Question Generation for Learning by Reading

Leo C. Ureel II, Kenneth D. Forbus, Chris Riesbeck, Larry Birnbaum

Northwestern University, Computer Science Department
1890 Maple Avenue, Evanston, IL 60201
{ ureel, forbus, c-riesbeck, birnbaum} @ northwestern.edu

Abstract

One source of questions is reflecting upon new information, as part of assimilating it into one's conceptual model of the world. We describe how we have implemented question generation as part of creating a computational system whose goal is to learn by reading.

Introduction

Questions come from many sources. One of the most important uses of questions is reflection, improving our understanding of things we have found out. People often spend hours by themselves contemplating ideas and working through issues raised by what they have read. These ideas and issues are often articulated in the form of questions. We believe that analogous processes will be necessary for systems that take responsibility for maintaining and extending their own knowledge bases. Specifically, our hypothesis is that a system that can generate its own questions to work on will be able to learn the ways in which new information fits with existing knowledge, detect missing knowledge, and provide better explanations than systems without this ability.

This paper describes work on progress on a system which generates its own questions as part of learning by reading. We begin by summarizing the context, the Learning By Reading project at Northwestern, and briefly review the analogical processing ideas we are using. We then focus on the *Ruminator*, the subsystem which assimilates new knowledge from text by generating questions which will help it fit the new knowledge with what it already knows. The first version of the Ruminator is already implemented and running, and we use examples from its operation to illustrate our techniques. We close by discussing related work and future plans.

The Learning by Reading Project

Our goal is to create a computational account of learning from reading. Our focus is on conceptual knowledge, i.e., factual knowledge, including general principles, explicit strategies, as well as particular information about the world, the kinds of things that occur in it, and how it came to be. Language, especially written language, is human culture's best invention for enabling learners to accumulate substantial bodies of conceptual knowledge. We want to enable our machines to exploit this same resource.

This has been one of AI's grand challenges of course, and remains so today. We have an approach that we believe will enable us to make considerable progress. Human cultures for centuries have taught children by using simplified language. By restricting syntax, we better enable them to focus on understanding the content that is being communicated. We are using an analogous approach. That is, unlike many natural language researchers, we are not concerned with parsing the full syntactic complexity of, say, the New York Times. Instead, our goal is to have our system deeply understand what it reads, without *a priori* bounds on the contents of texts, other than having enough background in the knowledge base to be able to make some sense of it.

We believe this simplified language approach is promising for several reasons. First, even though rewriting texts is less convenient than, say, giving the system a URL and having it read whatever text is there, it is still easier to do than writing knowledge in predicate calculus. Second, we have already successfully used this technique as part of an exploration of the role of qualitative physics in natural language semantics [6]. That NLU system used an off-the-shelf parser and did not contain any knowledge assimilation facilities. Both of these limitations are being tackled in the current project. Instead of limiting ourselves to descriptions of physical domains, we are tackling world history, including everyday political reasoning, as our initial domain. This is far broader than any other system that we know of.

Our *Learning Reader* prototype is based on the following design. We start with a large knowledge base.

Specifically, we are using the contents of Cycorp's ResearchCyc knowledge base, plus our own extensions for qualitative reasoning and analogical processing. This gives us over 34,000 concepts and over 12,000 predicates to build upon, covering a wide range of basic knowledge. (We use our own FIRE reasoning engine rather than Cycorp's because we make heavy use of analogical processing, as discussed below.) As the architecture diagram in Figure 1 illustrates, Learning Reader has three main components:

