# QUADS: Question Answering for Decision Support

Zi Yang
Language Technology Institute
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
ziy@cs.cmu.edu

Ying Li James Cai
pRED Informatics
Roche Innovation Center
430 E 29th St
New York, NY 10016
{ying_l.li, james.cai}@roche.com

Eric Nyberg
Language Technology Institute
Carnegie Mellon University
5000 Forbes Ave
Pittsburgh, PA 15213
ehn@cs.cmu.edu

## ABSTRACT

As the scale of available on-line data grows ever larger, individuals and businesses must cope with increasing complexity in decision-making processes which utilize large volumes of unstructured, semi-structured and/or structured data to satisfy multiple, interrelated information needs which contribute to an overall decision. Traditional decision support systems (DSSs) have been developed to address this need, but such systems are typically expensive to build, and are purpose-built for a particular decision-making scenario, making them difficult to extend or adapt to new decision scenarios. In this paper, we propose a novel decision representation which allows decision makers to formulate and organize natural language questions or assertions into an analytic hierarchy, which can be evaluated as part of an *ad hoc* decision process or as a documented, repeatable analytic process. We then introduce a new decision support framework, QUADS, which takes advantage of automatic question answering (QA) technologies to automatically understand and process a decision representation, producing a final decision by gathering and weighting answers to individual questions using a Bayesian learning and inference process. An open source framework implementation is presented and applied to two real world applications: *target validation*, a fundamental decision-making task for the pharmaceutical industry, and *product recommendation from review texts*, an everyday decision-making situation faced by on-line consumers. In both applications, we implemented and compared a number of decision synthesis algorithms, and present experimental results which demonstrate the performance of the QUADS approach versus other baseline approaches.

## Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.4.2 [**Information Systems Applications**]: Types of Systems—*Decision support*; J.3 [**Computer Applications**]: Life and Medical Sciences

## Keywords

question answering; decision support; target validation; product recommendation

*SIGIR'14,* July 6–11, 2014, Gold Coast, Queensland, Australia.
Copyright is held by the owner/author(s). Publication rights licensed to ACM.
ACM 978-1-4503-2257-7/14/07 ...$15.00.
http://dx.doi.org/10.1145/2600428.2609606.

## 1. INTRODUCTION

As the scale of available on-line data grows ever larger, individuals and businesses must cope with increasing complexity in decision-making processes which utilize large volumes of unstructured, semi-structured and/or structured data to satisfy multiple, interrelated information needs which contribute to an overall decision. The field of *decision support systems* (DSS) [33, 4] has focused on how to help decision makers to: a) structure a decision-making process interactively, b) automatically access and retrieve decision-supporting information, c) adapt decision models or analytic techniques for evidence fusion, and d) produce a final report [19, 35, 4]. However, traditional DSSs mostly suffer from inflexibility in adapting to new decision scenarios. First, a well-tailored DSS is usually developed for one specific decision problem, such that users are unable to dynamically input new decision scenarios the way that users of an information retrieval system can formulate novel queries. Second, data retrieval and processing typically follows a fixed flow within a DSS, which does not allow decision makers to easily leverage additional decision support information, integrate data analysis algorithms or tools, or further reconfigure the decision support pipeline for better performance. A large team of developers who have some acquaintance with the decision problem and the application area are typically involved in development and deployment of DSSs [33]. As a result, DSSs are normally very expensive to build and maintain.

Defining a formal decision representation that accurately reflects a decision maker's information need is a challenging problem [34]. To support a more direct and effective communication between decision makers and the underlying analytic models and tools, and to provide a more flexible solution, we propose a novel decision representation that allows decision makers to formulate and organize natural language questions or assertions in an analytic hierarchy [31], which has been shown to be the most natural and least ambiguous way to express an information need [28]. We then introduce a new decision support framework, QUADS, which takes advantage of question answering (QA) technologies to automatically understand and process the decision scenario in real time, providing an overall weighted decision by combining evidence from answers to individual questions [19, 2].

Question answering and decision support systems have been independently developed for decades. With the recent development of high-performance question answering (QA) systems, which combine natural language processing and information retrieval, users can directly interact with an information system to evaluate evidence gathered automatically; IBM's Watson [7] is a prominent example. Only recently has automatic question answering been applied in support of complex human decision making for specific domains, e.g. cancer diagnosis [20].

To date, a general approach for decision support using QA technologies has not been thoroughly and systematically studied. Our goal in this work is to outline a formal procedure for representing any decision-making process as a complex question answering scenario. Following the desiderata expressed by other decision theorists [9, 6], we focus on tackling the following challenges in this paper:

- How should users formally characterize a decision process using natural language descriptions? How should they convert existing best practices (decision-making templates) into a decision scenario representation? Is there a way to express personal preferences and prior knowledge in the decision-making process?
- How do we process a decision scenario by assessing and prioritizing the various decision factors based on the available evidence (text resources)? Can we implement a flexible and extendable open source framework that allows users to easily define and refine decision scenarios and integrate various question-answering components?
- How can we apply the decision framework to real-world decision problems, such as target validation in the pharmaceutical industry? In particular, how can we formulate the real-world decision process as a decision scenario, and identify textual resources that are specific to the task? Can we apply learning techniques to determine how decision factors influence the decision, given sample decisions to learn from?

In this paper, we introduce a directed acyclic graph (DAG) representation for decision processes, and formalize the decision making problem in the context of natural language question answering. The decision process is then mapped to a Bayesian decision representation, to jointly model the entire decision scenario (including the users' preferences, assertions from QA systems, and hierarchical decision factors) in a unified model. We present a message-passing algorithm that is used to predict decisions and learn decision models from labeled examples. The QUADS framework was implemented as an open source software package, with wrappers for Apache UIMA[1] and the CSE framework [37] to support: a) integration of any QA system deployed as a UIMA Aggregate Analysis Engine, and b) configuration, optimization and extension of the QA system with various decision synthesis strategies. Finally, we describe how the QUADS framework was leveraged to help validate potential gene targets for diseases, one of the most fundamental tasks in the life sciences, using a real-world dataset containing nearly 6K diseases and over 6K known gene candidates. We also describe another application where QUADS was used to implement a decision process that recommends products based on on-line reviews. Figure 1 shows the overview architecture diagram of the QUADS framework.

## 2. RELATED WORK

Two research areas related to this work – *decision analysis* and *question answering* – have been extensively studied for decades. In this section, we first survey prior work in the field of decision analysis, focusing on comparison of decision analysis models and decision making approaches, and present a few decision support systems. We then focus on reviewing related work in the question answering domain, especially question decomposition methods and prior work on scenario or contextual question answering.

[1]`http://uima.apache.org`

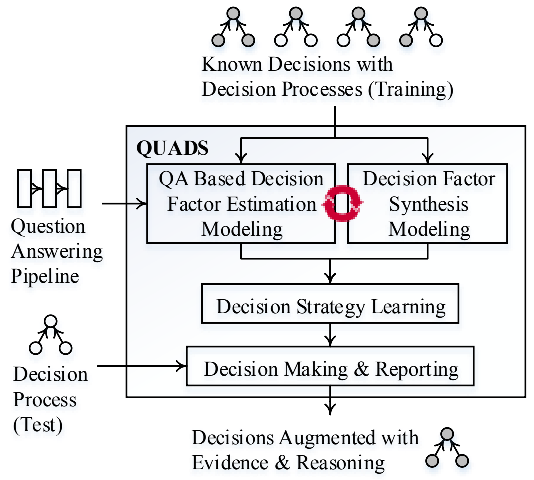

Known Decisions with Decision Processes (Training)
QUADS
QA Based Decision Factor Estimation Modeling
Decision Factor Synthesis Modeling
Question Answering Pipeline
Decision Strategy Learning
Decision Process (Test)
Decision Making & Reporting
Decisions Augmented with Evidence & Reasoning

Figure 1: Architecture overview of QUADS framework

**Decision Analysis Models.** Decisions are usually represented by a hierarchy or network of sub-criteria or sub-problems that are more easily comprehended or evaluated in isolation. Example representations include *analytic hierarchy process* (AHP), or *analytic network process* (ANP) representation [31], decision tree, or more compact *influence diagram* (ID, or decision diagram, a representation of Bayesian decision problems) [10], etc. AHP focuses on one single uncertainty with multiple alternatives, and decision makers are not allowed to explicitly assign priority judgments to non-*primitive criteria*. In contrast, an ID can more flexibly model complex decision situations with multiple alternatives from multiple uncertainties, and allows the decision makers to specify their preference at any node in the diagram. In this paper, we introduce a directed acyclic graph (DAG) representation for decision processes, and provide a solution framework that automatically finds answers to each question and gathers evidence from all decision factors.

