful and extensively used. Rather than focus on a definitive diagnosis, it provides the physician with a list of differential diagnoses along with descriptive information and bibliographic references.

Other systems in commercial use have adopted the unstructured medical text reference approach directly, using search technology to provide decision support. Isabel provides diagnostic support using natural language processing of medical textbooks and journals. Other commercial systems like UpToDate and ClinicalKey forgo the diagnostic support and provide a search capability to their medical textbooks and other unstructured references. Although search over unstructured content makes it easier to incorporate new knowledge, it shifts the reasoning load from the system back to the physician.

In comparison to the aforementioned systems, WatsonPaths uses a hybrid approach. It uses question-answering technology over unstructured medical content to obtain answers to specific subquestions generated by WatsonPaths. For this task, it builds on the search functionality by extracting answer entities from the search results and seeking supporting evidence for them in order to estimate answer confidences. These answers are then treated as inferences by WatsonPaths over which it can perform probabilistic reasoning without requiring a probabilistic knowledge base.

Another major area of difference between CDSS implementations is the extent of their integration to the health information system and workflow used by the physicians. Studies have shown that CDSSs are most effective when they are integrated within the workflow (Kawamoto et al. 2005; Roshanov et al. 2013). Many of the guideline-based CDSS implementations are integrated with the health information system and workflow, having access to the data being entered and providing timely decision support in the form of alerts. But this integration is limited to the structured data contained in a patient's electronic medical record. When a CDSS requires information like findings, assessments, or plans in clinical notes written by a health-care provider, existing systems are unable to extract them. As a result, search-based CDSSs remain a separate consultative tool. The scenario analysis capability of WatsonPaths provides the means to analyze these unstructured clinical notes and serves as a means for integration into the health information system.

Conclusions and Further Work

WatsonPaths is a system for scenario-based question answering that has a graphic model at its core. We have developed WatsonPaths on a set of multiple-choice questions from the medical domain. On this test set, WatsonPaths shows a significant improvement over our baselines, even outperforming its own subquestion-answering system, Watson. Although the test preparation question set has been important for the early development of the system, we have designed WatsonPaths to function well beyond it. In future work, we plan to extend WatsonPaths in several ways.

The present set of questions are all multiple-choice questions. This means that hypotheses have already been identified, and it is also known that exactly one of the hypotheses is the correct answer. Although they have made the early development of scenario-based question answering more straightforward, the overall WatsonPaths architecture does not rely on these constraints. For instance, we can easily remove the confidence reestimation phase for the closed-form inference systems and the "exactly one" constraint from the belief engine. Also, it will be straightforward to add a simple hypothesis identification step to the main loop. One way to do this is to find nodes whose type corresponds to the type being asked about in the punch line question. We already find such correspondences in the base Watson system (Chu-Carroll et al. 2012).

We also plan to extend WatsonPaths beyond the medical domain. For medical applications, it might have been easier to design Watson with certain medical aspects hard coded into the flow of execution. Instead we designed the overall flow as well as each component to be general across domains. Note that the Emerald could be replaced by a structure from a different domain, and the basic semantics we have explored: matching, indicative and causal, have no requirement that the graph structure come from medicine. Even the causal aspect of the belief engine could apply to any domain that involves diagnostic inference (for example, automotive repair). Most importantly, the way that subquestions are answered is completely general. By asking the right subquestions and using the right corpus, we can apply WatsonPaths to any scenario-based question-answering problem. We hope to develop a toolbox of expansion strategies, relation generators, and inference mechanisms that can be reused as we apply WatsonPaths to new domains.

The most important area for further work is on a collaborative user application. Question answering is not always just about returning the correct answer — often we must also explain to the user why the answer is correct. This is particularly important in domains like medicine, where users are justfied in challenging and validating answers because of the life-and-death nature of decisions. Question-answering systems that rely heavily on machine learning are often criticized for being too opaque to allow clear explanations. We have designed WatsonPaths with explanatory power in mind. For instance, we modeled the graphic model after the hand-drawn graphs that domain experts used to explain their answers. At the same time, we are able to use machine learning to improve our accuracy and other metrics, without losing the ability to explain our answers.