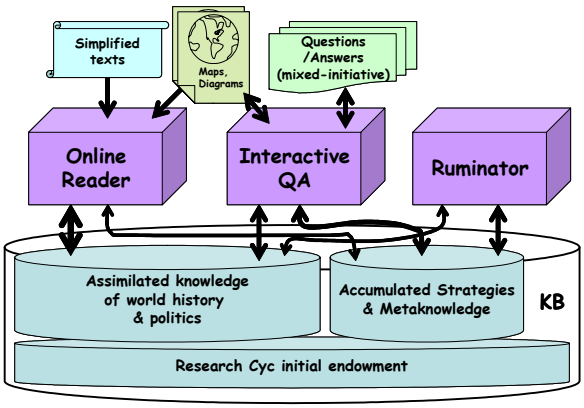

Figure 1: Learning Reader Architecture

The *Online Reader* processes simplified texts. We plan to use sketching to include maps and diagrams with texts, but currently we are focusing exclusively on text-based inputs. The *Interactive QA* system provides a means for us to test what the system has learned. We are designing this system to be mixed-initiative, so that the system can also ask us questions that it formulates, to improve its understanding. These questions are formulated in the *Ruminator*, which assimilates knowledge gleaned from texts by the Online Reader into the system's ongoing conceptual model, as represented in the knowledge base.

This paper focuses on the methods we are using in the Ruminator to generate questions, for the purpose of reflective examination and assimilation of what the system has read. Since we are using analogical processing heavily for this purpose, we briefly describe the modules we are using next.

Analogical Processing: A brief review

We are using Gentner's structure-mapping theory [5], which defines analogy and similarity in terms of alignment processes operating over structured representations. output of this comparison process is one or more *mappings*, constituting a construal of how the two entities, situations, or concepts (called *base* and *target*) can be aligned. A mapping consists of a set of *correspondences*, a set of *candidate inferences*, and a *structural evaluation score*. A *correspondence* maps an item (entity or expression) from the base to an item in the target. A candidate inference is the surmise that a statement in the base might hold in the target, based on the correspondences. The structural evaluation score indicates overall match quality.

We use three cognitive simulations based on structure-mapping theory here. The *Structure-Mapping Engine* (SME) [2] does analogical mapping. SME uses a greedy algorithm to compute approximately optimal mappings in polynomial time. The base and target descriptions can be pre-stored cases, or dynamically computed based on queries to a large knowledge base. MAC/FAC [4] models similarity-based retrieval. The first stage uses a special kind of feature vector, automatically computed from structural descriptions, to rapidly select a few (typically three) candidates from a large case library. The second stage uses SME to compare these candidates to the probe description, returning one (or more, if they are very close) of them as what the probe reminded it of. The third simulation is SEQL [7], which models generalization. Human generalization is conservative, but sometimes works over an order of magnitude faster than today's statistical algorithms. SEQL uses SME to combine highly similar examples into new generalizations and to assimilate examples that fit existing generalizations.

Question Generation in Rumination

An important part of learning by reading is the process of rumination; i.e., reflecting on what has been read and it fits into an evolving conceptual model of the work. This includes finding gaps and inconsistencies in its own knowledge and finding ways to resolve them. It is an off-line process, exploring ambiguities and other issues that the system did not have time to process during reading or interaction. We believe rumination is an important process in large-scale learning systems.

The Ruminator generates questions in reaction to the introduction of new knowledge into the system. These questions drive the reflection process, which seeks to answer them, perhaps generating yet more questions in the process of doing so. Questions which cannot be answered, and in particular open questions which come up repeatedly, provide evidence about gaps in its knowledge and provide the basis for formulating learning goals.

Historically, the Socratic Method has been recognized as a way of guiding students to understanding by prompting them to explore a domain through the use of thorough questioning. We believe that the Socratic tutor is also role-modeling the appropriate questions to ask when tackling a problem or exploring a new knowledge domain. We further believe that active learning involves internalizing the questions posed by the tutor and learning to apply them to apply them when related problem or novel domains present themselves. Because of this, we are wiring the Socratic Method into the heart of the Ruminator's mechanisms, as amplified below.

The Ruminator uses the following algorithm to generate questions:

1. Determine people, places, and events in story.
   - Apply axioms to generate specific questions based on the Journalist's Questions.
2. Retrieve related knowledge using similarity-based retrieval.
3. Compare to related knowledge and previous stories.
   - Use candidate inferences from analogy to generate Yes/No Questions.
4. Compare to generalizations of previous stories of the same type.
   - Generate questions concerning surprising differences.
5. Apply the Socratic Method.
   - Incrementally update generalizations with new story.
   - Ask whether or not the updated generalizations can be refuted

We discuss each of these operations in turn next.

