Using Rich Inference to Find Novel Answers to Questions

Using Rich Inference to Find Novel Answers to Questions

Kwabena Nuamah
School of Informatics
University of Edinburgh
Edinburgh, United Kingdom
k.nuamah@ed.ac.uk

Alan Bundy
School of Informatics
University of Edinburgh
Edinburgh, United Kingdom
bundy@inf.ed.ac.uk

Christopher Lucas
School of Informatics
University of Edinburgh
Edinburgh, United Kingdom
c.lucas@ed.ac.uk

Abstract*— The Web is continuously enriched with data and has become a large knowledge repository. However, machines are unable to fully exploit this vast knowledge space in performing reasoning tasks such as question answering. This inability limits the extent of inference and ultimately limits the range of questions they can answer. We argue that the quality and range of answers generated by a question-answering system is significantly improved when we use rich reasoning techniques to infer novel knowledge from web data. By finding and aggregating facts from different knowledge bases, an agent can obtain a better representation of a domain and hence infer new facts which did not exist in any of the original knowledge sources. We intend to explore rich semantic representations and rich forms of reasoning. These include the curation of data and the use of a combination of heuristics, logic and probabilistic techniques to infer answers. This approach will minimize noise and uncertainty in the knowledge for reasoning. Our customized representations will suit the problem to be solved rather than being restricted by the formalisms used in the sources. We plan to implement this in a question-answering system that exploits a vast set of knowledge bases such as ontologies and Linked Data repositories. Our question-answering system will focus on questions which require rich inferences such as prediction and composition of answers from several pieces of information.

Keywords*— inference, question-answering, knowledge representation, heuristics, uncertainty

I. INTRODUCTION

The increasing availability of knowledge bases, such as ontologies on the web, has opened up the possibility of computer agents taking advantage of the massive amounts of information on the web for reasoning and information retrieval tasks that were previously intractable. Logical inference can enable an agent to infer implicit relationships between concepts in the knowledge base, provided appropriate techniques are employed to deal with ambiguous, incomplete and sometimes erroneous data.

When given a question, humans possess the ability to choose from a gamut of possible strategies the one that best solves the question. This ability allows us to answer questions even when the answer is not pre-stored in our memory or knowledge base. In contrast, question answering (QA) systems, although originally designed to use inference, tend to assume that the answer is pre-stored in a knowledge base. Consider the question "*What will be the UK population in 2021*?". A QA system will typically attempt (unsuccessfully) to find the pre-stored fact *population(UK, 2021, p)*. It most likely will not find this, and so will give up and return no answer. Wolfram|Alpha [1] highlights this point in a statement on its website: "Only what is known is known to Wolfram|Alpha". In contrast, humans are able to answer this kind of question, by indirectly inferring answers that we do not already have, from other readily available information. In the example above, we could look up the population values for past years, and then estimate the population in 2021 using regression. We could also find the population growth rate from Wikipedia and use that to predict the population in 2021. In so doing, we use a combination of heuristics, logic and probabilistic techniques to infer answers. We refer to this as *rich inference*.

We believe that rich inference, applied to the heterogeneous and ever-growing sources of information on the web, is critical to realizing the promise of automated question-answering.

More specifically, we claim that the quality and range of answers generated by a question-answering system is significantly improved when we automatically curate data and use richer forms of inference to infer novel knowledge from Semantic Web data [2]. This improvement can be achieved by finding and aggregating facts from different knowledge bases, obtaining a better representation of the domain, discovering and caching new facts that are not already stored in any of the original knowledge sources. The rich inference that supports the project includes heuristics for decomposing questions, logical and probabilistic reasoning and higher-order functions which we use to aggregate data into answers to questions, not limited by the formalisms of the source data.

In practical terms, we intend to build a system that can respond to questions where no suitable answer is contained in any available data source, e.g., as a Resource Description Framework (RDF) triple [3], stored phrase, or table entry. This requires rich inference applied to pre-stored facts, logical relationships, e.g., web ontology language (OWL) [4] and description logics [5], and other formal semantics. Further, any novel facts or relationships that have been inferred can then be propagated back to customized knowledge-bases, facilitating future question-answering. Unlike current question answering systems which focus on the natural language processing (NLP) problems inherent in QA, our core contribution will emphasize mapping machine-readable queries to answers. Although natural

language processing is not our main focus, we will use third-party tools to map natural-language questions to representations that our system can use.

II. Inference in Existing QA Models

To varying extents, recent QA systems apply different forms of question transformations, decompositions, rules and inference techniques to get answers to questions. We classify these into three models. Fig. 1 shows the three main types of models that current QA systems use.

