INFUSING AUTOMATIC QUESTION GENERATION
WITH NATURAL LANGUAGE UNDERSTANDING
Karen Mazidi

Dissertation Prepared for the Degree of
DOCTOR OF PHILOSOPHY

UNIVERSITY OF NORTH TEXAS
December 2016

APPROVED:
Paul Tarau, Major Professor
Rada Mihalcea, Co-Major Professor
Cornelia Caragea, Committee Member
Kathleen Swigger, Committee Member
Barrett Bryant, Chair of the Department of
Computer Science and Engineering
Costas Tsatsoulis, Dean of the College of
Engineering
Victor Prybutok, Vice Provost of the Toulouse
Graduate School

Automatically generating questions from text for educational purposes is an active research area in natural language processing. The automatic question generation system accompanying this dissertation is MARGE, which is a recursive acronym for: MARGE automatically reads generates and evaluates. MARGE generates questions from both individual sentences and the passage as a whole, and is the first question generation system to successfully generate meaningful questions from textual units larger than a sentence. Prior work in automatic question generation from text treats a sentence as a string of constituents to be rearranged into as many questions as allowed by English grammar rules. Consequently, such systems overgenerate and create mainly trivial questions. Further, none of these systems to date has been able to automatically determine which questions are meaningful and which are trivial. This is because the research focus has been placed on NLG at the expense of NLU. In contrast, the work presented here infuses the questions generation process with natural language understanding. From the input text, MARGE creates a meaning analysis representation for each sentence in a passage via the DeconStructure algorithm presented in this work. Questions are generated from sentence meaning analysis representations using templates. The generated questions are automatically evaluated for question quality and importance via a ranking algorithm.
Copyright 2016
by
Karen Mazidi
ACKNOWLEDGMENTS

For my family, thank you:

- To my husband, Muhammad Ali Mazidi, whose support made this degree possible.
- To our sons, Nabil and Michael, for their encouragement.
- To my parents, Bob and Charlotte Gillispie, the heart and soul of our family.

To my advisor, Paul Tarau, who was the first to say that I should publish my question generation work back when it was a class project: thank you for your encouragement.
TABLE OF CONTENTS

ACKNOWLEDGMENTS iii

CHAPTER 1 INTRODUCTION 1
  1.1. The QG (Question Generation) Task 2
  1.2. Research Overview 3
  1.3. Contribution Summary 4
  1.4. Research Goal 4
  1.5. Dissertation Organization 5
  1.6. Associated Publications 5

CHAPTER 2 PEDAGOGICAL VALUE OF QUESTIONING 6
  2.1. Overview of Research on Questioning 6
      2.1.1. Adjunct Questions 6
      2.1.2. Assessment Questions 7
      2.1.3. Self-study Questioning Techniques 7
      2.1.4. Types of Questions and Effectiveness 8
  2.2. Taxonomy of Questions 9
      2.2.1. Bloom’s Taxonomy 9
      2.2.2. Lehnert-inspired Taxonomies 10
      2.2.3. MARGE Taxonomy 12
  2.3. Question Levels in Practice 13
      2.3.1. Questions in the Classroom 13
  2.4. Conclusions 14

CHAPTER 3 DESIGN DECISIONS IN PRIOR WORK 16
  3.1. Design Decisions 16
      3.1.1. Determine QG System Purpose 17

iv
3.1.2. Determine Question Types
3.1.3. Identify an Input Source
3.1.4. Select an Approach
3.1.5. Wolfe’s AUTOQUEST

3.2. Recent Work in QG

3.2.1. Design Approach: WordNet
3.2.1.1 Aist: 2001
3.2.1.2 Brown, Frishkoff, and Eskenazi: 2005

3.2.2. Design Approach: QG from Key Terms
3.2.2.1 Agarwal and Mannem: 2011
3.2.2.2 Becker, Basu and Vanderwende: 2012

3.2.3. Design Approach: QG from Ontologies
3.2.3.1 Olney, Graesser, and Person: 2012
3.2.3.2 Afzal and Mitkov: 2014
3.2.3.3 Jouault and Seta: 2014
3.2.3.4 Chaudhri, Clark, Overholtzer and Spaulding: 2014

3.2.4. Design Approach: PSG Parser and Templates
3.2.4.1 Rus, Cai, and Graesser: 2007
3.2.4.2 Wyse and Piwek: 2009
3.2.4.3 Liu, Calvo and Rus: 2010

3.2.5. Design Approach: PSG Parser and Transformation Rules
3.2.5.1 Gates: 2008
3.2.5.2 Heilman and Smith: 2009, 2010
3.2.5.3 Kalady, Elikkottil and Das: 2010
3.2.5.4 Ali, Chali and Hasan: 2010

3.2.6. Design Approach: SRL parsing and Transformation Rules
3.2.6.1 Mannem, Prasad and Joshi: 2010
3.2.6.2 Chali and Hassan: 2015
3.2.7. Design Approach: SRL parsing and Templates
   3.2.7.1 Lindberg, Popowich, Nesbit and Winne: 2013
   3.2.7.2 Mazidi: 2014
3.2.8. Design Approach: Discourse Analysis
   3.2.8.1 Agarwal, Shah and Mannem: 2011
3.2.9. Design Approach: Dependency Parse and Templates
   3.2.9.1 Mazidi: 2015

3.3. One-Off Approaches
   3.3.1. Kunichika, Katayama, Hirashima and Takeuchi: 2004
   3.3.2. Curto, Mendes and Coheur: 2012
   3.3.3. Yao and Zhang: 2010

3.4. Special-Purpose Question Generation
   3.4.1. Project Listen: 1996-present
   3.4.2. AutoTutor: 1997-present
   3.4.3. Comprehension SEEDING: 2011-2014

3.5. Conclusions

CHAPTER 4 MARGE-S

4.1. MARGE
4.2. The Passage Class
4.3. The Section Classs and the Sentence Class
4.4. The DeconStructure Algorithm
   4.4.1. Theoretical Foundations
   4.4.2. Parser Comparisons
   4.4.3. MAR versus other Representations
   4.4.4. Practical Advantages of Multiple Parsers
   4.4.5. Determining Sentence Structure
4.5. Question Generation
   4.5.1. Templates
4.5.2. Ranking Question Importance 55
4.6. Summary 56

CHAPTER 5 MARGE-P 58
5.1. Topic Modeling 58
5.1.1. Topic Modeling Questions 59
5.2. TextBlob 61
5.2.1. Noun Phrase Questions 61
5.3. Head Words of Subjects and Direct Objects 61
5.3.1. Terminology Questions 62
5.4. Section Topic 62
5.4.1. Summary Question 62
5.5. Discussion 63

CHAPTER 6 EVALUATING AUTOMATICALLY GENERATED QUESTIONS 64
6.1. What are we Measuring? 64
6.2. IR Metrics: Precision, Recall and F Measure 65
6.3. Pedagogical Metrics: Item Test Theory 65
6.4. Acceptability Determined by Selected Judges 66
6.5. Acceptability Determined by Crowd Sourcing 68
6.6. Pedagogical Utility 69
6.6.1. Pedagogical Utility compared to Human-authored Questions 69
6.6.2. Pedagogical Utility Given a Tutoring Scenario 70
6.6.3. Pedagogical Evaluation of ITS 71
6.7. Turing-type Evaluations 71
6.8. Ranking Generated Questions Programmatically 73
6.9. Discussion of the Merits of Evaluation Schemes 74

CHAPTER 7 MARGE EVALUATIONS 75
7.1. Evaluation 1: All MARGE-S Sentence-Generated Questions 75
7.1.1. Test Data and Results