Story Analysis

The input to the Ruminator comes from the Online Reader. (We plan to take input from the Interactive Q/A system as well in the near future.) It receives one or more conjectured interpretations of the story, as produced by the Online Reader. It also receives some meta-information: The person who entered the story, the original text of the story[1], and the topic of the story, as indicated by the user. The topics currently include people, countries, and terrorist events.

A standard way of understanding an event is to ask what are known as the *Journalist's Questions:* Who, What, When, Where, Why, and How. The Ruminator begins by extracting the Who, What, and Where of the story, since these often are stated explicitly. Background knowledge about the entities identified by this analysis is extracted from the background KB and added to the representation of the story.

For example, suppose the following fact is given to the Ruminator as part of the Online Reader's understanding of a story:

```
(deathToll
 TerroristAttack-06-25-1996-Dhahran-Saudi-Arabia
 UnitedStatesPerson 19)
```

The Ruminator identifies *UnitedStatesPerson* as a Who and *TerroristAttack-06-25-1996-Dhahran-Saudi-Arabia* as a What. At this point, the Ruminator tries to determine the primary subject of the story by counting which of the subjects identified is mentioned most frequently. To confirm, the Ruminator generates a question such as:

- "Is Gabriela Mistral central to the story of Chile?"

Examples of the types of journalist questions generated by the system for events include:

- Who caused this event?
- Who did this event?
- Who is affected by this event?
- Who is responsible for this event?
- Who benefits from this event?
- Where did this event happen?
- When did this event happen?
- Why did this event occur (i.e., what are its casual antecedents)?
- What are the likely consequences of this event?

Specific forms of the Journalist's Questions are generated via a query using prolog-style backchaining. Some can be answered by simple queries, others may be answered by inference, and those of the compare/contrast variety may require analogy be performed to determine an answer. Here is an example of such an axiom:

```
(<== (ruminatorQuestion (QuestionFn ?qid)
       (JournalistQuestionFn
         WhoIsResponsibleForThisEvent?
         (TheList ?event)))
   (storyMentionsEvent
     (StoryMeaningCaseFn ?story ?index) ?event)
  (individualRepresenting
    (JournalistQuestionFn
       WhoIsResponsibleForThisEvent?
      (TheList ?event)) "Question" ?qid))
```

Internally, questions generated by the Ruminator are represented in CycL:

```
(ist-Information (StoryMeaningMetaCaseFn
                   lr-story-1 8)
 (ruminatorQuestion (QuestionFn Question-24)
  (JournalistQuestionFn WhoIsAffectedByThisEvent?
   (TheList
    (NthEventOfTypeInOnFn TerroristAttack
     SaudiArabia
     (DayFn 25 (MonthFn June (YearFn 1996)))
     19533771)))))
```

These questions are often more understandable in pidgin English:

- "Who is affected by the terrorist attack in SaudiArabia on 25 of the month of June in the year 1996?"

We use natural language generation templates in ResearchCyc to automatically produce such text. Other

[1] We intend to make the same extended DMAP [9] language parser used in the Online Reader available to the Ruminator, so that it can try rereading texts as it learns more.

examples of journalist-style questions generated by the system are:

- "Is Bin Laden associated with Saddam Hussein?"
- "How is Gabriela Mistral related to Augusto Pinochet Ugarte?"
- "Who benefits from the TerroristAttack-October-12-2002-Bali-Indonesia?"
- "What caused BombingOfSantiagoOfficeBuildingHousingFluorDaniel-01?"