Model 1 is the simplest type, characterized by avoiding any transformations of the representation of the question. Model 1 systems query knowledge bases directly, with the hope that the data that best answers the questions are immediately available. This model is often restricted to a specific domain using curated knowledge bases, and a query language with a restricted vocabulary. This is found in simple QA systems which place a user interface over the knowledge base and then find answers that best match the user query. Most basic information retrieval system and database systems such as SQL (Structured Query Language) follow this model.

Model 2 adds a question transformation feature. The objective is to transform the question so that it exploits the knowledge representation formalism used in the knowledge base. This allows the QA system to work with knowledge bases whose formalisms are known. These transformation rules are usually fixed and specific to the knowledge bases that the QA system depends on. AskMSR[6] uses this technique to reformulate questions. Because its core strategy is to leverage search engines, the reformulation of questions allows it to rewrite the same query in different ways, and then submit each query to a search engine. AskMSR exploits the redundancy in web data by collecting summaries of the search results, mining and filtering N-grams, and determining the best answers from the remaining data. Initial versions of START [7] similarly approach QA by transforming questions into templates which it uses to search its knowledge base. It uses its rule-based approach (*S-rules*) and its *natural language annotations* to find the matches in its knowledge base to the question and then returns an answer from the best matches.

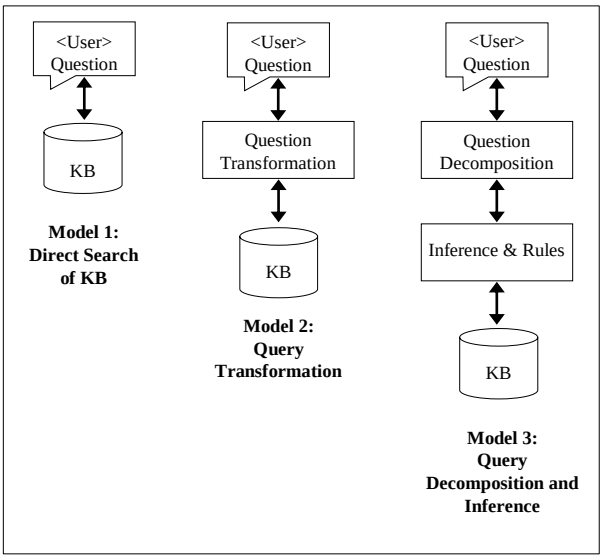

Fig. 1. QA Models

Model 3 does not just transform the question into some specific representation, but also decomposes it by some criteria. For instance, the IBM Watson system uses parallel decomposition [8] when questions contain mutually independent facts about the answer. An example used by the authors was *"[Which] company with origins dating back to 1876 became the first U.S. company to have 1 million stockholders in 1951?*". In this question, knowing the *company with origins dating back to 1876* is important, but not necessary to determining *the first U.S. company to have 1 million stockholders in 1591*. So both can be determined independently and a common answer that both sub-questions find, will most likely be the answer to the whole question. IBM Watson also uses nested decomposition for questions containing an independent fact about an entity related to the correct answer and a separate fact that links that entity to the correct answer. An example of this type of question is "*Which surgical procedure is required to deal with an aortic condition associated with bicuspid aortic valves*?". In this question, it is necessary to first determine the *aortic condition associated with bicuspid aortic valves* before the *surgical procedure required to deal with it* is found.

Saquete et.al [9] also device a temporal decomposition strategy in their QA system which was designed to answer questions of a temporal nature. The question: "*Where did Bill Clinton study before going to Oxford University*?" is decomposed into the questions "*Where did Bill Clinton study*" and "*When did Bill Clinton go to Oxford University*". Model 3 approaches also perform some degree of inference from knowledge bases. IBM Watson does this by taxonomic reasoning to check whether the candidate answer type it generates matches the proper lexical answer type. It also uses some form of inference to reason over semantic web and linked data resources (such as DBpedia [10], YAGO [11] and Cyc [12]) it uses as part of its knowledge base.

Another system using a Model 3 is GORT [13]. In the GORT project, the Semantic Web was used to "guesstimate" answers to user questions by combining disparate facts to provide approximate answers to numeric questions. However, GORT is limited to specific types of proof-trees, and basic functions such *as Min, Max, Count and Average* from which it can compute to infer new facts. It also requires user inputs to fill in answers to some sub-questions. PowerAqua [14] and ANGIE [15] also use semantic web data to answer questions. PowerAqua uses its *triple mapping component* (triple similarity service) to find suitable answers to questions which have been decomposed and transformed into query triples. A later version of START [16] uses syntactic and semantic decompositions to help answer questions using multiple web data sources. It is limited in its use of formal inference and rather uses rules which match specific patterns in the questions to connect domain questions to sub-questions that are answerable by specific knowledge resources.