7.2. Evaluation 2: Highest Ranked MARGE-S Questions
   7.2.1. Test Data
   7.2.2. Results

7.3. Evaluation 3: MARGE-P Question Breadth and Depth
   7.3.1. Question Depth

7.4. Evaluation 4: MARGE-P Questions

CHAPTER 8 SUMMARY OF THIS WORK AND LOOKING FORWARD

APPENDIX A SAMPLE MARGE-S SENTENCE-GENERATED QUESTIONS

APPENDIX B TEMPLATES AND DESCRIPTIONS

APPENDIX C SAMPLE MARGE-P PASSAGE QUESTIONS

APPENDIX D SAMPLE TOPIC MODELING RESULTS

APPENDIX E COMPARISON OF MAR AND AMR

BIBLIOGRAPHY
CHAPTER 1

INTRODUCTION

In the 1980s, working as a computer programmer on an IBM 360, I noticed a program residing on disk called Eliza. None of my fellow programmers knew what it was, so I fired it up to see what would happen. Eliza started communicating with me! I found it fascinating to have a “conversation” with Eliza but since there was no available Internet at that time, I had no way of finding out more about this program. Eliza now has a Wikipedia page\footnote{https://en.wikipedia.org/wiki/ELIZA} and there are various implementations online you can interact with, as in the following exchange:

Mazidi: What do you think of natural language generation?
Eliza: We were discussing you, not me.

Eliza was a chimera, but one I am happy to have encountered since it sparked a decades-long interest in the field of natural language generation, a subfield of natural language processing, which in turn is a subfield of AI [97]. This serendipitous encounter with Eliza decades earlier motivated my interest in generating questions automatically. Specifically, this dissertation research involves automatically generating questions from text for educational purposes.

It’s almost a given that any NLG (Natural Language Generation) system will be met with the challenge: “How is this different from Eliza?” and on some level, it will always be a legitimate question, until the day when we can state with the certainty provided by a Turing test that computers understand natural language. Nevertheless, the examination of current and ongoing work in automatic question generation discussed in this dissertation demonstrates that the state of the art has reached the point where automatically generated questions are useful for learners, educators, and educational applications, and will continue to increase in utility with further advances in the field.
1.1. The QG (Question Generation) Task

Rus et al. [103] describe QG as a dialogue and discourse task, drawing on both NLU (Natural Language Understanding) and NLG (Natural Language Generation). This description is particularly apt for a system such as the one outlined in this dissertation which takes raw text as an input, performs NLU analysis [6], and then performs the NLG task of transforming the intermediate representations into English language questions. McDonald describes NLG as the process by which thought is rendered into language [54]. In generating questions from text, we are taking a thought represented in declarative text and rendering it into another form of thought: questioning on that original thought. Piwek and Boyer [96] observe that Question Generation could be viewed as a search for algorithms to transform inputs to certain types of outputs, and that the combinations of inputs, outputs and algorithms are already quite varied in this developing research area.

Question generation has also received attention from fields other than AI. Examining questions as logical entities, Cohen [29] identified a question with a propositional function in which constants are replaced by variables, such as wh-words. From the prospective of the philosophy of language, Groenendijk [47] introduces an interrogation dialogue game which serves as an example for recasting propositional logic as cooperative information exchange. A logical notion of pertinence was proposed with elements such as contextual consistency, non-entailment, and licensing, which correspond to elements of the Gricean Cooperation Principle. Grice’s maxims identify the assumptions that humans make in communication. Paraphrasing the maxims: let your communication be accurate, of appropriate length, relevant, and clear [46]. Note that these maxims could serve as guides for automatically generated text as well as human communication. Moving from the philosophy of language to linguistics, Ginzburg’s work on QUD (Questions Under Discussion) [41] frames discourse as a series of questions to be addressed. Expository text could be viewed from this perspective also: it is a monologue discourse from which the author assumes the reader would be able to answer questions. Automatic question generation, then, could be viewed as a process of discovering unasked questions within the monologue.
These rich and diverse ideas about questions indicate the importance of questions, and by extension, the goal of generating them automatically. A discussion of the importance of questions in a pedagogical setting can be found later in this dissertation.

1.2. Research Overview

Prior work in QG treats a sentence as a string of constituents to be rearranged into as many questions as allowed by English grammar rules. Consequently, such systems overgenerate and create mainly trivial questions. Further, none of these systems to date has been able to automatically determine which questions are meaningful and which are trivial. This is because the research focus has been placed on NLG at the expense of NLU. In contrast, the work presented here infuses the questions generation process with NLU.

The automatic question generation system accompanying this dissertation is MARGE\(^2\), which is a recursive acronym for: MARGE Automatically Reads Generates and Evaluates. MARGE generates questions from both individual sentences and the passage as a whole, and is the first QG system to successfully generate meaningful questions from textual units larger than a sentence. The word marge has its origins in the Latin margo, which means margin, or edge. MARGE pushes the edges of the state of the art by informing the NLG process with NLU techniques at both the sentence and passage level.

From the input text, MARGE creates a Meaning Analysis Representation for each sentence in a passage via the DeconStructure algorithm presented in this work. Questions are generated from sentence Meaning Analysis Representations using templates. The generated questions are automatically evaluated for question quality and importance via a ranking algorithm.

MARGE also generates conceptual questions from the passage as a whole in addition to fact-based questions generated from sentences. By combining NLP techniques with topic modeling, the key information of the passage is identified. Additional techniques identify concepts and their relations to each other. In this way, questions can be generated which identify the high-level concepts contained in the passage.

\(^2\)http://www.karenmazidi.com/projects.html
1.3. Contribution Summary

MARGE has pushed the boundaries of automatic question generation from the domain of simple text processing into the domain of Artificial Intelligence by infusing NLG with NLU. Specifically, MARGE demonstrates its capacity to:

1. Create a semantic representation, the Meaning Analysis Representation, of each sentence in the passage by means of the Deconstructure Algorithm.
2. Classify the sentence meaning via analysis of constituent patterns.
3. Match templates to the Meaning Analysis Representation to generate questions over sentence content that carries a significant semantic load.
4. Generate questions that leverage sentence constituent patterns to probe the central semantic import of the sentence.
5. Automatically evaluate the importance of generated questions by means of heuristics based on the TextRank algorithm.
6. Generate passage-level questions by identifying key passage topics as well as important entities and their relations. MARGE is the first QG system to successfully break through the sentence barrier.

1.4. Research Goal

Infusing automatic question generation with natural language understanding analysis substantially advances the state of the art. At the sentence level, the overgeneration problem of prior work is solved by determining the central point of a sentence, and generating a question that hones in on that point. At the passage level, NLU analysis determines what the passage is trying to communicate, so that questions can be generated about these key ideas. This work advances the state of the art of automatic question generation to the point that the overwhelming majority of MARGE’s output questions are meaningful, quality questions. This has significant educational implications as these questions can be used by educators, independent learners, and incorporated into educational technologies such as intelligent tutoring systems and other advanced learning technologies.
Research question:

Can infusing NLU techniques into QG lead to higher quality questions compared to approaches that employ syntactic manipulation alone, and can these techniques move the state of the art closer to the quality of human-authored questions?

1.5. Dissertation Organization

The remainder of this dissertation is organized as follows:

• Chapter 2 discusses the pedagogical value of questioning.
• Chapter 3 examines prior work in automatic question generation.
• Chapter 4 gives an overview of the question generation system components that generate questions from sentences (MARGE-S).
• Chapter 5 describes system components that generate questions from the passage as a whole (MARGE-P).
• Chapter 6 surveys methods of evaluating automatically generated questions.
• Chapter 7 describes experiments to evaluate the quality of generated questions.
• Chapter 8 provides a summary of the work.