In addition to explaining our answers, a collaborative application gives us the opportunity to have humans assist WatsonPaths in tasks that are still difficult for machines. For instance, factor identification and supporting passage evaluation may benefit

from human input. In a fully automatic system, the user receives an answer using little or no time or cognitive effort. In a collaborative system, the user spends some time and effort, and potentially gets a better answer. We suspect that, in many applications of scenario-based question answering, this will be an attractive trade-off for the user, because of the complexity of the scenario and the importance of the answer. Our objective is to minimize the time and effort required of users and to maximize the benefit they receive. The combination of the user and WatsonPaths should be able to handle more difficult problems more quickly than either could alone.

Acknowledgements

We would like to acknowledge the following colleagues at IBM who helped to create WatsonPaths: Ken Barker, Eric Brown, James Fan, Bhavani Iyer, Benjamin Segal, Parthasarathy Suryanarayanan, and Chris Welty. We would also like to acknowledge the Cleveland Clinic Lerner College of Medicine of Case Western Reserve University, with whom we are working to show positive impact of WatsonPaths on educational outcomes. This work was conducted at IBM Research and IBM Watson Group, Thomas J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598.

Notes

1. See the American College of Physicians 2014 Doctor's Dilemma competition, www.acponline.org/residents fellows/competitions/doctors dilemma.

2. Building a comprehensive answer key for such questions is very time consuming, and an incomplete answer key can be less effective. Although this approach has not yet succeeded, it may still succeed if we invest much more in building a bigger, better answer key for actual WatsonPaths subquestions.

References

Barnett, G. O.; Cimino, J. J.; Hupp, J. A.; and Hoffer, E. P. 1987. DXplain: An Evolving Diagnostic Decision-Support System. *JAMA* 258(1): 67–74. doi.org/10.1001/jama.1987.03400010071030

Bowen, J. L. 2006. Educational Strategies to Promote Clinical Diagnostic Reasoning. *New England Journal of Medicine* 355(21): 2217–2225. doi.org/10.1056/NEJMra054782

Breiman, L. 1996. Stacked Regressions. *Machine Learning* 24(1): 49–64. doi.org/10.1007/BF00117832

Chang, R. W.; Bordage, G.; and Connell, K. J. 1998. Cognition, Confidence, and Clinical Skills: The Importance of Early Problem Representation During Case Presentations. *Academic Medicine* 73(10): S109–111. doi.org/10.1097/00001888-199810000-00062

Cheng, P. W. 1997. From Covariation to Causation: A Causal Power Theory. *Psychological Review* 104(2): 367. doi.org/10.1037/0033-295X.104.2.367

Chu-Carroll, J.; Fan, J.; Boguraev, B. K.; Carmel, D.; Sheinwald, D.; and Welty, C. 2012. Finding Needles in the Haystack: Search and Candidate Generation. *IBM Journal of Research and Development* 56(3/4): 6: 1–6: 12.

De Dombal, F. T.; Leaper, D. J.; Staniland, J. R.; McCann, A. P.; and Horrocks, J. C. 1972. Computer-Aided Diagnosis of Acute Abdominal Pain. *British Medical Journal* 2(5804): 9. doi.org/10.1136/bmj.2.5804.9

Ferrucci, D. 2012. Introduction to This is Watson. *IBM Journal of Research and Development* 56(3/4): 1: 1–1: 15.

Ferrucci, D.; Brown, E.; Chu-Carroll, J.; Fan, J.; Gondek, D.; Kalyanpur, A. A.; Lally, A.; Murdock, J. W.; Nyberg, E.; Prager, J.; Schlaefer, N.; and Welty, C. 2010. Building Watson: An Overview of the DeepQA Project. *AI Magazine* 31(3): 59–79. doi.org/10.1016/j.artint.2012.06.009

Ferrucci, D.; Levas, A.; Bagchi, S.; Gondek, D.; and Mueller, E. T. 2013. Watson: Beyond Jeopardy! *Artificial Intelligence* 199–200(June–July): 93–105.

Goldstein, M. K.; Hoffman, B. B.; Coleman, R. W.; Tu, S. W.; Shankar, R. D.; O'Connor, M.; Martins, S.; Advani, A.; and Musen, M. A. 2001. Patient Safety in Guideline-Based Decision Support for Hypertension Management: ATHENA DSS. In *Proceedings of the 2001 AMIA Annual Symposium,* 214–218. Bethesda, MD: American Medical Informatics Association.