Earlier QA system also used different formal methods of inference and deduction to answer questions. QA3 [17] and DEDUCOM [18] are examples of such systems. However, an analysis of these early QA systems by Simmons in [19] showed that several of these system use inference rules to expand and transform the formal expression of the question until it matches some combination of facts in its knowledge base.

In general, the extent of inference applied in Model 3 is only to the point of directly retrieving fact(s) from the knowledge base and using that as the answer, as part of the answer or as input into a placeholder in a nested question.

III. RICH INFERENCE

The crucial difference between these past approaches and our proposal is the idea of *rich inference*, to which we have already alluded. Here we describe the characteristics that distinguish this idea from the kinds of inference in Model 3 systems, and outline a plan for implementing it. First, rich inference involves reasoning about higher-order facts, such as functional relations in the data, which expands the range of answers that can be sought. Second, rich inference involves dynamic curation and re-use of previously inferred facts and relations, which makes it possible to find answers to increasingly difficult questions and abstract questions. Third, rich inference must be robust in the face of uncertainty, and must make is possible to express degrees of certainty to end users. Fourth and finally, rich inference must be able to exploit efficient heuristics for composing known facts and relations to find answers, without which our enterprise becomes intractable.

Our model is shown in Fig. 2.This model emphasizes the use of heuristics and rich forms of inference. The decomposition of questions is based on heuristics which create possible alternatives to compute an answer. The model makes of use of data from different knowledge bases and automatically curates the necessary facts that it uses to compute its answers for a specific question.

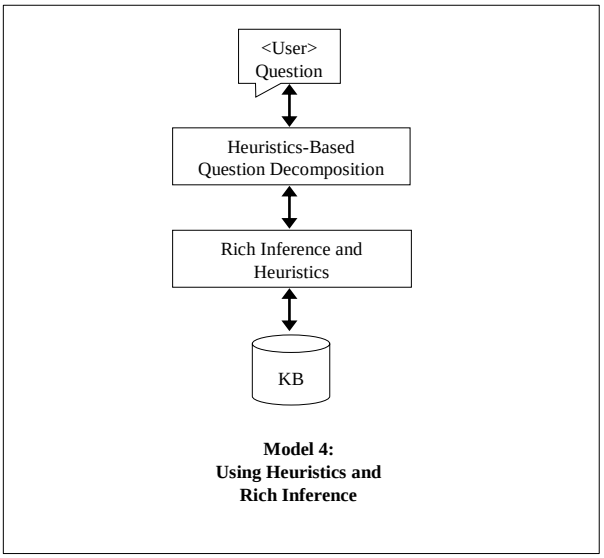

Fig. 2. Proposed model

A. Reasoning and Curation

Inference will form the core of our question-answering system. Current open domain question-answering systems are tasked with finding facts from existing data. However, question answering does not have to be limited to finding pre-stored facts since that approach fails to answer questions where the answer is not explicitly stated, or is not observed in the existing dataset. By searching and finding facts from various knowledge bases, we will dynamically create a formal representation of the knowledge. By loosely coupling the representation of the source data and the reasoning task, we are able to deal with many different data sources, including RDF triples, tabular data and other data formats. This lessens the impact of heterogeneous data representation on the web on our reasoning tasks. The local representation will be customized to the question being answered, in which case we can factor in formalisms of time and other concepts as arguments. In particular, a representation of the states of the world using first order logic and dynamic Bayesian networks enables us to reason over different time slices and about uncertainty. This rich and expressive representation will enable use of rich forms of inference to determine missing facts from a sequence of observations with gaps, or predict a future observation from past facts. In order to discover robust, generalizable relationships within and between data sources, we will use a combination of the logic-based methods described above, and, in the case of numerical data, recent methods from machine learning, e.g., flexible regression methods [20], and compositional approaches [21].

Humans use heuristics that are capable of making inferences in the face of incomplete and noisy information. Todd and Gigerenzer considered the spectrum rationality from *unbounded* to *bounded* rationality and *ecological* rationality [22]. They looked at the common types of heuristics which humans use to answer a wide range of questions. These heuristics guide the search to focus on retrieving only essential information and stopping the search when sufficient information has been retrieved to answer the question. Such heuristics can be found in causal relationships between concepts in a domain. Suppose we have a concept A which has a causal relationship with concept B such that concept A causes (or influences) B. If data on A is readily accessible, we can solve for A and infer the solution for B from solution of A.