1.6. Associated Publications

This dissertation represents on-going research that I have been pursing since early in my PhD work. Iterations of this research have been published in top-tier conferences:

(1) 2014 Intelligent Tutoring Systems [75]
(2) 2014 Association for Computational Linguistics [76]
(3) 2015 Artificial Intelligence in Education [77]
(4) 2016 Intelligent Tutoring Systems (ITS 2016) - nominated for best paper [78]
(5) 2016 International Natural Language Generation Conference (INLG 2016) [79]

The work has seen a series of iterative development cycles, driven by the goal of seeking productive algorithms for automatic question generation and extending the state of the art of question generation.
CHAPTER 2

PEDAGOGICAL VALUE OF QUESTIONING

The questions generated by MARGE are suitable for a wide range of educational applications as well as for use by learners and teachers. Although automatic question generation is an interesting Natural Language Processing task in its own right, the utility of the generated questions should not be overlooked, as this utility is a powerful motivation for the work. All of the question generation systems discussed in the next chapter were predicated on the idea that questioning improves learning outcomes. This chapter explores the research supporting this idea, discusses the pedagogical value of different question types, and proposes a taxonomy suitable for classifying questions generated for educational purposes from expository text.

2.1. Overview of Research on Questioning

In the educational psychology literature, questioning is a vast subject. Here, we focus on research about questioning that has been replicated in numerous studies over decades.

2.1.1. Adjunct Questions

Adjunct questions are assessment questions interspersed in text that have been shown to boost retention and comprehension in studies that have been conducted for several decades; notable research includes Rothkopf [100], and Anderson and Biddle [8]. There are two kinds of adjunct questions. Look-back questions ask about material previously read, look-ahead questions ask questions about the material to come, essentially priming the student to think about the upcoming material. Research by Roediger and Karpicke [98] indicates that the look-back questions promote better retention that look-ahead questions. Peverly and Wood [95] note that most adjunct question research has focused on college students. In contrast, they tested the effects of questions on reading disabled adolescents. Their study found that inserted questions were more effective than massed postquestions and that the effects on comprehension increased over the 6-week period of their study.
2.1.2. Assessment Questions

In an article applying cognitive psychology to educational practice, Roediger and Pyc [99] outline strategies for improving education that are inexpensive and backed by empirical research. Based on an exhaustive review of research providing data about which educational strategies work and which do not, they have distilled this information into three principles:

1. the distribution (spacing and interleaving) of material and practice during learning
2. the frequent assessment of learning (direct and indirect positive effects of quizzing and testing)
3. explanatory questioning (elaborative interrogation and self explanation; that is, having students ask themselves questions and provide answers or to explain to themselves why certain points are true.

Note that two of the recommendations involve questioning, and the other one involves the timing and presentation of questions. When a student answers a factual question over material they have previously studied, this is called retrieval practice. Numerous studies have shown that retrieval practice is a powerful means of improving retention of general knowledge facts [21], middle school science and social studies topics [80], university level statistics and biological basis of behavior material [72], medical education [63], to cite just a few studies.

Testing, whether by an instructor or self-testing while studying, has several positive benefits. First, retrieving information makes it more retrievable in the future and can transfer to other concepts. Testing helps students identify what they know and what they need to study further. Test potentiation studies show that students learn more restudying after taking a test than if they have not taken a test [99].

2.1.3. Self-study Questioning Techniques

Self-explanation and elaborative interrogation are two techniques that have been described in the literature for questioning during reading. Both strategies slow reading, but have been shown to improve comprehension and learning [99]. Self-explanation has been
shown to be an effective strategy across various ages and content areas [36]. Self-explanation involves a learner explaining a concept out loud during reading or problem solving. Fonseca and Chi [37] surveyed 20 years of research supporting the efficacy of self-explanation. In exploring why this technique works, Chi divided learning activities into four categories: passive, active, constructive, and interactive. Activities are categorized not only by what the learner is doing but by the cognitive processes inferred to be occurring during the activity. Research indicates that: interactive > constructive > active > passive. Self-explanation is in the constructive category. It could be in the interactive category when paired with a partner in a dialogue in which they may challenge each others’ explanations.

Elaborative interrogation involves having students provide explanations for each significant sentence they have read. In other words, for facts that are read, learners must generate their own explanation of why it is the case. Seifert [104] reported average effect sizes in the range 0.85 to 2.57. The forms of elaborative interrogation in various studies have been: Why is this true?, Why does this make sense?, Why?, but the most common format is Why would this fact be true of this [X] and not some other [X]? By analogy to machine learning, this could be seen as learning through positive and negative examples. Researchers believe that elaborative interrogation works because it supports a learner integrating new information with previously learned material, and helps the learner discriminate similarities and differences, and encode this in their learning. [36]

2.1.4. Types of Questions and Effectiveness

Researchers have also compared types of questions to determine if test format affects learning outcomes. Both short-answer and multiple-choice tests have been shown to have positive effects on retention but a study by Duchastel [35] showed better retention on a post test given after an interval of two weeks for students previously given short answer questions. Later studies such as Butler and Roediger [19] and McDaniel et al. [81], have confirmed that short answer questions improve learning outcomes more than multiple choice questions.
2.2. Taxonomy of Questions

Developing a taxonomy of question types can be helpful for classifying generated questions and quantifying the relative portions of questions at each level. Taxonomies of questions generally assume that some questions are deeper than others. This is an intuitive but somewhat subjective concept. How do we know if a question is deep as opposed to shallow? For example, a question may appear to be deep if it asks about cause and effect, but suppose that a student has just read a sentence explaining the cause and effect. Some may consider this to be a shallow recall question in that case. However the prior discussion of self-explanation indicates that by answering such a question, even one closely tied to the source text, the student is required to engage in a cognitive process that will help them encode this cause-and-effect concept if they understand it, or equally important, help them realize that their understanding is not sufficient on this concept.

In discussing taxonomies, we are typically talking about this elusive concept of question depth, which is unrelated to the lexical form of a question. A what question could appear at various levels on most taxonomies, it is not necessarily a low-level question. Similarly, a why question can appear at different levels and is not necessarily a higher-level question merely because of the word why. Question depth is also unrelated to question type (multiple choice, cloze, etc.) because each of these types could form questions at various levels of a taxonomy. Classifying question depth is an attempt to identify the cognitive load of a question.

2.2.1. Bloom’s Taxonomy

Perhaps the most well-known taxonomy is Bloom’s taxonomy. Beginning in 1949, Benjamin Bloom worked on developing the taxonomy in the hopes that it would facilitate sharing of test items among faculty so that banks of items could be built based on different educational objectives. The taxonomy had 6 categories: knowledge, comprehension, application, analysis, synthesis, and evaluation. This organizes objectives from simple to complex, from concrete to abstract. In 2001, Anderson and Krathwohl revised the taxonomy to include a second dimension that specified the type of knowledge involved. These
are: factual knowledge, conceptual knowledge, procedural knowledge, and metacognitive knowledge [7, 59].

Classifying questions according to the original 6 levels can be challenging, especially discerning between remembering and understanding, or between analyzing and evaluating. Classifying with the added second dimension described above increases the complexity. Studies of people skilled in the taxonomy show that they have difficulty agreeing on the classification of items [31, 9].

Bloom’s taxonomy is not without its critics for reasons other than its complexity in implementation. Booker [15] makes the observation that the taxonomy was developed for university students but later was applied to K-12 education in the U.S. This has led to focus on pushing students towards advanced thinking at the expense of thoroughly grounding them in the knowledge needed to support higher cognitive levels. Oddly, although the taxonomy is universally accepted, it has never been validated experimentally. Cox and Wildeman surveyed 118 studies on Bloom’s taxonomy and could find no experimental evidence supporting the taxonomy. Booker also observes that some K-12 educators like to encourage “Socratic thinking” without awareness that Plato wrote that higher order thinking could not start until the student had mastered conventional wisdom.