**Decision Support Systems.** Many DSSs have been developed in support of various decision scenarios during the past few decades [19, 35, 4]. Although most DSSs are built for predetermined tasks, which requires domain knowledge obtained from humans and stored in a structured data store with a predefined schema, a few DSSs have explored how to understand decision needs from natural language input. For example, the Structured Evidential Argumentation System (SEAS) [19] provides decision references for national security crisis warnings, and Intelligent Decision System (IDS) [35] analyzes business innovation self-assessments written in free-text. However, neither approach is general or extensible enough to cover the cases we implemented with our proposed solution and open-source implementation.

**Question Decomposition.** Complex factoid questions usually contain more than one factor or assertion about the answers [12], and the question answering systems need to correctly identify each factor, reconstruct subquestions, and merge evidence from subquestions. Most approaches leverage syntactic or semantic properties of the question, e.g. lexical cues [12, 32], coreference [12, 16], relation triples [13], semantic dependencies [16] to identify subquestions, and then deploy discourse [12] or semantic decomposition such as knowledge templates [13, 29], textual entailment [16], constraint networks [29]. Compared with question decomposition in open-domain QA (such as Watson [12, 7]), where the entire information need might be expressed in a single, paragraph-length question, QUADS assumes that the decision maker has already decomposed the overall information need into individual, atomic subquestions that can be understood (and answered) independently, based on best practices and templates developed for domain decision-making.

**Scenario and Contextual QA.** Our work is directly related to prior work on scenario QA, where the user's input can include background information, questions with multiple parts, and even follow-up questions, requiring a more complex representation of the information need [3]. However, little research has been done to systematically study how to support global decision making using a scenario QA system. Contextual QA is a related research area which focuses on how systems can automatically track the information context through a series of questions in dialogs with real users [36]; however, this research has focused primarily on context identification using discourse analysis to improve answering of individual questions, rather than hierarchical decision-making.

## 3. DECISION MAKING IN QA CONTEXT

In this section, we formally define the concepts related to the decision process and provide a formal definition of the decision making problem in the context of question answering.

For complex multi-criteria decision problems, decision analysis usually employs the general divide-and-conquer approach, via problem reduction and solution synthesis phases. In the problem reduction phase, decision situations are often formalized by a recursive procedure that decomposes the decision goal into a hierarchy or network of sub-criteria or sub-problems that are more easily comprehended and/or evaluated independently [23]. In the solution synthesis phase, the global priority or probability of each option for each uncertainty is estimated by synthesizing the local priorities that are estimated against primitive criteria. Following the conventional analytic approach, we formally define *decision factor* and *decision process* respectively.

**Definition 1** (Decision factor, dependency)**.** *Given a* ***decision goal*** *$f_0$ with* ***alternatives*** *$\mathbf{a} = \{a_k\}_k$, a* ***decision factor*** *$f$ is an aspect that may impact decision making. A* ***preference*** *variable $y_i$ is associated with each decision factor $f_i$ with possible* ***outcomes*** *identical to the alternative set $\mathbf{a}$. A conditional* ***dependency*** *$p(y_i|y_j)$ is established between factors $f_j$ and $f_i$ if the aspect of the problem that $f_j$ implies can directly contribute to the decision of $f_i$.*

Both probabilities $\{p(y_i)\}_i$ and $\{p(y_i|y_j)\}_{i,j}$ can be specified to reflect a decision maker's personal preference or prior knowledge of how to prioritize the options for each factor, and how to make a decision based on dependent factors. If users do not have any preferences, we may always assume that they follow uniform distributions assumption.

**Definition 2** (Decision process)**.** *A* ***decision process*** *is described by a directed acyclic graph $S = (Y, E)$ with the decision goal $y_0$ being the only sink, the node set $Y = \{y_i\}_i$ representing the decision factors and the edge set $E$ representing conditional dependencies between factors. Given a decision process, we use $F_i$ to denote the index set of all dependent factors of $y_i$. If $F_i = \emptyset$, then we call $f_i$ a* ***primitive*** *factor.*

A simple decision process example is shown in Figure 2a, which consists of three random variables $y_0$, $y_1$, and $y_2$, representing factors $f_0$, $f_1$, and $f_2$. Factor $f_0$ is analytically decomposed into two dependent factors $f_1$ and $f_2$. We note that different directions or depths of analytic thinking may lead to different decision processes with distinct decision factors and/or dependencies even if the decision goal is the same. For example, a primitive factor in one decision process can be further decomposed into sub-factors in a more in-depth decision process. We will experiment with different levels of process of the same decision goal in Section 5 to demonstrate how decision problem solving is different from traditional single question answering.

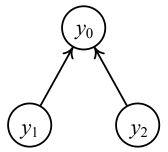
Traditional decision analysis approaches such as analytic hierarchy process (AHP) [10] can help decision makers to outline the decision process in a similar fashion, but cannot automate the evaluation process for each factor to make any overall conclusion. An automatic question answering system is able to overcome this problem by evaluating all possible assertions or input questions associated with each factor to produce an estimation, for multiple types of questions (e.g. factoid questions, yes/no questions, Jeopardy! clues[7], etc.), and evidence gathering and evidence-based answer reranking have become important research topics in QA [24]. To enrich the evidence, various question decomposition approaches have been proposed to analyze factors within the question [12, 32, 16, 13, 29, 29]. However, the additional information gained from the decomposed questions must be combined effectively to support overall decision-making, which can be improved by incorporating human intuitions and reasoning strategies from decision analysis (e.g. weights for sub-factors and rules for combining evidence). In this paper, we study how to leverage the mutual benefit of decision analysis and question answering for better overall decision support. Before we give the definition of the problem, we first define the *question answering process* as a probabilistic event, as follows:

**Definition 3** (Question answering process)**.** *Given a factor described by a natural language question in our context, a* ***question answering (QA) process*** *produces assertions assigned with probabilities, and thus we associate the QA process for factor $f_i$ with a random variable $x_i$ representing the result estimated by the process and* ***outcomes*** *$\mathbf{o}_i = \{o_{ik}\}_k$ being all possible assertions or candidate answers.*

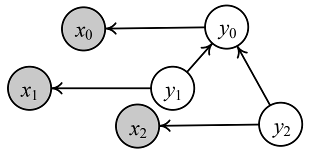
For example, in a decision problem to choose a cell phone, the decision maker may create two question answering processes: "is the PHONE light" and "what is the brand of the PHONE". Then the outcomes for the first factor $\mathbf{o}_1$ can be within {`light, heavy`}, and the outcomes for the second factor $\mathbf{o}_2$ can be the set of all phone brands. We note that a mixture of factoid questions and yes/no questions or assertions can co-exist in our setting. We now give the definition of *decision making problem in the QA context*.