In our system, we will use heuristics which select specific combination functions when certain features are identified from the question and the facts retrieved from the knowledge bases. Thus, we will explore the idea that heuristics can help to decompose a question and constrain the problem to known functions which can be solved in a more tractable way as opposed to using the brute force approach which tries to explore the entire search space of solutions. These kinds of meta-knowledge heuristics will allows us to select only the most appropriate and promising reasoning strategy to solve the QA problem. We will also exploit existing techniques in higher order logical reasoning and machine learning. We will use regression, classification and other inference methods in third-party machine learning libraries. This richness will allow our reasoning process to capture a wide range of computations necessary to estimate novel facts from the pre-stored facts in the local knowledge base.

B. *Heuristics and Commonsense Knowledge*

We believe that in a search space which is very large, as found in question-answering, heuristics are vital to making the problem tractable. Our approach to QA will use reasoning techniques which adopt different heuristics that constrain the problem to known functions that can solve sub-problems of the main problem, and aggregate the sub-solutions to give an answer to the user. This forms one of the core strategies we use in our rich inference techniques.

Heuristics are vital to the problem solving and decision making strategies of rational agents. Several applications of heuristics are found in computational problems that would otherwise be hard when brute force algorithms that search the entire space of solutions are used. Informed search algorithms such as greedy best first search and A* search use knowledge about the search domain to prioritize branches of the search space which lead to an optimal solution. Proponents of heuristics for problem solving such as Todd and Gigerenzer [22], Pearl [23] and Tversky and Kahneman [24] make strong arguments for the use of readily accessible information about the domain to solve problems in the domain. In [25] Simon argues that humans use a bounded form of rationality which uses approximate methods to solve tasks within reasonable amounts of time. This is in contrast to unbounded rationality where there is an assumption that agents possess some form of unbounded knowledge, unlimited memory, unbounded computational capacities and unlimited time for computations. Despite the well-defined nature of the optimal solutions to problems and the possibilities to be searched, the search space is just too large to consider all these possibilities within a reasonable amount of time. Todd and Gigerenzer further argue that these heuristics for bounded rationality can incorporate environmental structure, thereby adapting the heuristics to the environment, in what they term *ecological rationality*. Our first use of heuristics will be to make the search space manageable. When reasoning about existing knowledge from which to find novel information or do prediction, more data is usually better. However, in an automated system, where search for relevant knowledge as well as reasoning over the knowledge are both essential, more data could significantly slow down the process of finding or computing the required data. For instance, if you are interested in finding the city with the largest population, we could search for the population of all cities in every country and use a maximum function to determine the largest one. However, if you are told that large cities are found in countries with large populations, we could eliminate several countries from our search and focus only on those cities in countries with large populations. Using Todd and Gigerenzer's *recognition heuristic* [22], we could further reduce the search space to those that are recognized from the existing knowledge base. This fast and frugal heuristic can yield answers with reasonable accuracy, especially when past knowledge about the domain is limited. We will extract such commonsense knowledge from knowledge bases such as ConceptNet [26]. We refer to this category of heuristics as *space heuristics*.