Perhaps the most important criticism of Bloom’s is that it is an inferential construct: it infers what is going on in a student’s head which obviously cannot be directly observed [38]. Bloom himself [14] acknowledged that it was not always possible to know whether a student answered a question using a higher-order cognitive reasoning process or by a lower-order recall process.

2.2.2. Lehnert-inspired Taxonomies

More recently proposed question taxonomies [88, 44] are similar to the question categories Wendy Lehnert identified for her question answering system, which in turn were influenced by the conceptual dependency theories of Schank [65]. Schank’s conceptual dependency, CD, was developed in the early days of AI as a representational theory for NLU (natural language understanding). CD was capable of representing inferences that could
be made about physical actions. The hope of CD was to construct a representation of the conceptual basis of all natural language. Over time it became apparent that a set of representational elements and primitive actions connecting them could not scale up to some kind of simulation of human understanding [73]. Lehnert required that her taxonomy be able to predict the kinds of memory searches needed to answer a question. The algorithm for automatically determining the question type is essentially a decision tree that first splits on whether or not there is a casual chain, that is two ideas connected in some manner. Lehnert’s 13 question categories were: causal antecedent, goal orientation, enablement, causal consequent, verification, disjunctive, instrumental/procedural, concept completion, expectational, judgmental, quantification, feature specifications, and request [65].

Note that there is no context awareness in Lehnert’s taxonomy, which makes sense because it was designed for answering questions not asking them. It stands to reason that a taxonomy developed for a question-answering system would not consider context because users typically ask questions randomly. However, in an educational application, the context of questions is important in classifying them. Take the question What are properties of a bar graph? which is classified as a feature specification question in one of the Lehnert-inspired taxonomies [44]. Imagine scenario 1, in which a student reads a description of the features of bar graphs, and scenario 2, in which a student is shown a series of bar graphs with perhaps some textual explanation of what the graphs show. In scenario 1 the student is being asked to recall information previously read whereas in scenario 2 the student is being asked to discover information not previously read. Clearly these two scenarios are not at the same cognitive level, but Lehnert-inspired taxonomies would classify them at the same level.

A further limitation of these taxonomies is that they cover only a few educational objectives, namely to assess students’ recall and develop their critical thinking skills. Gall [38] suggests that student answers provide clues to the cognitive depth of questions. While answers to factual questions can be evaluated by comparison to the source text, answers to higher-cognitive questions should be more complex and original, and use facts to support the response. Gall goes on to suggest that factual instruction could best be accomplished
through technologies that do not require teacher intervention, thus freeing class discussion
time for higher-level discussions [38]. This last suggestion is in keeping with Bloom’s admo-
nition that higher-order reasoning is built on a foundation of lower-level content knowledge
[14]. Sternberg and Grigorenko noted that teachers need to abandon the false notion of a
dichotomy between teaching critical thinking and teaching facts because the former rests
upon the later: “One cannot apply what one knows in a practical manner if one does not
know anything to apply.” [106]

2.2.3. MARGE Taxonomy

The desired qualities in a taxonomy for MARGE are: (1) it should distinguish between
low-order recall questions and higher-order questions that require a student to apply their
factual knowledge in creating some conceptual, cohesive answer, and (2) it should be easy to
implement, meaning that it will not rely on inference about what is going on in a student’s
mind, and (3) it assumes no prior knowledge of the topic by the student. In Table 2.1, a
two-level taxonomy is presented that divides questions into factual comprehension questions
and conceptual comprehension questions.

<table>
<thead>
<tr>
<th>Level</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>Factual Comprehension</td>
<td>Recognition and recall of factual information encountered in the text.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The answer will be found in one sentence in the source text.</td>
</tr>
<tr>
<td>2.0</td>
<td>Conceptual Comprehension</td>
<td>Synthesis of factual information encountered in the text.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The answer must be constructed from multiple sentences in the text.</td>
</tr>
</tbody>
</table>

Table 2.1. MARGE Question Taxonomy

What percentage of questions should be in each of the levels? It is beyond the scope of
this research to recommend target percentages for each group. Given the above discussion,
however, it seems that an educational application should question students to a certain
threshold of accuracy on level 1 before moving on to a few questions from level 2. The
higher-level questions should be fewer in number because they require much more time from
students in thinking about their responses and organizing them in written form.
The advantage of this taxonomy is that it explicitly considers the educational context of a question: Has the student read this information or must he construct this knowledge independently?

2.3. Question Levels in Practice

There is often a disconnect between educational research and what is happening in classrooms. Teachers and administrators often don’t have access to the latest research either in practice or in their formal training. Further, they have been led into a false choice between teaching for facts and teaching for thinking [99]. Studies discussed below indicate that, historically at least, teachers asked overwhelmingly factual questions. On the other hand, some teachers advocate an avoidance of factual questions entirely [15]. Where is the balance?

No one wants to see school children mindlessly memorizing facts and dates from a history book, as was common practice in the high school I attended. We want students to see the “big picture,” to understand the causes and effects of historical events, and be able place events in larger contexts of historical movements and eras. However, the pendulum has swung too far away from facts when it is relatively easy for interviewers on a college campus to find students who do not know who won the civil war, what the opposing sides were, or say things like you mean the civil war in 1965? ¹

This is a concern that spans domains. Booker, a philosophy professor, observes that his fellow community college instructors rarely complain about the intelligence of their students but rather their readiness for college work. Booker does not expect recent high school graduates to know anything about philosophy, but it would be nice, he notes, if they had some knowledge of world history, science, English and basic math [15].

2.3.1. Questions in the Classroom

Studies over the past 100 years indicate that teachers ask a lot of questions, and historically, the questions they asked were overwhelmingly factual in nature. In 1912, Stevens

¹https://youtu.be/yRZZpk_9k8E
[107] estimated that 80% of school time was occupied by questioning. He sampled high
school teachers and found an average of 358 questions per school day. Moving forward
50 years, researchers found that teachers were still asking a lot of questions: elementary
science teachers asked an average of 180 questions in a science lesson, and elementary social
studies teachers asked an average 128 questions per hour [38]. Moving forward to more
recent research, Graesser et al. [45] found that 96% of the questions in classrooms came
from teachers, and only 4% were higher order questions. The most distressing thing about
this research, however is not that teachers ask a lot of factual questions. It is that teachers
don’t wait for an answer, they typically wait less than one second for student response, and
then answer the question themselves [101], thus entirely depriving students of the cognitive
exercise.

Let’s consider some possible explanations for these findings. Teachers may be reluctant
to ask higher-order questions when they are acutely aware that their students don’t yet
have a basic command of the factual information. Teachers face a class full of students with
a wide range of skills and knowledge. Leading a high-level conceptual discussion might be
engaging for top students but might leave others behind. What is a teacher to do? As Gall
[38] suggested, the only way to free a teacher to lead a higher level discussion is to first have
students engage with technology that teaches, assesses and reteaches the factual foundation
of the subject under discussion.

2.4. Conclusions

The research cited above has consistently shown that pausing students in the learning
process to ask them questions improves learning outcomes. Automatically generated
questions have been shown to be as effective as human-authored ones in studies dating from
the late 1970s [111] to 2015 [52]. Further, selecting among automatically generated ques-
tions saves significant amounts of time compared to manually generating them [48]. The
above discussion also included research that specifically supports the efficacy of short answer
questions such as those generated by MARGE.