Gondek, D. C.; Lally, A.; Kalyanpur, A.; Murdock, J. W.; Duboue, P. A.; Zhang, L.; Pan, Y.; Qiu, Z. M.; and Welty, C. 2012. A Framework for Merging and Ranking of Answers in DeepQA. *IBM Journal of Research and Development* 56(3/4): 14: 1–14: 12.

Kawamoto, K.; Houlihan, C. A.; Balas, E. A.; and Lobach, D. F. 2005. Improving Clinical Practice Using Clinical Decision Support Systems: A Systematic Review of Trials to Identify Features Critical to Success. *British Medical Journal* 330(7494): 765–72. dx.doi.org/10.1136/bmj.38398.500764.8F

Lafferty, J.; McCallum, A.; and Pereira, F. C. N. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the 18th International Conference on Machine Learning 2001* (ICML 2001), 282–289. San Francisco: Morgan Kaufmann Publishers.

Lally, A.; Prager, J. M.; McCord, M. C.; Boguraev, B. K.; Patwardhan, S.; Fan, J.; Fodor, P.; and Chu-Carroll, J. 2012. Question Analysis: How Watson Reads a Clue. *IBM Journal of Research and Development* 56(3/4): 2: 1–2: 14.

McCord, M. C.; Murdock, J. W.; and Boguraev, B. K. 2012. Deep Parsing in Watson. *IBM Journal of Research and Development* 56(3/4): 3: 1–3: 15.

Miller, R. A.; McNeil, M. A.; Challinor, S. M.; Masarie, Jr, F. E.; and Myers, J. D. 1986. The Internist-1/Quick Medical Reference Project — Status Report. *Western Journal of Medicine* 145(6): 816.

Murdock, J. W.; Fan, J.; Lally, A.; Shima, H.; and Boguraev, B. K. 2012a. Textual Evidence Gathering and Analysis. I*BM Journal of Research and Development* 56(3/4): 8: 1–8: 14.

Murdock, J. W.; Kalyanpur, A.; Welty, C.; Fan, J.; Ferrucci, D.; Gondek, D. C.; Zhang, L.; and Kanayama, H. 2012b. Typing Candidate Answers Using Type Coercion. *IBM Journal of Research and Development* 56(3/4): 7: 1–7: 13.

Musen, M. A.; Middleton, B.; and Greenes, R. A. 2014. Clinical Decision-Support Systems. In *Biomedical Informatics: Computer Applications in Health Care and Biomedicine,* ed. E. H. Shortliffe and J. J. Cimino, 643–674. Berlin: Springer. doi.org/10.1007/978-1-4471-4474-8_22

National Library of Medicine (US). 2009. UMLS Reference Manual. Bethesda, MD: National Library of Medicine (US). www.ncbi.nlm.nih.gov/books/NBK9676/.

Nelder, J. A., and Mead, R. 1965. A Simplex Method for Function Minimization. *Computer Journal* 7(4): 308–313. doi.org/10.1093/comjnl/7.4.308

Osheroff, J. A.; Teich, J. M.; Middleton, B.; Steen, E. B.; Wright, A.; and Detmer, D. E. 2007. A Roadmap for National Action on Clinical Decision Support. *Journal of the American Medical Informatics Association* 14(2): 141–145. doi.org/10.1197/jamia.M2334

Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco: Morgan Kaufmann.

Prager, J. M.; Chu-Carroll, J.; and Czuba, K. 2004. Question Answering Using Constraint Satisfaction: QA-by-Dossier-with-

Constraints. In *Proceedings of the 42nd Association for Computational Linguistics,* 575–582. Stroudsburg, PA: Association for Computational Linguistics, Inc. doi.org/10.3115/1218955.1219028

Roshanov, P. S.; Fernandes, N.; Wilczynski, J. M.; Hemens, B. J.; You, J. J.; Handler, S. M.; Nieuwlaat, R.; Souza, N. M.; Beyene, J.; Spall, H. G. C. V.; Garg, A. X.; and Haynes, R. B. 2013. Features of Effective Computerised Clinical Decision Support Systems: Meta-Regression of 162 Randomised Trials. *British Medical Journal* 346(f657). doi.org/10.1136/bmj.f657

Shortliffe, E. H. 1976. *MYCIN: Computer-Based Medical Consultations*. New York: Elsevier.