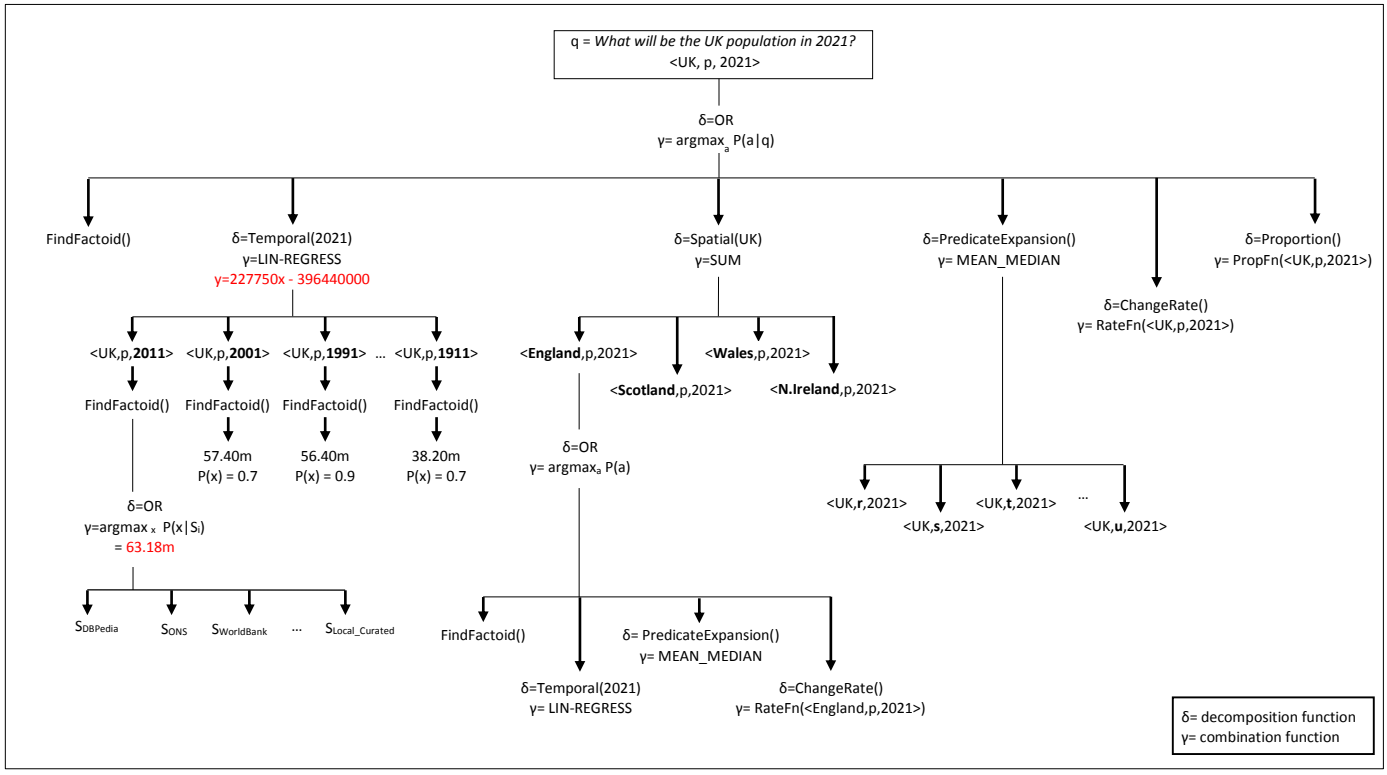

Fig. 3. Question decomposition example

Our second use of heuristics will determine the question decomposition functions and the corresponding fact combination functions to answer the question. Cues or keywords identified in the question will determine if, for instance, a temporal and (or) geo-spatial decomposition of the question is required. This would create a tree of strategies that can be used to find a solution. Multiple strategies can be pursued, and their final answers compared to estimate accuracy and confidence. We refer to the heuristics used here as *strategy heuristics*. An example of this is illustrated in Fig. 3.

C. Handling Uncertainty

The variety of sources from which data is published, the variety of representation formats, naming conventions and other cultural and contextual issues which arise in the creation and publishing of ontologies add an even greater level of uncertainty about the facts. Formal knowledge representations, such as OWL used in the Semantic Web, are based on the Open World Assumption. This means that if a fact is not stated explicitly in the ontology, it does not necessarily make it false. This carries with it a great deal of uncertainty when being used as a data resource for question-answering. Each ontology also models a different aspect of the universe (or its domain). In addition, representations of knowledge model a partial view of the world.

Our approach to QA will encounter three main forms of uncertainty:

- Understanding of the question
- Heuristics and their inherent simplification assumptions
- Noise and incompleteness of data in knowledge base.

It is a hard problem to know how well we understand the question to be answered until we get user feedback on accuracy. The use of heuristics also introduces assumptions which may oversimplify the problem and thereby create more uncertainty in the accuracy of the answer generated. Furthermore, we expect data from knowledge bases to be noisy and incomplete in some cases. To capture these uncertainties in our QA process, we will assign uncertainty weights to each *space* and *strategy* heuristic based on the extent to which it is likely to affect the accuracy of our answer. At the start of our QA process we set our confidence value, *C*, to *1.0*. Supposing we apply heuristics h_i in the process which has a weight of $w_i \epsilon [0,1]$. Our updated confidence value will be:

$$C = C(1 - w_i)$$

We intend to use a representation which is expressive enough to capture time slices of the world, the dependencies between concepts and the uncertainties associated with the data available for the QA process. We will explore a representation which combines first-order logic (FOL) and Bayesian Networks (BN). Similar to first order probabilistic logic in [27], we are able to harness the strengths of both FOL and BN in terms of expressivity and uncertainty. For the range of questions we expect our QA system to answer, we need to be able to represent both temporal and geospatial features of data in the knowledge representation. In some cases, we consider the dataset required to answer the question as spanning over a time range. This consideration helps to identify the appropriate inference technique to find answers from the representation.

Additionally, this representation will be able to capture some elements of uncertainty. Data from web sources contain a lot of noise and variance. We will define combination functions which combine these data and generate appropriate probabilities that define our belief in the data. These combination functions will also be capable of propagating the uncertainty measurements through the QA process to the final answer that is generated. Other sources of uncertainty such as unrealizability, where the appropriate combination function does not exist in our function domain, and computational complexity, where the search space is too large or is computationally intractable, mean that some of the functions yield only approximations to the expected results. Probabilistic techniques can be used to determine the prior probability of the required data variables. The large size of the search domains means that we will use approximations such as maximum likelihood estimations to compute the likelihood of each data item being consistent with the required data variables based on other data items that the search returns. Combined with learning approaches like inductive learning, which find suitable hypothesis that agree with the observed data and generalize well to new data queries, we can incorporate uncertainty directly into the representation of knowledge and computations of answers in the QA system.