Knowing that questions, including automatically generated ones, have educational
efficacy, the next question is: *what kinds of questions should MARGE generate?* The overwhelming majority of prior QG systems generate questions from sentences. These are questions at the first level of the proposed taxonomy: Factual Comprehension. They are questions that support learning foundational knowledge which is a prerequisite to mastering higher levels of understanding, as discussed in this chapter. As noted by the National Academy of Sciences:

> Experts, regardless of the field, always draw on a richly structured information base; they are not just “good thinkers” or “smart people” The ability to plan a task, to notice patterns, to generate reasonable arguments and explanations, and to draw analogies to other problems are all more closely intertwined with factual knowledge than was once believed. \[17\]

Factual questions are important and could have a place, as Gall suggested, in building basic knowledge of all students prior to higher level class activities and discussions. MARGE should generate these kinds of questions, but can MARGE generate higher-level questions? The proposed research in this dissertation includes the following generation goals for MARGE: (1) Improve the percentage of quality factual (level 1) questions generated by methods such as NLU analysis to identify important content, and using TextRank to select the most relevant questions, and (2) Go beyond the sentence barrier to ask conceptual questions via NLP techniques and algorithms discussed in later chapters. It is expected that the number of questions generated from level 2 will be less than those generated from level 1. The focus on this work is on the generation, not on the selection from available questions. It is also anticipated that answers to questions in level 2 will not be generated in some cases and in others only suggested possible answers could be generated.
CHAPTER 3

DESIGN DECISIONS IN PRIOR WORK

This chapter examines prior work in automatic question generation from an NLP software engineering perspective. In addition to providing a useful framework for exploring prior work in QG, it is hoped that this will also provide helpful background knowledge for future developers seeking to create QG systems. In the survey of prior work in this chapter, brief system descriptions are followed by samples of generated questions, and key results statistics if provided in the associated papers. By giving examples of generated questions, the systems will be allowed to speak for themselves about the types and quality of questions they are capable of generating.

3.1. Design Decisions

When constructing QG systems, designers should ask the following questions:

(1) What is the purpose of the generated questions?
(2) What types of questions should be generated?
(3) What input source is appropriate and available?
(4) What approach is optimal for generating these questions?

Sometimes design decisions are made, and then cannot be carried out, forcing the designer(s) to take another approach, as seen later in the story of Wolfe’s AUTOQUEST system.

Many of the systems described in this chapter are from QGSTEC2010, the Question Generation Shared Task Evaluation Challenge, the first, and to date, only STEC for QG. It should be noted that these systems were all required to use the same input source, which was a set of paragraphs from Wikipedia, OpenLearn, and Yahoo! Answers. While the paragraphs from Wikipedia and OpenLearn were well edited, the paragraphs from Yahoo! Answers varied greatly, which affected the quality of generated questions from that source.
3.1.1. Determine QG System Purpose

When developing a question generation system, the first design decision is to clarify the purpose of the generated questions. Automatically generated questions have a role in various types of dialog systems: chatbots, information retrieval systems, customer service systems, medical/psychological support systems, educational applications such as intelligent tutoring systems, and more. In this work the focus is on questions developed for educational purposes.

3.1.2. Determine Question Types

Once the purpose of the questions is clarified, it is important to determine what types of questions should be asked. Within an intelligent tutoring system or educational application, questions may fall into two categories: assessment questions, i.e., questions that probe a student’s understanding, and social coordination questions, such as: *Are you ready to continue?* [45] The focus in this work is on assessment questions.

Within the general category of assessment questions, there are many forms: cloze, fill-gap, multiple choice, open-ended, and others. Note that educators distinguish many types of assessment (diagnostic, formative, summative, etc.); however, since the generation method would be the same, we include them all under the term *assessment questions*. The design decision as to what type(s) of questions to generate can be influenced by many factors, such as the medium in which they will be delivered. If the questions are generated for classroom response systems (clickers) then multiple choice questions are the obvious choice. For fast, accurate feedback on student responses in a MOOC, multiple choice or fill gap questions may be preferred. For classroom discussion questions, such as for the Comprehension SEEDING system [93], constructed response questions are preferred. MARGE, the question generator described in Chapter 5, generates short answer questions. However, it is useful to explore generation techniques for other questions types since a generator designed to output one type could in many cases be modified to output additional types of questions.

There are numerous ways to classify questions. Chapter 2 provided a survey of question classification schemes. One common division is based on what is required of the student.
Supply type questions ask the student to supply a word or short answer. Selection type questions ask the student to select correct answers from available options [68]. This survey of question generation systems finds that both types are common. In Chapter 2 research was presented that indicated that supply type questions have more pedagogical utility, but this does not imply that selection type questions have no value. Here we are interested in the varieties of the physical form of the question and how they are created. Sometimes these various forms of assessment questions are given different names in different texts in the literature. Table 3.1 at the end of this chapter lists forms of questions observed in relevant question generation systems.

In cloze questions, a portion of text has words or phrases deleted which the student must fill. The term cloze is a spoken abbreviation of closure, inspired by Gestalt psychology. Cloze questions may be generated by the rational method, in which content words and cohesion devices are deleted, or the semi-random deletion method, such as deleting every 7th word [83]. Gap-fill questions look similar to cloze questions, and these terms are sometimes used as if there is no distinction between them. However, other sources note that the gap-fill will have one key term or phrase removed, whereas the cloze generally has more than one blank to fill. Sometimes possible answers are provided in a word bank in which case it would be a select-type question, rather than a supply-type question. Another supply-type variation is to have multiple choice options for the gap. Other question types not seen in the surveyed QG systems include matching, ordering, and performance task items.

3.1.3. Identify an Input Source

In creating a question generation system, some type of content must be input to the system. The input could be in the form of plain text, structured text, or even a knowledge base. The form of that input may be determined by the purpose of the question generation system. For example, a question generator specifically designed for the 2010 Question Generation Challenge [16] was required to read predefined input in the form of an xml file.

1https://www.teachingenglish.org.uk/article/test-question-types
Questions generated for many educational applications may have pre-stored text which students read. This would be an appropriate input source for generating questions for those learners. If a system wants to support student learning from open domain texts, then the system should be able to read in text from various sources, perhaps after some preprocessing. Although the input form of the source material is relevant mainly for a preprocessing function within the question generator, the choice of input can affect the types of questions that can be asked. For example, one passage from the QGSTEC2010 data set \(^2\) was a Yahoo! Answers article about what to feed a pet duck. It is unlikely that many deep-reasoning questions could be generated from this input source.

3.1.4. Select an Approach

Question generators are transformers: they take input and transform it to questions for output. The QG systems examined in this chapter transform input text into questions by looking for predefined patterns in the source material (source patterns) and using transformation patterns and heuristics to create questions. In some systems these transformation patterns are referred to as templates. How is this transformation accomplished? There are many design decisions within the approach decision. These include, but are not limited to:

- What parsing software should be used?
- What source patterns can be detected in the output of the parsing software and/or the raw text?
- Should the transformation patterns be specified in internal rules or external templates?
- What other tools and resources could help: named entity recognizers, coreference resolution software, gazetteers, and more?
- Should there be some restrictions on generation to prevent unacceptable questions?

The majority of the following systems start with a text source. Then, the two main decisions will be (1) selecting parsing software, and (2) deciding whether to use external

\(^2\)http://www.questiongeneration.org/QGSTEC2010
templates or internal rules for sentence-to-question transformation. In a recent survey of
question generation approaches for educational applications, Le et al. [64] observed that
template-based approaches tended to perform better than systems that syntactically rear-
ranged the source text. Our observation is that generating any question type is theoretically
possible in any approach, but that some approaches make some question types easier to
generate than others.

3.1.5. Wolfe’s AUTOQUEST

This chapter explores prior work in question generation, with an eye to design deci-
sions which can also be seen as ways to categorize QG systems. Before delving into recent
work in question generation, it is worth noting that a viable system was developed over 40
years ago.