Sittig, D. F.; Wright, A.; Osheroff, J. A.; Middleton, B.; Teich, J. M.; Ash, J. S.; Campbell, E.; and Bates, D. W. 2008. Grand Challenges in Clinical Decision Support. *Journal of Biomedical Informatics* 41(2): 387–392. doi.org/10.1016/j.jbi.2007.09.003

Torczon, V. J. 1989. Multi-Directional Search: A Direct Search Algorithm for Parallel Machines. Ph.D. Dissertation, Department of Mathematical Sciences, Rice University, Houston, TX.

Voorhees, E. M. 2003. Overview of TREC 2002. In *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003)*. Gaithersburg, MD: National Institute of Standards and Technology.

Yuille, A. L., and Lu, H. 2007. The Noisy-Logical Distribution and Its Application to Causal Inference. In *Advances in Neural Information Processing Systems 20,* ed. J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis. Red Hook, NY: Curran Associates, Inc.

Adam Lally is the primary developer of the WatsonPaths system architecture as well as the original Watson question-answering system architecture. His background includes developing natural language-processing algorithms and frameworks for a variety of applications.

Sugato Bagchi is a research staff member at IBM Research and IBM Watson Group. He works on decision support systems based on unstructured information analytics. Currently, he is part of a team that is working with leading medical institutions on developing analytics that can help physicians make clinical decisions using content from medical textbooks, evidence-based guidelines, and patient consultation notes.

Michael A. Barborak has been a technical leader of the WatsonPaths project, which has as its mission to advance cognitive computing through the application of Watson question answering to complex decision support tasks. He has an entrepreneurial background in software development and technical services. His education is in electrical engineering with degrees from the University of Texas at Austin (BSEE 1989 and MSEE 1992).

David W. Buchanan is interested in joint inference problems in large AI systems, especially using Bayesian approaches. He built the core inference system for WatsonPaths and other systems. He has published in causal inference, graphic models, and hierarchical Bayesian models. He earned his Ph.D. in Cognitive Science from Brown University in 2011.

Jennifer Chu-Carroll serves on the editorial boards of the *Journal of Dialogue Systems* and the *ACM Transactions of Intelligent Systems and Technology.* She previously served on the executive board of the North American Chapter of the Association for Computational Linguistics and as general chair of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics — Human Language Technologies. Her research interests include question answering, semantic search, and natural language discourse and dialogue.

David A. Ferrucci is an award-winning AI researcher and IBM Fellow. He worked at IBM Research from 1995–2012 where he built and led a team of researchers to create the open-source UIMA framework and Watson, the landmark open-domain question-answering system. In 2012, after initiating the work on WatsonPaths and its application to medicine, he joined Bridgewater where he is exploring ways to combine machine learning and semantic technologies to accelerate the discovery, application, and continuous refinement of interpretable knowledge systems.

Michael R. Glass is a software engineer at IBM Research. His research focuses on deep learning in natural language processing and probabilistic joint inference. Glass completed his doctorate in computer science at the University of Texas at Austin in 2012.

Aditya A. Kalyanpur's primary research interests include knowledge representation and reasoning, natural language processing, machine learning, and question answering. He has served on W3C working groups, as program chair of several international workshops, and is currently a coeditor of the *International Journal of Web Information Systems*. Kalyanpur earned his Ph.D. from the University of Maryland, College Park in 2006.

Erik T. Mueller's books include *Commonsense Reasoning, Daydreaming in Humans and Machines,* and *Natural Language Processing with ThoughtTreasure*. He received an S.B. in computer science and engineering from the Massachusetts Institute of Technology and an M.S. and Ph.D. in computer science from the University of California, Los Angeles.

J. William Murdock is a research staff member in IBM Watson Group. Before joining IBM, he worked at the United States Naval Research Laboratory. His research interests include question answering, natural language semantics, analogical reasoning, knowledge-based planning, machine learning, and computational reflection. In 2001, he earned his Ph.D. in computer science from the Georgia Institute of Technology.

Siddharth Patwardhan pursues research in natural language processing, with interests spanning a variety of topics, including information extraction, question answering, and computational representation of lexical semantics. He has investigated these topics as a researcher at IBM, the University of Utah, and the Mayo Clinic and at the University of Minnesota. He was conferred a Ph.D. in computer science by the University of Utah in 2009, and is the coauthor of more than 30 peer-reviewed technical publications.

John M. Prager is a research staff member at IBM Research. His background includes natural language–based interfaces and semantic search, and his current interest is on incorporating user and domain models to inform question answering, in particular in the medical domain. He is a member of the TREC program committee.