IV. Key Modules of Proposed QA System

A general overview of the key steps in the proposed solution is shown in Fig. 4. A more detailed view of each module with its inputs and outputs is described in Fig. 5.

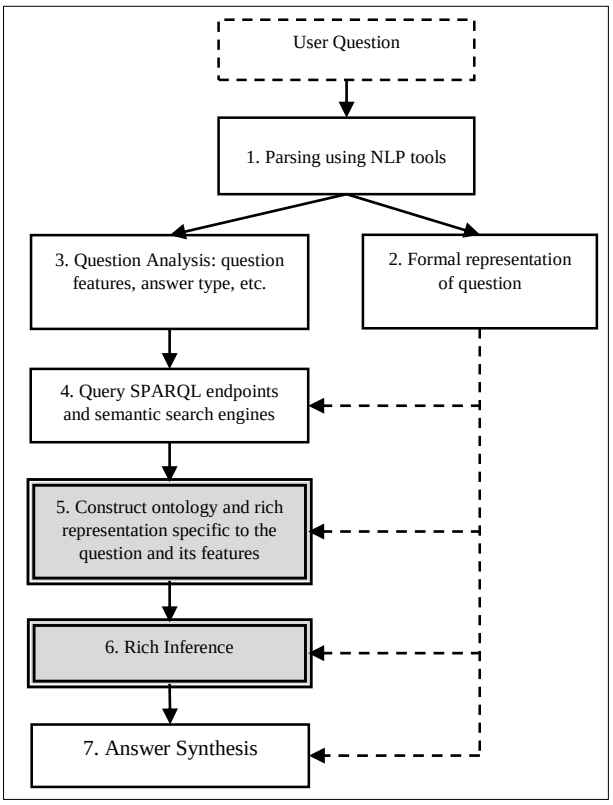

Fig. 4. QA pipeline. Shaded modules represent the primary contributions of this project

1. Parsing: This module will take as input the question text and apply syntactic and semantic parsing using NLP tools such as CoreNLP [28]. The key outputs of this module will be a semantic parse tree and a dependency tree.

*2. Feature Extraction***:** The parse trees will be analyzed to identify important features of the question. Features include the type of '*wh*-' question, the expected type of answer (a number, date, time, name, etc.), the concepts inquired about, identifiable predicates, elements of time and location and units of measurement.

3. Semantics: The parse trees will be used to generate a formal logical representation of the question. We will exploit the structure of the dependency trees, and use the part of speech tagging to extract entities and predicates. This module will use techniques in [29] to identify relationships between concepts.

4. Search: We will use heuristics which, based on the extracted features of the question and its formal representation, will create candidate strategies to solve the question. This will include spatial and temporal decompositions of concepts and predicates in the questions. The heuristics will also take advantage of indicators which influence concepts of interest to create strategies which are capable of inferring values of the variable of interest. These decompositions will guide the generation of semantic search queries for SPARQL[30] endpoints.

5. Curation: The system will construct a local, question-specific ontology for the question being processed. Data returned from the searches will be used to update this local ontology. We will use a representation which is capable of capturing time-sensitive facts [31] and question-specific details in the representation. We will also incorporate features to manage uncertainty such that results returned from queries are tagged with confidence measures based on the extent to which they align to the question's features, its representation as well as correlation or inconsistencies with data from other knowledge bases.

6. Rich Inference: We will apply rich forms of inference to compute answers to questions. We will not limit the system to RDF/OWL formalisms of the data source during inference since our local knowledge representation in the curation module is rich and expressive. This gives us room to explore additional data sources such as HTML tables. We define variables over functions and algorithms such that specific features within the question trigger the corresponding combination functions to execute. The combination functions will include numerical functions such as correlation, regression, classification, etc., using third-party mathematical, statistical and machine learning libraries.

7. Answer Synthesis: The final answer that best matches the question from module 6 will be used to construct an answer string. The form of the response will depend on the question features identified in module 3. We will adopt techniques such as [32] and [33] in answer ranking and merging in this module.

Our emphasis in this project will be on modules 5 and 6. We will use third party components as far as possible for the remaining modules since we do not intend to reinvent the wheel in these cases.