In the mid 1970s, working for the Navy Personnel Research and Development Center,
John H. Wolfe created the first documented automatic question generation system, which
he called AUTOQUEST [110]. Wolfe’s original intention was to process sentences with a
syntax parser, then transform the parsed output into questions. The state-of-the-art parser
available at the University of California at Irvine at the time occupied so much memory that
it could only be run after midnight. Wolfe found that 50% of the questions failed to parse
due to insufficient memory. Of those that parsed, only 60% were parsed correctly. Before
abandoning this approach, Wolfe ran a failed sentence on the original version of the parser,
available at BBN (Bolt, Beranek, and Newman) which also failed due to insufficient memory
after processing for an hour. Realizing the infeasibility of this approach, Wolfe undertook an
approach that looked for source patterns in the raw text itself. Each sentence was matched
against a table of pre-stored source patterns. When a sentence matched a source pattern,
a question was generated using transformation patterns and heuristics. For example, one of
the sentence patterns was: S1 so that S2. A sample matching sentence, and the question
and answer generated from it, follow:

Source sentence: The dd name identifies a DD statement so that
subsequent control statements and the data control blocking the
Question: Why does the dd name identify a DD statement?

Answer: So that subsequent control statements and the data control block in the processing program can refer to it.

The AUTOQUEST system matched approximately 20 different sentence patterns. These patterns were developed based on manually exploring source texts and manually authoring questions. After generating questions, AUTOQUEST checked and rejected questions for certain conditions associated with low quality such as prevalence of pronouns, lengthy questions, and so forth.

Wolfe conducted experiments and evaluations on AUTOQUEST, described in Chapter 6. Here, it will just be briefly noted that AUTOQUEST generated 68% grammatically acceptable questions, a number that many recently published systems could envy. Despite the project’s initial setbacks with the parser, Wolfe was able to complete his system and answer affirmatively his research question on whether it was possible to develop a system to automatically generate question from text.

3.2. Recent Work in QG

The past decade has witnessed a renaissance in the field of automatic question generation. Evidence is in the growth in both the number and diversity of recent approaches. Despite the early, promising work demonstrated in AUTOQUEST, the field of automatic generation from text appears to have been relatively quiet in the closing decades of the 20th Century. In fact, up until the early 2000s and beyond, computer-based instructional systems continued to use frame-based methods in which all course content, including questions, was authored by hand [112]. The recent resurgence of interest in automatic question generation is motivated in part by the evolution of intelligent tutoring and computer-assisted learning systems, and the need for more rapid development of questions for these systems.

Rather than provide a historical narrative of the advancement of the state of the art in question generation, this section will group the top recent systems by the 4th design decision, approach, discussed earlier. All of systems explored below have a similar purpose:
to support student learning. Pedagogical issues were discussed in Chapter 2, but note that all of these systems were motivated by a belief in the educational value of questions during the learning process, and many were designed as a key component of an educational application.

This section provides an analysis of recent work, selected based on two factors: (1) unique contributions to the state of the art, and (2) influence as indicated by citation count and/or publication venue. Table 3.2 lists the top systems selected based on these criteria, in chronological order. For each QG system, the table lists the year of the latest paper on the system, authors, approach and types of questions generated.

3.2.1. Design Approach: WordNet

WordNet, a lexical database of the English language, was created in 1985 under the direction of George Miller at Princeton University [85]. For nouns, verbs, adjectives and adverbs, WordNet provides synsets (synonym sets), which are words (and collocations) from the same lexical category that are roughly synonymous. The meaning of a synset is explained with a brief definition and examples of usage are provided. These synsets are arranged in a hierarchy so that a word may be linked to other words, if known, for the following relations: hypernym (what this word is a type of), hyponyms (words that are a type of this word), coordinate terms that share a common hypernym with this word, meronyms (words that are part-of this word), holonyms (words that this word is a part-of), troponym (manner of doing a verb), entailment (a verb implied in this verb). WordNet encodes sophisticated word knowledge, and soon researchers interested in educational applications began to exploit WordNet for question generation.

3.2.1.1. Aist: 2001

Aist [4] used WordNet to assist children with vocabulary acquisition, by creating a limited few types of factoid questions. These questions were constructed to help children learn the meaning of words by comparison to other words. The questions were incorporated into the Project LISTEN reading tutor at Carnegie Mellon University. For example, when a learner encountered the word *astronaut*, this factoid was generated: *astronaut can be a*
kind of traveler. Is it here? Their research indicated that these little facts about vocabulary words, the factoid questions, helped students learn the meaning of rare words.

As an aside, the term factoid\(^3\) has been used in the QG literature in slightly contradictory ways. The usage described above in Aist [4] to indicate small bits of factual information seems to be the most appropriate use. Later QG system designers, in critiquing other QG systems, use the term factoid in a pejorative sense, as if any questions related to factual information cannot be valuable. This misconception was addressed in Chapter 2.

3.2.1.2. Brown, Frishkoff, and Eskenazi: 2005

Brown et al. [18] extended this approach for the REAP Project (also at CMU), a system designed to retrieve level-appropriate material for students learning to read. Given a target vocabulary word, the system could produce 6 types of questions: definition, synonym, antonym, hypernym, hyponym, and recalling the target word. These 6 types of questions were produced primarily in two forms: multiple choice and cloze. Distractors were chosen from WordNet that had the same POS and similar frequency to the word that is the correct answer. The distracters were chosen randomly from the 20 top words that fit the criteria, giving preference to words found in the source text. Additional tools used were a word frequency database, and a POS tagger. An example of a multiple choice question which prompts the student to retrieve the target word follows.

Choose the word that best completes the phrase below:
the child’s misery would move even the most _____ heart
A) torpid
B) invidious
C) stolid
D) obdurate

WordNet continues to be used in QG systems, not as the foundation tool as in the above two systems, but as one of many NLP resources at a developer’s disposal.

\(^3\)The term factoid was coined by American author Normal Mailer in his 1974 biography of Marilyn Monroe by combining the word fact with the suffix -oid, meaning similar but not the same. He coined this word to refer to spurious information but over time, factoid has come to also mean trivial bits of factual information or news. https://en.wikipedia.org/wiki/Factoid
3.2.2. Design Approach: QG from Key Terms

The QG systems examined next identify key terms in text and then generate questions based on those key terms.

3.2.2.1. Agarwal and Mannem: 2011

Agarwal and Mannem [2] developed a system to create gap-fill questions from source text, which in this work was two chapters from a Biology textbook. Their questions had one key word or phrase removed with multiple-choice options for filling the gap. The first step in their approach is to identify informative sentences in the text using techniques from summarization. Sentences were scored with features such as number of tokens in common with the title, presence of superlatives, number of nouns and pronouns, etc., with weights on features determined empirically. Then a question is generated by removing content, the key, and finding appropriate distractors for the key from the text. Candidate key selection begins with identifying noun chunks, reduced to one of two head words, that are ranked as important in the document. The use of the source text for distractor generation is in contrast to the above approaches with used an external source, WordNet, for distractor generation.

Sample questions are shown below.

An electron having a certain discrete amount of ________ is something like a ball on a staircase.
(a) charge (b) energy (c) mass (d) water

Lipids are the class of large biological molecules that does not include ________.
(a) acid (b) polymer (c) glucose (d) know

According to two human evaluators, 76% of the generated questions were useful, meaning that they had at least one good distractor. The first sample question above has 3 good distractors whereas the second one only has 2 good distractors. The authors observed that if they key contained more than one word, the distractor quality tended to go down. A larger corpus might give more options for distractor selection. Further, POS-tagging and/or syntax processing could improve distractor quality.
3.2.2. Becker, Basu and Vanderwende: 2012

Becker et al. [11] take an approach similar to Agarwal and Mannem, above, but train a logistic regression classifier to select the gaps. Their classifier was able to select appropriate gaps compared to human annotations, with an 0.83 true positive rate. In future work they plan to train a separate classifier for sentence selection. The form of question generated was the cloze type. Examples are shown below.