**Definition 4** (Decision making problem in the QA context)**.** *Given a decision process $S$ representing a decision maker's preferences and question answering pipelines producing probabilistic assertions for the decision factors in $S$, the* ***decision making problem in the QA context*** *aims to make the optimal decision for $S$ by prioritizing alternatives $\{a_k\}_k$ for the decision goal $f_0$, with considerations of decision maker's preferences, assertions produced from QA processes, and augmented evidence from dependent decisions.*

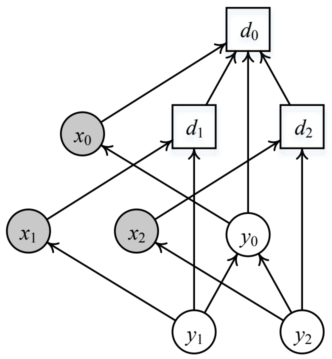
Unlike simple question answering, the decision making problem requires an analytic decision process instead of a single natural language question as its input. A user can specify a handcrafted decision process specific to a particular decision situation, which is referred to as an *ad hoc decision process*, whereas instantiating a generic *decision process template* (DPT) for specific decision problems is an alternative way to create a decision process. In contrast to a decision process, assertions in a DPT may contain variables. For example, users can create a DPT *a priori* if repeated decision support is required for similar decision situations, or in another case, novices may want to use DPTs created by more experienced users as a starting point. Furthermore, a DPT can also group decision processes to alleviate the data sparsity problem, by



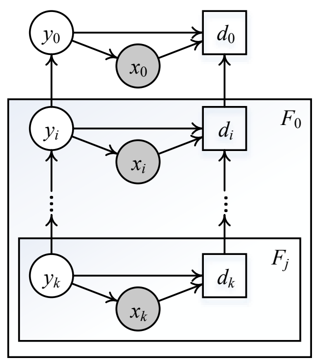
(a) A simple decision process consisting three decision factors $f_0$, $f_1$, $f_2$, where $y_0$ is a random variable for the decision goal $f_0$, and $y_1$ and $y_2$ represent the true facts of two aspects.



(b) Factors $f_0$, $f_1$, and $f_2$ are independently considered by a QA component, which estimates each $y$ using an observed answer $x$.



(c) Factor synthesis by additional synthesis components $d_0$, $d_1$, and $d_2$. Outputs from all synthesized sub-factors ($d_1$ and $d_2$) are merged with the primitive estimation for the factor ($x_0$) to produce $d_0$.



(d) Plate notation for a complex decision process for a general decision problem with factors $y_0, \ldots, y_k$, where $F_0$ is the set of first-level factors of the decision goal $f_0$, and $F_j$ represents the factors of an internal factor.

Figure 2: Graphical representation for decision process and QUADS framework

allowing the system to learn a better evidence synthesis strategy to achieve higher prediction performance. DPTs have existed in many domains that require formalized decision-making expertise, although they are often referred to as *best practices* or *business rules*. In Section 5 and 6, we will present in detail DPTs converted from the standard principles for disease target validation and in the context of user product recommendation.

## 4. QUADS FRAMEWORK

In this section, we present the proposed QUADS framework for solving the decision making problem defined in Section 3. We first model the decision process by introducing additional auxiliary random variables to represent the internal assertion synthesis process in Section 4.1, where we also establish a connection to the traditional decision support problem outside of the QA context. We then derive algorithms for making decisions and learning decision strategies based on the model in Sections 4.2 and 4.3, respectively.

### 4.1 Modeling Decision Process

The QUADS framework is comprised of two major types of components: QA-based decision factor estimation components and factor synthesis components. We first model them individually, and then obtain the joint probability to represent the overall QUADS framework by multiplying these factors.

**Modeling QA based decision factor estimation.** We here make two assumptions on the variable $x_i$. First, given the fact that the existing external QA pipeline is executed to answer each question individually, we assume $x_i$ and $x_j$ are independent. Second, we assume $x_i$ is conditioned on $y_i$ and its value is determined by a **estimation policy**

$$\xi_i : \mathbf{a} \rightarrow \mathbf{o}_i$$

We make no assumption on the outcome-alternative mapping or QA system accuracy, i.e. $p_i^\xi(x_i = o_{il} | y_i = a_k)$ can be any number between 0 and 1, although a higher QA performance is always preferred. Intuitively, certain assertions often imply a deterministic tendency of prioritizing the corresponding alternatives. For example, an assertion that “the item is light” is an indicator that it tends to be preferred in the decision process, which corresponds to a higher conditional probability $p_i^\xi$.

This assumption allows us to collectively model decision making and answer rectifying in a unified way. In Section 5, we compare the solution proposed in next section with a baseline method which does not consider estimation policy. In Figure 2b, the decision process is augmented with QA estimation variables $x_0$, $x_1$, and $x_2$ accordingly.

**Modeling decision factor synthesis.** Synthesizing factor-level assertions into a comprehensive decision is non-trivial. Intuitively, not only does it need to evaluate the importance of each factor, i.e. the weight or priority in traditional decision analysis, but it is also more important to map combinations of assertions from sub-factors into assertions of their super-factor. In the “phone recommendation" example, the synthesizer needs to understand how to combine two assertions `light` and `Samsung` from two QA processes and how they jointly influence a super-decision in the hierarchy.

We introduce a random variable $d_i$ for each decision factor $f_i$ to represent the decision synthesized from the predecessors and the assertion $x_i$ made from a local QA estimation. The outcomes from $d_i$ are identical to those of $x_i$, i.e. $\mathbf{o}_i$. The value of $d_i$ is determined by a **synthesis policy**

$$\delta_i : \mathbf{a} \times \prod_{k \in F_i} \mathbf{o}_k \rightarrow \mathbf{o}_i$$

We make a simple assumption that it depends on only immediate predecessors, i.e. $F_i$. Analogously, we further create a dependency on $y_i$ from each $d_i$, which captures the intuition that, similar to $x_i$, the synthesized estimation also observes the latent variable but from a more comprehensive perspective. Formally, the synthesizer needs to estimate the probability $p_i^\delta(d_i | y_i, d_{F_i})$ for each $i$.

We combine the decision maker’s preference $p(y_i | y_{F_i})$, the QA based decision factor estimation components $p_i^\xi(x_i | y_i)$ and factor synthesis components $p_i^\delta(d_i | y_i, d_{F_i})$ to obtain a joint probability to represent the overall QUADS framework as follows

$$P(Y, D, X; \Delta, \Xi) = \prod_i p(y_i | y_{F_i}) p_i^\xi(x_i | y_i) p_i^\delta(d_i | y_i, d_{F_i})$$

where $Y$, $D$, $X$, $\Delta$, and $\Xi$ represent $\{y_i\}_i$, $\{d_i\}_i$, $\{x_i\}_i$, $\{\delta_i\}_i$, and $\{\xi_i\}_i$ respectively. A *strategy* refers to the collection of $\Delta \cup \Xi$. The graphical model representation of modeling decision process is shown in Figure 2c for the example used in Figures 2a and 2b. Given a general complex decision, we may use the compact plate notation to represent the nodes and dependencies equivalently as shown in Figure 2d.

**Comparison with other decision analysis formulations.** Similar to traditional multiple criteria decision analysis such as AHP [31], the decision factors are represented in a hierarchy. In contrast to AHP [31], an *influence diagram* (ID) or *decision network* [10], an extension of a Bayesian network of *chance* nodes augmented with *decision nodes*, *utility functions* specifying the preferences of the decision maker, and a precedence ordering, can represent any structured decision problem under uncertainty in a more flexible way. Limited memory influence diagrams (LIMIDs) [17, 21] relax the *regularity* and *no forgetting* assumptions of ID to allow a decision to be conditioned on a limited number of relevant previous observations and decisions. The representation in our work can be considered as a special case of LIMID, where rather than allowing chance nodes and decision nodes to be arbitrarily connected, we explicitly create two hierarchies (or DAGs) individually for analytic thinking ($Y$) and decision synthesis ($D$) and then establish necessary connections between the nodes in the two hierarchies. In the QUADS framework, users do not need to specify an additional utility function to represent their preference besides the priors $Y$.

In Section 4.2, we present how to make optimal decisions in the QUADS framework with a bottom-up dynamic programming if policies $\Delta$ and $\Xi$ are predefined by users, similar to the Viterbi algorithm or the max-sum algorithm [25]. Otherwise, similar to LIMIDs, finding optimal decision rules can be difficult [18]. We leverage an iterative approach similar to *maximizing expected utility* (MEU) problem [14, 18, 26], which is discussed in Section 4.3.