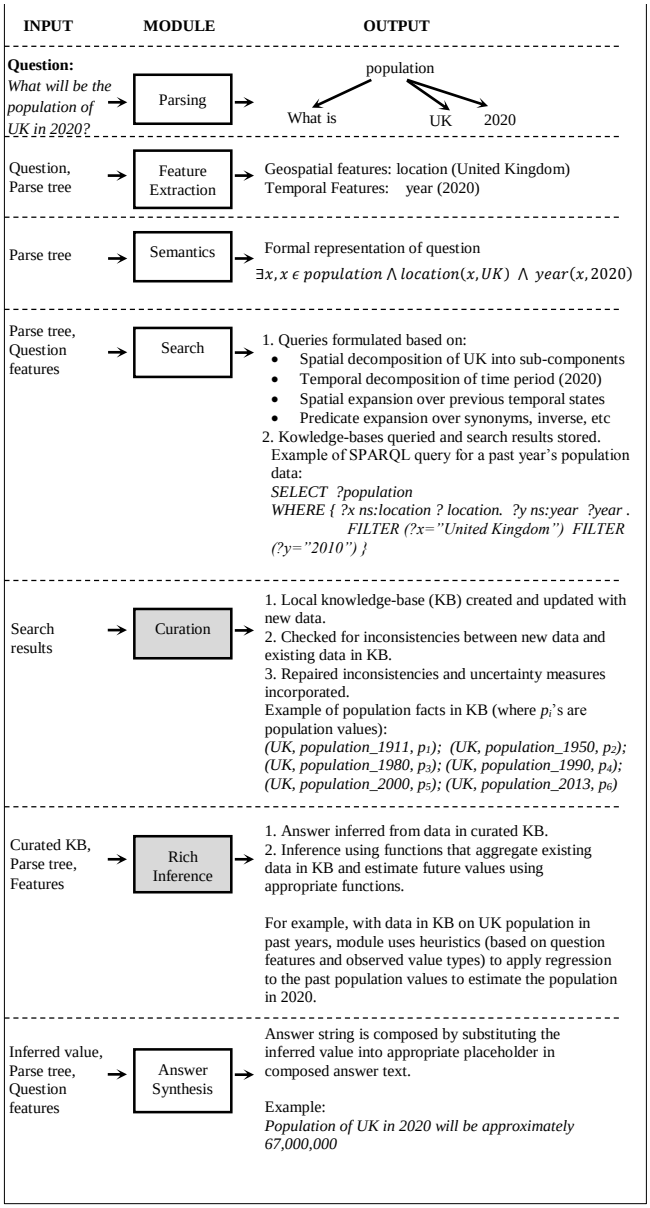

Fig. 5. Module inputs and outputs. Shaded modules represent the primary contributions of this project.

V. CONCLUSION AND FUTURE WORK

Question-answering using existing data to find novel information, not explicitly stated is possible with the vast amount of information on the web. This work aims at equipping question-answering system with the ability to decompose questions to identify the bits of information needed to infer new knowledge from existing ones. The QA system would be capable of building its own local ontologies and refining its representation incrementally as it gets more information from semantic web endpoints and semantic search engines. We aim not just to retrieve factoids from existing knowledge bases, but to use inference which enables the system to automatically find relevant data from which it can infer novel information using

rich inference techniques. These heuristics and inference techniques will allow the QA system to identify sub-questions and the corresponding combination functions which it will ultimately use to construct its final answer. The QA system would be capable of defining its level of confidence in answers that it retrieves depending on the sources of information. These techniques will improve the range of questions that question answering systems can handle successfully.

To realize this system, we will focus next on researching the best approach for identifying heuristics that will help to narrow the search for useful knowledge. We will also explore question decomposition strategies and their corresponding knowledge combination functions that constrain the QA problem to known machine learning inference methods such as regression. We will also select the appropriate third-party tools and software libraries that we can reuse for the non-core modules of our system.

REFERENCES

[1] "Wolfram|Alpha." [Online]. Available: https://www.wolframalpha.com/. [Accessed: 23-Apr-2015].

[2] A. Bundy, "The interaction of representation and reasoning," *Proceedings of the Royal Society A*, no. July, 2013.

[3] "RDF/XML Syntax Specification (Revised)." [Online]. Available: http://www.w3.org/TR/REC-rdf-syntax/. [Accessed: 04-Oct-2014].

[4] D. L. McGuinness, F. Van Harmelen, and others, "OWL web ontology language overview," *W3C recommendation*, vol. 10, no. 10, p. 2004, 2004.

[5] D. Nardi and R. Brachman, "An Introduction to Description Logics.," *Description Logic Handbook*, 2003.

[6] M. Banko, E. Brill, S. Dumais, J. Lin, and M. Way, "AskMSR : Question Answering Using the Worldwide Web," no. March, 2002.

[7] B. Katz and B. Katz, "Annotating the World Wide Web Using Natural Language," in *Proceedings of the 5th RIAO Conference on Computer Assisted Information Searching on the Internet (RIAO '97)*, 1997.