**Question:** Caesar then pursued Pompey to Egypt, where Pompey was soon ______.  
**Answer:** murdered

**Question:** About 7.5% of world sea trade is carried via the canal _____.  
**Answer:** today

3.2.3. Design Approach: QG from Ontologies

Going beyond identifying key terms, some QG systems first create an ontology based on the source text, then generate questions from that ontology. The word *ontology* was borrowed from philosophy but redefined by early AI researchers and continues to be reinterpreted within Computer Science literature [69]. Although it remains variously and at times loosely defined, a general description of an ontology is a set of entities that model domain knowledge, along with relationships between these entities.

3.2.3.1. Olney, Graesser, and Person: 2012

In Olney et al. [92], the authors state that the primary goal of their work is to bridge the computational-psychological gap, in other words, to find computational methods of question generation that align with psychological models of learning. Their approach first automatically creates conceptual graphs for the domain, in this case Biology. Then questions are created from the graphs. The concept map they created from a Biology textbook has key terms and the relationships between key terms, or properties of key terms, are specified in a triple of the form: $x$ relation $y$. An example is: *abdomen is-a part*. The top 5 edge relations were: has-property, has-consequence, has-part, location, and is-a.

Question templates specify what questions can be generated from the various relation types. For example, for the has-consequence relation, a question template is *What do KT*
do?, where KT is the key term and the first do could be replaced by do, does, did as appropriate. Since these questions are designed to be used in an ITS, the question templates have varying degrees of specificity, such as pumps, hints, and prompts. An example of a pump is: Can you say more? whereas a hint example is: What’s going on with friction here? and a prompt example is: What’s the force resisting the sliding motion of surfaces? The following are sample hint templates, where KT will be replaced by the key term.

KT, what is that?
What do KT do?
What do KT have?

3.2.3.2. Afzal and Mitkov: 2014

Afzal and Mitkov [1] extract key Biology concepts and the relations between them using IE methodologies. The system identifies potential key terms by extracting named entity NPs that are closely linked in a dependency parse to the main verb of a sentence. The NEs are ranked according to their importance in the domain corpus. Questions are generated by simplifying the source sentence, and replacing the NE with the appropriate wh-word. The system generates multiple choice questions, and distractors are created by distributional similarity measures over the GENIA corpus. An example follows.

Source sentence: The predicted periplasmic domain of the PhoQ protein contained a markedly anionic domain that could interact with cationic proteins and that could be responsible for resistance to defensin.

Generated question:
- Which protein of the PhoQ protein contained a markedly anionic domain?

The above example is from their paper. Although this question may be acceptable to a domain expert, to the non-expert, a question perhaps closer to the meaning of the original sentence could be: What domain of the PhoQ protein contained a markedly anionic domain?

3.2.3.3. Jouault and Seta: 2014

Jouault and Seta [55] created a QG system for the history domain that operates in an open learning space such as Wikipedia or other sources. In the learning framework,

\[\text{http://www.ncbi.nlm.nih.gov/pubmed/12855455}\]
a student begins with content, such as information from Wikipedia on World War I. The student builds a chronological concept map with causal links between events. Meanwhile the system builds its own map using additional content from Freebase and DBpedia so that it can have additional information to guide the student in their independent study. Questions are generated by comparing the system concept map to the student map. The system uses question types defined by Graesser et al. [42]. Sample questions are shown below.

- What were the consequences of World War I on Austria-Hungary?
- Did the Siberian Intervention change the course of World War I?
- Would World War I have been different without Ferdinand Foch?
- Did the First Battle of the Marne change the course of World War I?
- How was the German Papiermark used during World War I?

3.2.3.4. Chaudhri, Clark, Overholtzer and Spaulding: 2014

Chaudhri et al. [25] developed a QG system to be used with a learning system composed of a curated knowledge base, an intelligent textbook, and a reasoning system. If a user types in a query to a KB the system may not understand what the user needs. Therefore, the system can respond with hopefully relevant questions that it can answer from the KB. Example questions based on a user query are shown below. Similar questions can be displayed within the intelligent textbook. Questions are formed from question templates which use relations from the KB.

Questions are generated by crawling the KB and storing all generated questions in a database, which is used at run time to select questions. There are 20K questions in the database, so only the top-ranked questions are presented to the learner. Questions are ranked by (1) the importance of that question within the set of all questions on that concept, and (2) the rank of that question within all questions of that type that cover that concept. Sample questions for a query are shown below.

User query
- explain the structure of chloroplast?

Generated questions that are answerable from the KB:
- What are the differences between a cell and a chloroplast?
- What is the shape of a chloroplast?
- What is the function of a chloroplast?
- What are the differences between an amyloplast and a chloroplast?
- What is the structure of a chloroplast?

Three of the four systems discussed in this section cover the Biology domain. Some domains may lend themselves more readily to ontology extraction than others, making it unclear to what extent the ontology-based approach could be domain independent.

3.2.4. Design Approach: PSG Parser and Templates

In contrast to the above approaches which identify words, key terms, or relations between key terms as a basis for QG, most QG systems examine parsed source text to find patterns suitable for question generation. The most commonly used parser is a PSG (phrase structure grammar) parser which creates a hierarchical representation of phrases in a sentence. This section looks at QG systems which combine a PSG parser and templates, and the next section will look at QG systems which combine a PSG parser with internally coded transformation rules.

3.2.4.1. Rus, Cai, and Graesser: 2007

In 2007, Rus et al. [102] saw automatic question generation as a means to reduce the amount of manual authoring for ITS content. Motivated in part by the observation [115] that humans use pattern matching and mapping rules when they generate sentences, they sought to engineer a similar approach in their question generator. Building on earlier NLP work for chatbots, they modified the AIML (artificial intelligence markup language) to enable authoring categories of questions. Each category contained a pattern that would be matched against the sentence and a template used for generating that sentence. The following is a sample category:

```
<category>
<pattern>
<NP>_np_</NP>
</pattern>
<template>
```
The source pattern in this case is a noun phrase. Source text was parsed using the Charniak parser [23, 24]. Once a noun phrase is found in the parse of the text, it can be placed in the template. The following shows three sample questions generated from the source sentence. These questions are created by instantiating \textit{What can you say about \_np\_?} with the three NPs identified in the source sentence.

Source sentence: There are no horizontal forces on the packet after release.  
Question 1: What can you say about horizontal forces?  
Question 2: What can you say about the packet?  
Question 3: What can you say about the horizontal forces on the packet?

The question generation system had over 150 categories such as this to generate \textit{who, what, where, when, how,} and other questions. Approximately 60\% of the questions were acceptable. These categories were created to automatically generate questions for the AutoTutor [43] system, discussed below.

Rus et al. also discussed the benefits of using templates in a QG system, which essentially decouples the generation engine from the pattern authoring. First, it abstracts away the generation engine from the authoring process. This makes it easier to have pedagogical and linguistic experts assist with the pattern and template authoring. Their QG system was domain independent, but they note that moving to a new, specialized domain would not require modifications of the generation engine.

3.2.4.2. Wyse and Piwek: 2009

Wyse and Piwek [113] created a proof-of-concept system that followed a similar approach to Rus et al. [102]. For an input source they used xml files from OpenLearn units\footnote{http://www.open.edu/openlearn/}.

The QG system used the Stanford parser for POS and PSG parsing, Stanford tool Tregex to identify patterns of interest in source sentences, and templates to transform sentences into questions. Answers were also generated. Interestingly, the system included a visualization
of the PSG tree in order to assist QG developers with creating templates. The authors developed an xml format in which to store rules, and propose the development of a standard format for representation of rules. However, this standard representation would severely limit QG systems creativity in exploring different NLP tools and approaches. It is not currently feasible to create a set of rules that would match the output of different types of parsers, and no good reason to limit developers in their choice of software tools. A sample generated question from the paper follows. No evaluation was done on the generated questions.