## 4.2 Decision Making

Given policies $\Delta$ and $\Xi$ and QA-produced assertions $X$, decision making corresponds to finding the values of $Y$ and $D$ that maximize the conditional probability, or formally as follows,

$$Y^*, D^* = \underset{Y,D}{\operatorname{argmax}} P(Y, D|X; \Delta, \Xi) \tag{1}$$

Loopy belief propagation algorithms such as max-sum algorithm [25] can be utilized to maximize $Y$ and $D$ on a general Bayesian network with undirected loops. In our case, we make an additional assumption that each decision factor only belongs to a single super-factor, i.e. $F_i \cap F_j = \emptyset$ for any $i$ and $j$. If a factor is included in the same decision process more than once, we will treat them as different factors. Although the Bayesian network still contains undirected loops with the additional assumption, we could find that optimal solution of Eq. 1 can be efficiently obtained with a bottom-up message passing approach[2]. Specifically, we start by calculating a local message vector $u_i$ containing information about the known assertion $x_i$ for each decision factor $f_i$,

$$u_i(y_i) = \log p_i^{\xi}(x_i|y_i) + \log p(y_i) \tag{2}$$

Then we calculate messages $v_i$ and $w_i$ corresponding to $y_i$ and $d_i$ for each primitive factor $f_i$,

$$v_i(y_i) = u_i(y_i) \tag{3}$$

$$w_i(d_i) = \max_{y_i} \log p_i^{\delta}(d_i|y_i, \emptyset) + v_i(y_i) \tag{4}$$

The messages $v_j$ and $w_j$ corresponding to internal decision factor $f_j$ are ready to be calculated once calculation for messages $v_{F_j}$ and $w_{F_v}$ have been done,

$$v_j(y_j) = u_j(y_j) + \max_{y_{F_j}} \log p(y_j|F_j) + \sum_{k \in F_j} v_k(y_k) \tag{5}$$

$$w_j(d_j) = \max_{y_j, d_{F_j}} \log p_j^{\delta}(d_j|y_j, d_{F_j}) + \sum_{k \in F_j} w_k(d_k) + v_j(y_j) \tag{6}$$

Finally, $Y^*$ and $D^*$ correspond to the maximal elements in each message $v_j$ and $w_j$, i.e. $y_j^* = \arg\max_{y_j} v_j(y_j)$ and $d_j^* = \arg\max_{d_j} w_j(d_j)$. We can see from Eqs. 2 to 6 that user preference, assertions made by QA processes and decisions from sub-factors are jointly considered.

## 4.3 Learning Decision Strategies

If the policies $\Delta$ and $\Xi$ are unknown to the decision maker, but past decision processes and values to the random variables are available (all $Y$, and $D$ are known) or partially available (in the extreme case, we only know the final decision $a_k$ for the decision goal $y_0$ of each decision process), then we can learn the policies by solving a similar MEU problem. Many algorithms have been proposed to solve MEU problem, such as the single policy update algorithm [17], EFBP [26], or belief propagation based algorithm [18]. Here based on the efficient message passing algorithm described in Section 4.2, we utilize an iterative belief propagation approach [18].

Specifically, we start by assigning random discrete distributions (e.g. uniform distributions) to all policies in $\Delta$ and $\Xi$, and calculate marginalization messages if some variables are unavailable, which are similar to maximization messages (Eqs. 2 to 6), except that the max-sum calculation is replaced by sum-product calculation. Then, we find a local optimal policy for each $\delta_i \in \Delta$ and $\xi_i \in \Xi$. Since utility function is degenerated, the local MEU problem becomes maximum likelihood estimation, i.e.,

$$p_i^{\xi\,(\text{new})}(x_i = o_{ik}|y_i) = \frac{n(x_i = o_{ik}, y_i)}{\sum_l n(x_i = o_{il}, y_i)} \tag{7}$$

$$p_i^{\delta\,(\text{new})}(d_i = o_{ik}|y_i, d_{F_i}) = \frac{n(d_i = o_{ik}, y_i, d_{F_i})}{\sum_l n(d_i = o_{il}, y_i, d_{F_i})} \tag{8}$$

where $n(\cdot)$ is the number of times all the conditions are satisfied in the training set. We may also smooth $p_i^{\xi}$ and $p_i^{\delta}$ by adding pseudo-counts. Recall that if we have a decision process template, then we assume the policy should exhibit similar behavior across different instances of the template. With the updated $\Delta$ and $\Xi$, if true values of $D$ and $Y$ are partially unknown, they will also be updated according to Eqs. 2 to 6, which completes one iteration. The learning will stop if convergence or a certain stop condition is satisfied.

## 4.4 QUADS Implementation

We implemented an open source software package[3] to support the QUADS solution framework described in this section, which is highly compatible with UIMA and CSE framework [37] to allow users to integrate existing UIMA based QA pipelines and decision synthesis components. In this subsection, we highlight some of the main features of the implementation.

**Pluggable decision factors**. A YAML[4] file is created to represent a single decision factor ($f_i$) including the `question` and additionally a list of `factors` ($F_i$ and $p(f_i|F_i)$) by referencing all YAML files corresponding to the factors and optionally giving preference, unless primitive. We chose decision factors as basic building blocks of the decision process, which allows users to reuse previously created factors or existing decision processes to expand and create similar or more complex decision processes.

[2]In fact, there is always only a single likelihood vector passed from $d_i$ to a factor $d_j$, $j \in F_i$, which implies that the likelihood vectors remain identical for all $d_j \in D$ and therefore also for all $y_j$, which always combines identical messages from the single $y_i$ and a single $d_i$

[3]The source code, DPT files, as well as other examples can be downloaded from `http://oaqa.github.io`.

[4]`http://www.yaml.org`

**Declarative descriptors for templates and decision reports**. The concept of *decision process template* (DPT) introduced in Section 3 is also supported in the QUADS implementation. The list of variables (`vars`) is explicitly specified in each YAML file if the factor is used as part of a template. For example, in the decision factor template that is used in Section 5 to seek genes for targeting diseases, one can specify the `vars` as `[GENE, DISEASE]`. Finally, a YAML-based decision report is generated by the QUADS implementation, which highlights the prioritization of all alternatives and a detailed explanation based on QA pipeline outputs and reasoning process done by the decision synthesis algorithm.

**Configurable QA pipelines and extendable decision synthesis interface**. The two major components in the QUADS implementation (QA pipelines and factor synthesis components) can be configured specifically for each decision factor at each level. In addition, the CSE framework allows multiple QA pipelines or factor synthesis components to be specified as alternatives for the same decision factor. Each configuration combination in the configuration space will be executed separately to enable the decision makers to determine the optimal QA pipeline(s) and synthesis component(s) for each decision factor.

## 5. QUADS FOR TARGET VALIDATION

Developing a new drug is a complex and costly process that proceeds from the identification of a potential therapeutic candidate to marketing a drug product, usually taking more than a decade and costing in excess of 1 billion US dollars [11, 30]. At each phase of the process, decision makers need to consider a series of predetermined criteria about each drug candidate to reduce risk to human subjects and to increase the chance of picking a winning therapeutic molecule [30]. In the earliest stage, *target identification* and *target validation* (or *target assessment*) aim to select and prioritize a number of disease *targets* (agents with a particular biological action that are anticipated to have therapeutic utility), and estimate the "druggability" of each target influenced by a complex balance of scientific, medical and strategic considerations, including efficacy, safety, commercial profits, etc. [15, 11].

In this experiment, we focus on leveraging QUADS to support the target validation problem. The goal is not only to identify targets but also to emphasize a deeper understanding of targets in the full disease context. We implemented a *target validation DPT* to jointly consider multiple scientific and medical criteria, and applied it to support target validation on a dataset of 5.6K human diseases and 6.2K genes. We first describe how the experiment was set up, and then present the experimental results and analysis, which are further followed by a specific case study on breast cancer.

### 5.1 Preparations

In this section, we describe how we implemented the target validation DPT, integrated the biomedical QA system, and how we collected the dataset and prepared the baseline methods.

**Dataset preparation**. We created a Lucene[5] index from a corpus containing more than 22 million PubMED[6] abstracts, with only minimal preprocessing for each document (tokenizing and downcasing the text, and removing common English stopwords). The index was then integrated into the QA system to support evidence gathering. Several databases have been developed to store associations between genes and diseases, each of which focuses on different relational aspects. We used the DisGeNET 2.0 dataset [1], which contains 6,029 diseases and 9,313 genes integrated from a number of gene-disease association databases, including OMIM[7], UniProt[8], PharmGKB[9], and CTD[10].