[8] I. B. M. Watson and F. Jeopardy, "Fact-based question decomposition in DeepQA," vol. 56, no. 3, pp. 1–11, 2012.

[9] E. Saquete, P. Martínez-Barco, R. Muñoz, and J. L. Vicedo, "Splitting complex temporal questions for question answering systems," *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics - ACL '04*, p. 566–es, 2004.

[10] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann, "DBpedia - A crystallization point for the Web of Data," *Web Semantics: Science, Services and Agents on the World Wide Web*, vol. 7, no. 3, pp. 154–165, Sep. 2009.

[11] F. Suchanek, G. Kasneci, and G. Weikum, "Yago: a core of semantic knowledge," *Proceedings of the 16th international conference on World Wide Web*, no. November, 2007.

[12] D. Lenat and R. Guha, "Cyc: A midterm report," *AI magazine*, vol. 11, no. 3, 1990.

[13] A. Bundy, G. Sasnauskas, and M. Chan, "Solving guesstimation problems using the Semantic Web: Four lessons from an application," *Semantic Web*, pp. 1–20, 2013.

[14] V. Lopez, M. Fernández, E. Motta, and N. Stieler, "PowerAqua: Supporting users in querying and exploring the Semantic Web," *Semantic Web*, vol. 3, pp. 249–265, 2012.

[15] N. Preda and G. Kasneci, "Active knowledge: dynamically enriching RDF knowledge bases by web services," *SIGMOD, Indianapolis*, 2010.

[16] B. Katz, G. Borchardt, and S. Felshin, "Syntactic and semantic decomposition strategies for question answering from multiple resources," in *Proceedings of the AAAI 2005 Workshop on Inference for Textual Question Answering*, 2005.

[17] C. C. Green and B. Raphael, "The use of theorem-proving techniques in question-answering systems," in *Proceedings of the 1968 23rd ACM national conference*, 1968, pp. 169–181.

[18] J. R. Slagle, "Experiments with a deductive question-answering program," *Communications of the ACM*, vol. 8, no. 12, pp. 792–798, 1965.

[19] R. F. Simmons, "Natural language question-answering systems: 1969," *Communications of the ACM*, vol. 13, no. 1, pp. 15–30, Jan. 1970.

[20] A. Wilson, E. Gilboa, J. P. Cunningham, and A. Nehorai, "Fast Kernel Learning for Multidimensional Pattern Extrapolation," *Advances in Neural Information Processing Systems*, pp. 3626–3634, 2014.

[21] R. B. Grosse, R. Salakhutdinov, W. T. Freeman, and J. B. Tenenbaum, "Exploiting compositionality to explore a large space of model structures," *arXiv preprint arXiv:1210.4856*, 2012.

[22] G. Gigerenzer and P. M. Todd, *Simple heuristics that make us smart*. Oxford University Press, 1999.

[23] J. Pearl, "Heuristics: intelligent search strategies for computer problem solving," 1984.

[24] A. Tversky and D. Kahneman, "Judgment under uncertainty: Heuristics and biases," *science*, vol. 185, no. 4157, pp. 1124–1131, 1974.

[25] H. A. Simon, "Invariants of human behavior," *Annual review of psychology*, vol. 41, pp. 1–19, 1990.

[26] H. Liu and P. Singh, "ConceptNet—a practical commonsense reasoning tool-kit," *BT technology journal*, vol. 22, no. 4, pp. 211–226, 2004.

[27] S. Glesner and D. Koller, "Constructing flexible dynamic belief networks from first-order probabilistic knowledge bases," *Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, 1995.

[28] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, "The Stanford CoreNLP Natural Language Processing Toolkit," *In Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pp. 55–60.

[29] K. A. Nuamah and L. Fu, "Biased-Incremental Clustering: A Flexible Knowledge Extraction Algorithm," *International Journal of Computer and Communication Engineering*, vol. 1, no. 1, pp. 8–12, 2012.

[30] E. Prud'Hommeaux, A. Seaborne, and others, "SPARQL query language for RDF," *W3C recommendation*, vol. 15, 2008.

[31] F. Lécué and J. Pan, "Predicting knowledge in an ontology stream," *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pp. 2662–2669, 2013.

[32] J. Ko, L. Si, and E. Nyberg, "Combining evidence with a probabilistic framework for answer ranking and answer merging in question answering," *Information Processing & Management*, vol. 46, no. 5, pp. 541–554, Sep. 2010.

[33] M. Surdeanu, M. Ciaramita, and H. Zaragoza, "Learning to rank answers to non-factoid questions from web collections," *Computational Linguistics*, no. September 2010, 2011.