Analogy-based Questions

Comparing the new to the old is a valuable technique for understanding something. The Ruminator uses MAC/FAC to retrieve relevant prior stories and concepts. The case libraries it uses are determined by the topic selected for the story, plus the categories of story elements (e.g., people, places). Recall that the FAC stage of MAC/FAC uses SME to compute a comparison between the probe and a retrieved memory item. This mapping process can produce candidate inferences, which in turn can become questions. That is, a candidate inference may or may not be true. Moreover, a candidate inference might suggest the existence of a new entity in the story, what is called an *analogy skolem*. We suspect that analogy skolems will be an important force in regularizing a system's knowledge. For example, in the pidgin English generated by the Ruminator below, the phrase "something like" indicates that the next constituent represents an analogy skolem:

- "By comparing the cases g294900 with g305522: Is it true that something like Al-Qaida is a perpetrator in the January 31, 1996 truck bombing of Central Bank Building in Colombo, Sri Lanka?"
- "By comparing the minimal case TerroristAttack-August-1998-Nairobi-Kenya and the case (StoryMeaningCaseFn lr-story-1 0) is it true that the number of Person which are injured in a something like TerroristAttack-August-1998-Nairobi-Kenya event is something like 85?"

The Socratic Method

Building on a tradition that has roots in classical rhetoric, philosophy, and logic, we are implementing the Socratic Method as a technique for evaluating new generalizations. The Socratic Method consists of the following steps:

1. Agree on topic or problem.
2. Propose a generalization.
3. Try to find a refutation by counter-example.
   - If refuted,
     Then abandon the generalization
     or go to 2 to effect a repair.
     Else accept generalization as provisionally true
     and go to 1.

Just as we organize story information into case libraries concerning people, places, etc. to support retrieval of specific prior knowledge, we also use SEQL to accumulate generalizations for each of these topics. The first step is to compare the new story with the most similar existing generalization for the given topic. This comparison is used to generate questions about differences, focusing on what is novel about the new story. After these questions are generated, the new story is provided to SEQL for assimilation.

Currently we generate a single question for each generalization: Can we find a counterexample to refute it? Tackling this question will be the next step in fully implementing the Socratic Method in the Ruminator.

Issues

These techniques generate a substantial volume of questions. For example, the single sentence story "The attack killed 19 people", generates 2052 questions from the information provided by current implementation of the Online Reader. Evaluation of questions is clearly a critical issue to tackle next. Our current approach is to prioritize them by type. Breaking them down for this example, we get:

1. Questions about Generalizations. (2)
2. Journalist's Questions. (124)
3. Questions about candidate inferences. (1926)

A key prerequisite to developing more sophisticated question evaluation methods is improving the Ruminator's ability to answer its own questions. Consider for instance the questions derived from candidate inferences. We suspect that many (or most) of them can be answered by relatively straightforward reasoning. For instance, simple taxonomic reasoning suffices to answer the following questions:

- "By comparing the case (StoryMeaningCaseFn lr-story-1 5) and the case (StoryMeaningCaseFn lr-story-1 2) is it true that TerroristAttack-June-25-1996-Dhahran-Saudi-Arabia is a attack?"
- Is it true that AugustoPinochetUgarte's biological father is Chile?
- Is it true that the City of San Antonio's spouse is Chile?

Our strategy will be to weed out the easy questions as quickly as possible, and use this process to learn more refined question-posing strategies to avoid producing silly or obvious questions in the first place. We also plan to make the question prioritization scheme adaptable, so that the Ruminator can learn over time which questions are likely to be interesting enough to ask of its human teachers via the Interactive Q/A system.

Related Work

Collins and Stevens [1] developed a formal theory of Socratic tutoring, which has greatly influenced our work. Lehnert [8] did an informative analysis of questions and the process of question answering. Sammut and Banerji [15] investigated the learning of concepts through experimentation and question asking using methods similar to the Socratic method described above. Ram [12,13] developed a theory of questions and question answering that was implemented in his thesis project AQUA. Moorman [10] developed a theory of creative reading. Recently, Straach and Truemper [16] developed a technique for learning to ask relevant questions in expert systems. Otero and Graesser [11] have developed the PREG cognitive model of human question asking.

Discussion

One source of questions is reflecting on new information, assimilating it as part of one's knowledge. We have described how question generation works in the Ruminator, part of the Learning Reader prototype being constructed by the Learning by Reading project at Northwestern. We are using a combination of template-style questions (e.g., the Journalist's Questions) and comparison-based questions, building on analogical processing models for matching, retrieval, and generalization. In particular, these techniques appear to give us the mechanisms we need to create a computational system that is capable of using the Socratic Method to improve its own knowledge.