The DisGeNET dataset was used for QUADS model training and evaluation, and we therefore focus on the manually-created items and exclude the LHGDN subset, which contains gene-disease relations automatically extracted by text mining algorithms. We split the manually created dataset into 5,605 subsets, each of which corresponds to one of the remaining diseases, and 6,158 genes were used as candidates for target validation. We ran 10-fold cross-validation over the dataset, where nine folds were used to learn the policies $\Delta$ and $\Xi$ in the model and one fold was used to test the model. At most 100 iterations were executed for each training run when optimizing the policies.

**Target validation DPT**. Following the DPT guidelines, we created a simple target validation DPT to test our hypothesis, based on literature review as well as personal knowledge and experience from a group of professional biochemists and bioinformaticians working at a pharmaceutical company. We first defined `GENE` and `DISEASE` as the variables in the target validation DPT[11], which could then be instantiated with actual `GENE` and `DISEASE` names from the relations in the DisGeNET dataset. We expanded the template containing questions in the form of Jeopardy! clues in a top-down fashion. Specifically, we started with defining the decision goal, which summarizes the target identification task from a high level: "this GENE is directly involved in DISEASE and can be a suitable target." As one can easily see that necessary detailed criteria for a "druggable" target cannot be thoroughly depicted in the decision goal, the template was then expanded to a second level, which contains six important decision factors such as gene expression, gene mutation, pathway, clinical trials, etc., that was further decomposed as necessary.

For this experiment, we used a target validation DPT with a total of 16 factors, which is shown in Figure 3. We can see from Figure 3 that some strong indicators can hardly be satisfied by a single gene, whereas some other weak factors only contribute to the decision if some gene satisfies all the criteria. We also pay special attention to the criteria 1.6 and 1.6.1, which intuitively are negative indicators. We did not specify any preferences to the decision processes as input, i.e. a uniform prior was used by default, and we see in Section 5.2 that the QUADS framework could automatically learn decision policies taking all these considerations into account.

**Supporting evidence based QA pipeline**. A supporting evidence based QA system such as Watson [7] commonly incorporates parsing, interpretation, retrieval, answer extraction, evidencing and ranking phases. We integrated a few simple biomedical QA components[12], which were designed to answer questions such as those in TREC Genomics tasks [8] or CLEF QA4MRE tasks [27]. We list the key components of the QA pipeline in Table 1. After a question analysis phase which introduces NLP annotations, different queries are constructed from all or part of the extracted focus terms, and other nouns, verbs, and named entities (with synonyms) are also combined via an `OR` operator. A Lucene index was used to retrieve candidate passages based on various queries, and document frequencies for these queries were then combined with weights learned using logistic regression.

[5]`http://lucene.apache.org`

[6]`http://www.ncbi.nlm.nih.gov/pubmed`

[7]Online Mendelian Inheritance in Man Database, `http://omim.org`

[8]Universal Protein Resource Database, `http://www.uniprot.org`

[9]Pharmacogenomics Knowledge Base, `http://www.pharmgkb.org`

[10]Comparative Toxicogenomics Database, `http://ctdbase.org`

[11]Conventionally, a gene can refer to either a gene or a gene product such as protein or mRNA.

[12]The QA pipeline used in this paper is available at `http://oaqa.github.org`.

- 1 the GENE is directly involved in the DISEASE and can be a suitable target.
  - 1.1 Any experiment showing that modulating the activity of the GENE with a chemical compound or genetic modification causes the DISEASE.
    - 1.1.1 Any human in vivo experiment showing that modulating the activity of the GENE affects biochemical function or phenotype of the DISEASE.
    - 1.1.2 Any in vitro experiment showing that modulating the activity of the GENE affects biochemical function or phenotype of the DISEASE.
    - 1.1.3 Any animal model study showing that modulating the activity of the GENE in animals causes the DISEASE.
  - 1.2 the GENE is expressed in the human tissue related to the DISEASE.
    - 1.2.1. the GENE is expressed in normal human tissue related the the DISEASE.
    - 1.2.2 the GENE expression is altered in human DISEASE tissue or human DISEASE cell.
    - 1.2.3 The alteration of the GENE expression is correlated with the DISEASE severity.
  - 1.3 Any mutation is associated with the DISEASE.
    - 1.3.1 Any mutation of the GENE has significantly associated with the DISEASE.
    - 1.3.2 Any mutations in other genes linked to the GENE associated with DISEASE.
  - 1.4 Any pathway involving the GENE supports that the GENE causes the DISEASE.
  - 1.5 Any clinical trials show that targeting the GENE can prevent or slow the progress of the DISEASE.
  - 1.6 Any evidence suggests that targeting the GENE will have side effects.
    - 1.6.1 Targeting the GENE will cause liver, heart, and kidney damage.

Figure 3: Questions in the three-level target validation DPT

Table 1: Summary of integrated components for QA pipeline

| Category | Components |
|---|---|
| Parsing | LingPipe HMM based tokenizer<br>LingPipe HMM based POS tagger<br>LingPipe HMM based named entity recognizer<br>Rule based lexical variant generator |
| Interpretation | UMLS for disease syn/acronym expansion<br>EntrezGene for gene syn/acronym expansion<br>OMIM for disease syn/acronym expansion |
| Retrieval | Lucene search engine |
| Evidencing and ranking | Document frequency (DF)<br>Weight learning with logistic regresion |

Since the types of questions in the DPT are all yes/no questions, either `y` or `n` will be assigned to label each assertion at the end of the pipeline. Compared with arbitrary nominal assertions, yes/no assertions can generalize the learned decision model[13]. We can see that the chosen QA pipeline lacks the algorithmic sophistication (i.e. inference capability) required to automatically interpret and match the term "suitable target" by filling in necessary information related to the specific factors.

**Evaluation methods.** Although the "druggability" of genes is sometimes subjective and difficult to measure until later phases are conducted in the drug development process, this experiment, as the first attempt to support target validation, demonstrates how to rediscover known targets for the diseases purely and automatically from the literature by considering various decision factors involved in the decision process. In particular, for each subset corresponding to a particular disease, QUADS evaluates each gene in the list and finally decides an ordered list of genes prioritized by the global confidence $p(d_0 = \mathtt{y})$ for each candidate (or equivalently $w_0(d_0 = \mathtt{y}) - w_0(d_0 = \mathtt{n})$), which were compared against the unordered known target list. We then focus on measuring precision, recall, and F-1 averaged over all subsets when the order is ignored, similar to list question answering evaluation [5], as well as mean average precision (MAP) for retrieval tasks when the order is considered. Detailed performance evaluation at each factor level is unavailable for the entire dataset, which is actually one of the disadvantages of existing databases that only give an incomprehensive view of the problem. In Section 5.3, we manually labeled a number of genes from the top of the returned list at the factor level to support deeper analysis.

**Baseline methods**. To motivate the QUADS framework, we compare it with several baseline methods. We first simplified the target validation DPT by leaving only the first level and the first two levels (denoted by **1 Level** and **2 Levels**) to demonstrate how additional decision factors can impact the decision results, and in addition, we flattened the template hierarchy but still kept all the factors to construct an unstructured decision process (denoted by **Flattened**) similar to a scenario question [3], which is used to motivate the importance of analytic thinking (decision synthesis) in decision making. We also compare with two different decision synthesis methods. One is an unsupervised method that simply synthesizes the decision by majority voting from all factors, which is denoted by **Voting**. The other baseline method simplifies the proposed QUADS solution by removing the extra latent variable layer $Y$ and thus $\Xi$, i.e. policies $\Delta$ directly map $d_{F_i}$ and $x_i$ to $d_i$ (denoted by **Simplified**). The learning process is then degenerated to a linear programming problem and no inference is required. In addition, we also simplified the QA pipeline by removing the synonym expansion components (denoted by **No Synonym**), which have been shown to play a crucial role in biomedical information retrieval tasks in previous research [8].

## 5.2 Experimental Results & Analysis

We present the performance results in terms of Precision, Recall, F-1, and MAP averaged across all 10 test folds in Table 2, with the significance test (t-test) results of each baseline method compared against the proposed QUADS framework within each run; scores are labeled with different significance levels († for $p < 0.05$ and ‡ for $p < 0.005$, and no difference was observed in significance levels across runs). We can see that the proposed QUADS solution framework significantly outperformed all baseline methods in all the metrics. In general, MAP scores are higher than precision scores, which indicates the most extensively studied genes contained in the DisGeNET dataset can be successfully retrieved and prioritized at the top of the list, however some other genes that QUADS and other baseline methods also consider likely to be target candidates are missing in the DisGeNET.

Compared with QUADS, **1 Level** and **2 Levels** achieved worse recall and could not properly prioritize the candidates, due to missing important factors that relate to the target validation goal. The only information that was utilized in prioritization in **1 Level** is the QA evidence score, which was unfortunately dominated by

[13]One can also consider different levels of confidence as assertions, e.g. `certain`, `probable`, `probably not`, `impossible`, etc.

Table 2: Experimental result for target validation decision support

| | Precision | Recall | F-1 | MAP |
|---|---|---|---|---|
| QUADS | **0.7834** | **0.8983** | **0.8369** | **0.8724** |
| 1 Level | $0.6885^{\ddagger}$ | $0.7334^{\ddagger}$ | $0.7102^{\ddagger}$ | $0.7721^{\ddagger}$ |
| 2 Levels | $0.7397^{\ddagger}$ | $0.8329^{\ddagger}$ | $0.7835^{\ddagger}$ | $0.8032^{\ddagger}$ |
| Flattened | $0.7423^{\ddagger}$ | $0.8610^{\dagger}$ | $0.7973^{\ddagger}$ | $0.8186^{\ddagger}$ |
| Voting | $0.7486^{\ddagger}$ | $0.8479^{\ddagger}$ | $0.7952^{\ddagger}$ | $0.8126^{\ddagger}$ |
| Simplified | $0.7743^{\dagger}$ | $0.8754^{\dagger}$ | $0.8217^{\dagger}$ | $0.8501^{\ddagger}$ |
| No Synonym | $0.7014^{\ddagger}$ | $0.5479^{\ddagger}$ | $0.6152^{\ddagger}$ | $0.7135^{\ddagger}$ |

Table 3: Learned estimation policy $p^{\xi}$

$p_i^{\xi}(x_i = \mathtt{y} | y_i = \mathtt{y})$

| $f$ | 1 | 1.1 | 1.1.1 | 1.1.2 | 1.1.3 | 1.2 | 1.2.1 | 1.2.2 |
|---|---|---|---|---|---|---|---|---|
| $p$ | .4743 | .5005 | .6577 | .6791 | .4699 | .8864 | .8895 | .8131 |
| $f$ | 1.2.3 | 1.3 | 1.3.1 | 1.3.2 | 1.4 | 1.5 | 1.6 | 1.6.1 |
| $p$ | .5442 | .8840 | .8873 | .9044 | .9107 | .5497 | .0118 | .0088 |

$p_i^{\xi}(x_i = \mathtt{n} | y_i = \mathtt{n})$

| $f$ | 1 | 1.1 | 1.1.1 | 1.1.2 | 1.1.3 | 1.2 | 1.2.1 | 1.2.2 |
|---|---|---|---|---|---|---|---|---|
| $p$ | .9736 | .9999 | .9998 | .9996 | .9998 | .9987 | .9986 | .9994 |
| $f$ | 1.2.3 | 1.3 | 1.3.1 | 1.3.2 | 1.4 | 1.5 | 1.6 | 1.6.1 |
| $p$ | .9999 | .9988 | .9989 | .9987 | .9984 | .9998 | .1767 | .2875 |

the popularity or history of research for each gene, since the QA pipeline couldn't capture any information about the original decision goal beyond the two named entities `GENE` and `DISEASE`.

**Flattened** could achieve a higher precision and recall than **1 Level** and **2 Levels**, but the MAP is unsatisfactory compared with QUADS. Since most biomedical QA systems are not tuned to answer scenario questions that involve complex discourse structures, this isn't surprising. Therefore, even when all necessary pieces of information are retrieved by the QA pipeline, without QUADS, a single QA pipeline can hardly assign a correct weight to each factor, nor can it properly merge evidence like any decision policy $\delta$ which can explicitly represent various combination strategies including `AND`, `OR`, `NOT`, etc.

**Voting** performed very similarly to **Flattened**, due to the similar setting that factors were undistinguished when the decision was made at $d_0$. The only difference is that in **Voting** each factor is associated with a weight, which is determined by its position in the hierarchy. Specifically, the deeper it is in the hierarchy or the more sibling factors it has, the lower it is weighted.

Among the baseline methods, the performance of **Simplified** was closest to that of QUADS in all metrics, which first motivates the importance of decision processes when complex information needs are required, and second, suggests that modeling the generation of QA assertions as a probabilistic event helps to make QUADS more sensitive to the different levels of uncertainty inside each QA process. In fact, if we look at the final trained model, we could see in Table 3 that most $\xi$ can learn a stochastic matrix being nearly an identity matrix, indicating a relatively trustworthy QA process with exceptions regarding factors 1.6 and 1.6.1.

Finally, we see that **No Synonym** unsurprisingly did the worst job in identifying the correct targets, which again suggests that synonym expansion is necessary for effective biomedical information retrieval in this context.

To further satisfy our curiosity, we sorted the input combinations of the decision policies $\Delta$ in descending order of the probability of making a $\mathtt{y}$ decision, i.e. $p^{\delta}(d = \mathtt{y}|y, d_F)$, which is partially shown in Table 4. We observe that among the top input combinations for all decision policies, the final decision $f_1$ is obviously more difficult than others, and the local QA estimation $y_1$ can positively impact the final decision, whereas most other decision policies tend to ignore the local estimation, e.g. $y_{1.1}$ and $y_{1.3}$.

Table 4: Top-ranked decision policies $p^{\delta}(d = \mathtt{y}|y, d_F)$

| $f$ | Rk | $y, d_F$ | $p^{\delta}$ | Rk | $y, d_F$ | $p^{\delta}$ |
|---|---|---|---|---|---|---|
| 1.1 | 1 | $y_{1.1} = \mathtt{y}, d_{1.1.*} = \mathtt{y}$ | .9984 | 2 | $y_{1.1} = \mathtt{n}, d_{1.1.*} = \mathtt{y}$ | .9561 |
| 1.3 | 1 | $y_{1.3} = \mathtt{y}, d_{1.3.*} = \mathtt{y}$ | .9957 | 2 | $y_{1.3} = \mathtt{n}, d_{1.3.*} = \mathtt{y}$ | .9431 |
| 1 | 1 | $y_1 = \mathtt{y}, d_{1.*} = \mathtt{y}$ | .9710 | 2 | $y_1 = \mathtt{n}, d_{1.*} = \mathtt{y}$ | .5015 |

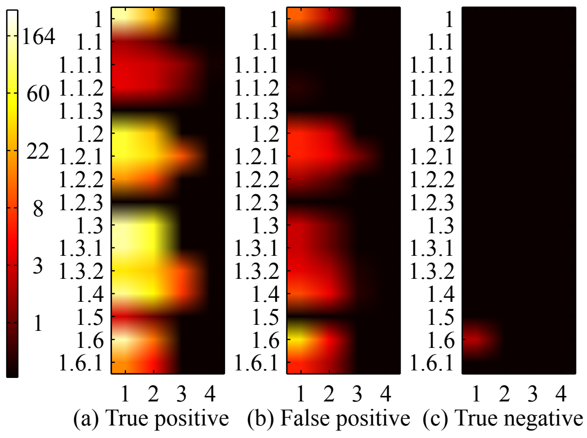


Figure 4: Evidence collected for all factors (y-axis) in terms of averaged document frequency for the subsets corresponding to true positive (a), false positive (b) and true negative. Queries of different complexity (x-axis) in terms of number of query terms were issued.