This project is in its early stages, and much work lies ahead. As discussed above, our next step is to improve the Ruminator's ability to quickly answer obvious or silly questions. This should winnow the set of questions down to a manageable size, making question ranking the next priority. Finally, we plan to add proactive knowledge gathering strategies, including asking questions of its human teachers and asking them for more stories, to achieve its own learning goals. One key strategy will be the Socratic Method.

Acknowledgements

This research was supported by the Information Processing Technology Office of the Defense Advanced Research Projects Agency. The opinions expressed here are those of the authors, and not those of the US Government.

References

1. Collins, A., & Stevens, A. L. (1982). Goals and strategies of inquiry teachers. In R. Glaser (Ed.), Advances in instructional psychology (Vol. 2, pp. 65-119). Hillsdale, NJ: Erlbaum. 1982.
2. Falkenhainer, B., Forbus, K. and Gentner, D. The Structure-Mapping Engine. *Proceedings of the Fifth National Conference on Artificial Intelligence.* 1986.
3. Forbus, Kenneth D. and Gentner, D. (2001) Exploring analogy in the large, Kenneth D. Forbus in Gentner, D., Holyoak, K., and Kokinov, B. (Eds) *The Analogical Mind: Perspectives from Cognitive Science*. MIT Press, 2001
4. Forbus, K., Gentner, D. and Law, K. (1995) MAC/FAC: A model of Similarity-based Retrieval. *Cognitive Science*, 19(2), April-June, pp 141-205. 1995.
5. Gentner, Dedre, (1997) Structure Mapping in Analogy and Similarity, Gentner, D. and Markman, A. *American Psychologist*, January, pp 45-56, 1997
6. Kuehne, S. and Forbus, K. 2004. Capturing QP-relevant information from natural language text. *Proceedings of the 18th International Qualitative Reasoning Workshop*, Evanston, Illinois, USA, August
7. Kuehne, Sven and Forbus, K., Gentner, D., Quinn, B. (2000) SEQL: Category learning as progressive abstraction using structure mapping, Kuehne, S., Forbus, K., Gentner, D. and Quinn, B. 2000. *Proceedings of CogSci 2000*, August, 2000.
8. Lehnert, W.G. (1978) *The Process of Question Answering*, Lehnert, W.G., Lawrence Erlbaum and Associates, Hillsdale, N.J., 1978
9. Martin, C.E. and Riesbeck, C.K. Uniform Parsing and Inferencing for Learning. *Proceedings of the Fifth National Conference on Artificial Intelligence*, Philadelphia, PA, August 11 - 15, 1986, pp 257-261.
10. Moorman, Kenneth. (1994) A Functional Theory of Creative Reading], Kenneth Moorman, Ashwin Ram. The Psycgrad Journal. Technical Report GIT-CC-94/01, College of Computing, Georgia Institute of Technology, Atlanta, GA, 1994.
11. Otero, Jose and Graesser, Arthur C. (2001) PREG: Elements of a Model of Question Asking, Cognition and Instruction, 19(2), 143-175, Lawrence Erlbaum Associates, 2001
12. Ram, Ashwin. (1994) AQUA: Questions that Drive the Explanation Process, Ashwin Ram. Inside Case-Based Explanation, R.C. Schank, A. Kass, and C.K. Riesbeck (eds.), 207-261, Lawrence Erlbaum, 1994.
13. Ram, Ashwin. (1994) A Theory of Questions and Question Asking, Ashwin Ram. The Journal of the Learning Sciences, 1(3&4):273-318, 1991.
14. Riesbeck, Christopher K. (1989) Inside case-based reasoning, Christopher K. Riesbeck, Roger C. Schank. Hillsdale, N.J. : L. Erlbaum, 1989.
15. Sammut, Claude and Banerji, Ranan B. (1986) Learning Concepts By Asking Questions, Machine Learning: An Artificial Intelligence Approach, 1986
16. Straach, Janell and Truemper, Klaus. (1999) Learning to ask relevant questions. Artificial Intelligence, 111 (1999) 301-327, Elsevier, 1999