## 5.3 Case Study: Breast Cancer

We used breast cancer as a case study to demonstrate how QUADS can help understand different ways of finding support for disease targets, focusing on the factor estimation part. We discovered that the system can achieve a recall of 100% for this particular disease, and then we selected and manually checked the correctness of nine genes that were ranked at top of the list to understand the loss of precision, which are `AKT1`, `BRCA1`, `BRCA2`, `BRIP1`, `ERBB2`, `FGFR1`, `KDR1`, `mTOR`, and `NBN`. We found that all the nine genes are known to be good targets, and three genes among them (`FGFR1`, `KDR1`, `mTOR`) were not included in the DisGeNET dataset.

Evidence scores collected from QA processes that relate to the factors for the genes are compared in a heat map in Figure 4, and are further grouped by (a) true positive, (b) false positive, and (c) true negative. The x-axis corresponds to the number of additional `OR`-combined query terms to represent queries of different complexity, and the value in the heatmap corresponds to the document frequency; these were used as features for learning the assertion generation. The Y-axis corresponds to the factors in Figure 3.

First, we can clearly see that more evidence was gathered for the true positive subset than the false positive and true negative subsets. Second, "answerability" of questions differs across factors. Some questions with narrow but clear concept, e.g. "human in vivo experiments", "expression alteration", "in disease tissues or cells" can be answered properly for all genes, were hence trusted by corresponding policies. A difficult-to-answer question can lead to low evidence, e.g. factors 1.1.3 and 1.2.3, 1.5, which correspond to relatively lower weights in Table 3. In order to maximize the benefit from these two observations, we may also consider the real-valued confidence score, in addition to the yes/no answer returned from the QA pipelines, in order to improve the decision synthesis process.

## 6. QUADS FOR PRODUCT RECOMMENDATION

In this section, we consider a common decision process faced by on-line consumers: *product recommendation from review texts*. Unlike the target validation problem, algorithmic methods for the product recommendation problem (as well as the related rating problem) have been extensively investigated [22]. In this section, rather than propose any new algorithm for this relatively well-studied problem, we adapt the QUADS framework (built originally for target validation) to product recommendation with minimal change, in order to test its flexibility and generalizability.

### 6.1 Preparations

We first describe the experimental settings and focus on what changes have been done to enable the adaptation of QUADS to the problem of product recommendation from review texts.

**Dataset preparation.** We employed the `Cell Phones & Accessories` subset of an Amazon product review corpus [22][14], which contains 78,931 reviews for 7,438 distinct products. Similar to the preparation procedure for PubMED abstract corpus, we here focused on the `review/text` field in each review and tagged with `productId` and ignored other fields such as `userId` or `time` (although they provide supplementary information for the recommendation task, incorporating structured data with a different schema often requires extra effort to adapt existing QA pipelines).

Since this task is more subjective than target validation, we can hardly rely on a manually-created gold-standard product recommendation list. Thus we use the average rating of each product as its overall recommendation weight, which is then converted from a 5-point rating scale to a binary recommendation. Specifically, ratings equal to or smaller than 3 are assigned `no`, otherwise `yes`, and then we obtained a list of 4,583 "recommended" products. We conducted a 10-fold cross validation experiment on 7,439 items. Various product reviews give insights about the same product from different perspectives, in the same way that different publications in PubMED discuss different perspectives of the same gene/disease.

**DPT for buying a cell phone.** Unlike the target validation task, where a few second-level aspects can cover most considerations for a majority of diseases, different products have different decision factors, which motivated us to focus on a particular product type (cell phones). We allowed QA pipelines to return nominal answer texts (e.g. `Apple`, `Sumsung`, `Nokia`, `Google`, etc. from a finite set of `brands`). Compared with the number of cell phone products and reviews available in the data set, this restriction did not cause any issues with data sparsity.

Most of the adaptation effort was spent in creating a DPT for cell phone recommendation. First, we defined the decision goal as "buying this PHONE is recommended" where `PHONE` is the variable in the template. Then, we considered the most important high-level features, including `design and usability`, `brand`, `functionality`, `carrier`, and `operating system`, and we created natural language questions corresponding to each feature, e.g. "this PHONE has good usability", or "this PHONE is made by Apple", etc. For broad questions such as the one asking about `usability`, we further decomposed the question down to two sub-factors: "this PHONE is light" and "this PHONE has a good interface design". Finally, we created a DPT with 17 decision factors[15].

Although many of the questions about each carrier or brand can preferably be answered by searching structured product information databases or parsing a semi-structured product specification page instead of raw review texts, in this experiment we focused only on unstructured texts (which could be searched after much simpler domain adaptation).

[14]`http://snap.stanford.edu/data/web-Amazon-links.html`

[15]The DPT and the QA pipeline can be found at `http://oaqa.github.org`.

Table 5: Experimental results for phone recommendation

| | Precision | Recall | F-1 | MAP |
|---|---|---|---|---|
| QUADS | **0.6433** | **0.7342** | **0.6858** | **0.7943** |
| 1 Level | $0.4328^{\ddagger}$ | $0.5632^{\ddagger}$ | $0.4895^{\ddagger}$ | $0.5496^{\ddagger}$ |
| 2 Levels | $0.5849^{\ddagger}$ | $0.6734^{\ddagger}$ | $0.6260^{\ddagger}$ | $0.6734^{\ddagger}$ |
| Flattened | $0.5993^{\ddagger}$ | $0.7046^{\ddagger}$ | $0.6477^{\ddagger}$ | $0.6842^{\ddagger}$ |
| Simplified | $0.6340^{\dagger}$ | $0.7197^{\dagger}$ | $0.6741^{\dagger}$ | $0.7795^{\dagger}$ |
| No Synonym | $0.3219^{\ddagger}$ | $0.4479^{\ddagger}$ | $0.3746^{\ddagger}$ | $0.4608^{\ddagger}$ |

**QA pipeline.** We applied almost the exact same set of QA components for this task. However, some changes were required. First, we replaced the POS tagging model and NER model to models trained on standard English news corpora. Second, we replaced the biomedical synonym expansion component with a sentiment word dictionary[16]. We also integrated the index for the reviews to replace the biomedical corpus. The rest of the pipeline remains the same.

The evaluation methods and baseline methods are also identical to those used for the target validation task, except that the **Unsupervised** condition is unavailable (without careful tuning, it is difficult to combine different nominal assertions from different factors).

### 6.2 Experimental Results & Analysis

In Table 5, we list the evaluation results for the phone recommendation problem. The scores are generally lower than those in the target validation evaluation, which surprisingly indicates that an everyday decision problem that seems easy is actually harder than a scientific decision based on technical publications. Nevertheless, the result still supports the original motivation for QUADS and illustrates the relative effectiveness of the proposed QUADS framework compared to baseline methods.

In fact, the two decision problems face different types of information overload. The biggest challenge in the target validation problem is to navigate through millions of publications to select the most relevant ones. It is always the case that, among the thousands of candidate genes, only a few of them have been studied extensively and documented in relevant papers; once the relevant documents are identified, the evidence is relatively easy to estimate. In product recommendation from review texts problem, two new challenges should be solved: informal texts and mixed reviews. There are many texts about each product, but they use widely varying language (degrees of formality/informality) and often mix positive / negative evidence within the same review.

## 7. CONCLUSION & FUTURE WORK

In this paper, we first introduced a novel decision representation which allows decision makers to define a decision-making best practice with *DPT*, or to structure an *ad hoc* decision process by formulating and organizing natural language questions or assertions in an analytic hierarchy. We then introduced a new decision support framework, QUADS, which takes advantage of question answering technologies to automatically understand the decision situation and make a decision in real time, using Bayesian learning and inference processes. An open source framework implementation was used in two real-world applications: *target validation*, one of the most fundamental tasks for the pharmaceutical industry, and *product recommendation from reviews*, an important decision process for on-line consumers. We implemented and compared a

[16]`http://www.cs.uic.edu/~liub/FBS/opinion-lexicon-English.rar`

number of baseline methods that partially accomplish some of the QUADS functions, and show that the QUADS framework outperformed these baseline methods for both datasets.

QUADS demonstrates a new way to provide structured decision support from unstructured text by incorporating QA technology, but many important (and interesting) research questions remain to be addressed. First, the QA pipelines used in the experiments are far from perfect, and can be improved by adding evidence from structured knowledge bases as another decision sub-factor in addition to unstructured text. On the other hand, since the estimation policy reveals the trustworthiness or comformity of each factor, we can also identify the hardest questions for the QA components. A joint optimization of question answering and decision making may be a good way to approach this combined problem in new domains. Another question is how to discover the hidden factors that influence a decision, in order to enable automatic decision process construction. Beyond existing *facet extraction* methods, one can also integrate websites that have already collected and organized "how-to" manuals and allow community content development, such as wikiHow. Moreover, the current framework requires a static decision process specified at the beginning of the system's execution, which can be effective for a decision problem with tens of factors and thousands of alternatives like the two applications we investigated; nevertheless, there are limits to how far one can scale the manual creation of decision process descriptions. An interesting idea for future research is to allow the decision process to emerge dynamically, with selective pruning of alternatives based on the partial evidence that has been aggregated thus far.

**Acknowledgement.** The authors would like to thank the Roche BioQA project team for their helpful discussion and suggestions.

## 8. REFERENCES

[1] A. Bauer-Mehren, M. Rautschka, F. Sanz, and L. I. Furlong. Disgenet: a cytoscape plugin to visualize, integrate, search and analyze gene–disease networks. *Bioinformatics*, 26(22):2924–2926, 2010.

[2] F. Bex, J. Lawrence, M. Snaith, and C. Reed. Implementing the argument web. *Commun. ACM*, 56(10):66–73, Oct. 2013.

[3] M. W. Bilotti and E. Nyberg. Evaluation for scenario question answering systems. In *Proceedings of LREC '06*, pages 1536–1541, 2006.

[4] S. Chaudhuri, U. Dayal, and V. Narasayya. An overview of business intelligence technology. *Commun. ACM*, 54(8):88–98, Aug. 2011.

[5] H. T. Dang, D. Kelly, and J. J. Lin. Overview of the trec 2007 question answering track. In *Proceedings of TREC '07*, 2007.

[6] W. Edwards, R. F. M. Jr., and e. Detlof von Winterfeldt. *Advances in Decision Analysis*. Cambridge University Press, 2007.

[7] D. Ferrucci, E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. Prager, et al. Building watson: An overview of the deepqa project. *AI magazine*, 31(3):59–79, 2010.

[8] W. Hersh and E. Voorhees. Trec genomics special issue overview. *Inf. Retr.*, 12(1):1–15, Feb. 2009.

[9] R. A. Howard. Decision analysis: Applied decision theory. In *Proceedings of ICORES '66*, pages 304–328, 1966.

[10] R. A. Howard and J. E. Matheson. Influence diagrams. In *Readings on the Principles and Applications of Decision Analysis*, volume 2. Sdg Decision Systems, 1981.

[11] J. Hughes, S. Rees, S. Kalindjian, and K. Philpott. Principles of early drug discovery. *Br. J. Clin. Pharmacol.*, 162(6):1239–1249, 2011.

[12] A. Kalyanpur, S. Patwardhan, B. Boguraev, A. Lally, and J. Chu-Carroll. Fact-based question decomposition for candidate answer re-ranking. In *Proceedings of CIKM '11*, pages 2045–2048, 2011.

[13] B. Katz, G. Borchardt, and S. Felshin. Syntactic and semantic decomposition strategies for question answering from multiple sources. In *Proceedings of AAAI '05 Workshop on Inference for Textual Question Answering*, pages 35–41, 2005.

[14] U. B. Kjrulff and A. L. Madsen. *Bayesian Networks and Influence Diagrams: A Guide to Construction and Analysis*. Springer Publishing Company, Incorporated, 2nd edition, 2012.

[15] J. Knowles and G. Gromo. Target selection in drug discovery. *Nature Reviews Drug Discovery*, 2(1):63–69, 2003.

[16] F. Lacatusu, A. Hickl, and A. Harabagiu. Impact of question decomposition on the quality of answer summaries. In *Proceedings of LREC '06*, pages 1147–1153, 2006.

[17] S. L. Lauritzen and D. Nilsson. Representing and solving decision problems with limited information. *Manage. Sci.*, 47(9):1235–1251, Sept. 2001.

[18] Q. Liu and A. T. Ihler. Belief propagation for structured decision making. In *UAI '12*, pages 523–532, 2012.

[19] J. D. Lowrance, I. W. Harrison, and A. C. Rodriguez. Capturing analytic thought. In *Proceedings of K-CAP '01*, pages 84–91, 2001.

[20] J. L. Malin. Envisioning watson as a rapid-learning system for oncology. *Journal of Oncology Practice*, 9(3):155–157, 2013.

[21] D. D. Mauá, C. P. de Campos, and M. Zaffalon. Solving limited memory influence diagrams. *J. Artif. Int. Res.*, 44(1):97–140, May 2012.

[22] J. McAuley and J. Leskovec. Hidden factors and hidden topics: Understanding rating dimensions with review text. In *Proceedings of RecSys '13*, pages 165–172, 2013.

[23] J. R. Miller. *Professional Decision-Making: a procedure for evaluating complex alternatives*. Praeger Publishers, 1970.

[24] J. W. Murdock, J. Fan, A. Lally, H. Shima, and B. Boguraev. Textual evidence gathering and analysis. *IBM Journal of Research and Development*, 56(3.4):8:1–8:14, 2012.

[25] K. P. Murphy, Y. Weiss, and M. I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proceedings of UAI '99*, pages 467–475, 1999.

[26] A. Nath and P. Domingos. Efficient belief propagation for utility maximization and repeated inference. In *Proceedings of AAAI '10*, pages 1187–1192, 2010.

[27] A. Peñas, E. Hovy, P. Forner, A. Rodrigo, R. Sutcliffe, and R. Morante. Qa4mre 2011-2013: Overview of question answering for machine reading evaluation. In *Information Access Evaluation. Multilinguality, Multimodality, and Visualization*, volume 8138, pages 303–320. Springer Berlin Heidelberg, 2013.

[28] A. Pollock and A. Hockley. What's wrong with internet searching. In *Designing for the Web: Empirical Studies*, 1996.

[29] J. Prager, J. Chu-Carroll, and K. Czuba. Question answering using constraint satisfaction: Qa-by-dossier-with-constraints. In *Proceedings of ACL '04*, 2004.

[30] J. F. Pritchard, M. Jurima-Romet, M. L. Reimer, E. Mortimer, B. Rolfe, and M. N. Cayen. Making better drugs: Decision gates in non-clinical drug development. *Nature Reviews Drug Discovery*, 2(7):542–553, 2003.

[31] T. L. Saaty. *What is the analytic hierarchy process?* Springer, 1988.

[32] E. Saquete, P. Martínez-Barco, R. Muñoz, and J. L. Vicedo. Splitting complex temporal questions for question answering systems. In *Proceedings of ACL '04*, 2004.

[33] J. Sprague, Ralph H. A framework for the development of decision support systems. *MIS Quarterly*, 4(4):pp. 1–26, 1980.

[34] D. von Winterfeldt. Bridging the gap between science and decision making. *Proceedings of the National Academy of Sciences*, 110(Supplement 3):14055–14061, 2013.

[35] D.-L. Xu, McCarthy, and J.-B. Yang. Intelligent decision system and its application in business innovation self assessment. *Decision Support Systems*, 42(2):664–673, 2006.

[36] F. Yang, J. Feng, and G. Di Fabbrizio. A data driven approach to relevancy recognition for contextual question answering. In *Proceedings of HLT-NAACL '06 Workshop on Interactive Question Answering Workshop (IQA)*, pages 33–40, 2006.

[37] Z. Yang, E. Garduno, Y. Fang, A. Maiberg, C. McCormack, and E. Nyberg. Building optimal information systems automatically: Configuration space exploration for biomedical information systems. In *Proceedings of CIKM '13*, 2013.