Source sentence: Emmanuel-Joseph Sieyès (1748-1836) trained as a priest and became assistant to a bishop.

Question: What did Emmanuel-Joseph Sieyès train as?
Answer: a priest

3.2.4.3. Liu, Calvo and Rus: 2010

The Rus et al. system above was a general-purpose question generator. Liu, Calvo and Rus [70, 71] later used a similar approach to develop a special-purpose generator to support students in academic writing. The system, called G-Asks, first extracts the citations from a student’s paper along with related text. The extracted citations are classified by a rules-based approach and questions are generated using templates and the extracted content.

The domain is supporting students in the task of academic writing, and developing skills such as sourcing and information integration. Generic questions such as Have you identified the research methods used in the literature reviewed? have proven to be helpful, but generating more specific questions is the goal of this work.

After citations and source text are extracted from the student’s paper, the extracted text is parsed with the Stanford parser. Other tools used are the LBJ NER Tagger, to detect author’s names, and SENTIWORDNET to detect opinion words. Based on the classification of the citation, different templates will be used to generate questions. Sample questions are shown below.

Opinion category:
Why did Cannon challenge this view mentioning that physiological changes were not sufficient to discriminate
emotions? (What evidence is provided by Cannon to prove the opinion?) Does any other scholar agree or disagree with Cannon?

Result category:
Does Davis objectively show that this classification accuracy gets higher from about 70% up to 98% while actors express emotions and computers perform the...? (How accurate and valid are the measurements?) How does it relate to your research question?

System category:
In the study of Macdonald, why does workbench tool provide feedback on spelling, style and diction by analyzing English prose and suggesting possible improvements? What are the strength and limitations of the system? Does it relate to your research question?

A pilot study showed that students found the automatically generated questions as helpful as human-authored ones, and that evaluators found it moderately difficult to distinguish the two groups of questions.

3.2.5. Design Approach: PSG Parser and Transformation Rules

Perhaps the most common approach to automatic question generation is the PSG-Transformation approach, in which sentences are parsed by a phrase structure grammar parser and then converted into questions by transformation rules.

3.2.5.1. Gates: 2008

The first known QG system to use the PSG-Transformation approach is Gates [40, 39]. Gates was interested in supporting children’s reading via a look-back strategy in which readers were to point back to the location in a text where the answer to a question is found. The system she developed\(^6\) has children “answer” the questions by clicking on the sentence containing the answer. Gates used a children’s news corpus as input, which was parsed using the Stanford parser. The Stanford tools Tregex and Tsurgeon were used to identify specific patterns in the parsed sentences and manipulate their structure, respectively.

\(^6\)Sample questions can be seen at: http://www.cs.cmu.edu/~dmg/MCALL/mcall.html
Transformation rules were manually constructed. The system generated wh-questions that were 81% acceptable according to a human judge.

3.2.5.2. Heilman and Smith: 2009, 2010

Although not the first to use this approach, the best-known and arguably most complete question generation system using this approach is the Heilman and Smith system [49, 50, 48]. The Heilman and Smith system takes an approach very similar to Gates, but adds two important elements. The first is a sentence simplification stage that will allow the system to handle more complex sentences than the children’s news corpus used as source text by Gates. The second addition to the approach, and the more significant, is their development of a statistical ranking module to identify better quality questions. This is especially important in an approach such as this that overgenerates, and Heilman and Smith were the first to create such a ranking component in a QG system.

Heilman notes in his dissertation [48] that the purely syntactic approach does not allow higher-level abstractions that may be possible with more semantically informed approaches. Nevertheless, as the QG system demonstrates, it is a robust and productive method for generating fact-based questions. The authors have made this question generator available for download.\footnote{http://www.ark.cs.cmu.edu/mheilman/questions/}

Source sentence: Widespread potato blight caused by P. infestans precipitated the well-known Irish potato famine in the nineteenth century.

Generated question:
- What did widespread potato blight caused by P. infestans precipitate?

3.2.5.3. Kalady, Elikkottil and Das: 2010

Kalady et al. [40, 39] utilized two approaches to question generation. The first approach is similar to the Heilman and Smith approach. The second question generation module in their system processed the document to identify key words and phrases, called \textit{Up-Keys}, using methods they had previously devised in document summarization. This module generated definitional questions of the form: \textit{Who/What/Where is Up-Key?} The
identification of the Up-Keys also enables Kalady et al. to give more importance to questions for which the answer is an Up-Key.

The questions are similar to the Heilman and Smith system. The following shows a sample source sentence and questions generated from the Kalady et al. system. These questions are directly from their paper [58]. Following their generated questions, the questions generated by inputting the same sentence into the Heilman and Smith system are listed for comparison.

Source sentence: Mexico City, the biggest city in the world, has many interesting archaeological sites.

Questions generated by the Kalady et al. system:
- Which/Where has many archaeological sites?
- What does Mexico City, the biggest city in the world have?
- In what does Mexico City, the biggest city, have many archaeological sites?
- Does Mexico City, the biggest city in the world, have many archaeological sites?

Questions generated by the Heilman and Smith System:
- What is the biggest city in the world?
- What has many interesting archaeological sites?
- What does Mexico City have?
- What is Mexico City?
- Does Mexico City have many interesting archaeological sites?
- Is Mexico City the biggest city in the world?

The Heilman and Smith system questions are placed in their ranked order, from high to low. The Kalady et al. questions are placed above in an order to match the Heilman and Smith questions, as much as possible. We note a few observations about the above questions:

- The Heilman and Smith system produced 6 grammatically correct questions; the Kalady et al. system produced 4 questions, only 2 of which appear to be grammatical.
- Both systems overgenerate; that is, they appear to generate as many questions as possible.
- Many QG systems generate yes/no questions, but the answer always seems to be yes.
Yes/no questions are the easiest to generate by performing simple subject-auxiliary inversion and supplying the appropriate auxiliary.

- Source sentences with the verb *have* have a tendency to generate vague questions, as above: *What does Mexico City have?*. These sentences also tend to generate bad questions because of the many uses of *have* in light verb constructions and idiomatic speech. Some systems [75, 66] filter questions generated from source sentences with verbs having multiple functions in English such as do, be, and have.

3.2.5.4. Ali, Chali and Hasan: 2010

Another work in this category is Ali et al. [5]. It is similar to the Heilman and Smith system in that it uses a sentence simplification step, followed by a PSG parse and transformation rules to generate questions. What is unique in the Ali et al. system is that they include an additional sentence classification step which assists them in determining what type of question can be generated from the sentence. The classification step first identifies the subject, verb, object and prepositional phrases as well as their entity type, based on the PSG parse and a Named Entity parse. They used the Oak (http://nlp.cs.nyu.edu/oak/) parser which provides 150 NE types, organized into 5 categories: human, entity, location, time, and count. Then a set of interaction rules were developed to specify what types of question(s) could be generated from different permutations of SVO-NE types. For example, from a sentence with a human subject and a human object, a *who verbed whom?* type of question. The system had 90 interaction rules. An example from their paper includes the following source sentence and question. The source sentence fits the interaction rule: human verb entity time.

**Source sentence:** Tom ate an orange at 7 pm.

**Interaction rule:** human verb entity time

**Generated questions:**
- Who ate an orange?
- Who ate an orange at 7 pm?
- What did Tom eat?
- When did Tom eat an orange?
Although the above example provided in their paper is a bit simplistic, it seems that this approach has a unique perspective in question generation, namely, that it is important to know what kinds of things we are dealing with in the source sentence. Prior QG systems were concerned primarily with either identifying phrases (NPs, PPs, etc.) or constituents (subjects, objects, etc.). In contrast, the Ali et al. approach seeks to first know what kind of things instantiate these phrases/constituents. This could be an important precondition to integrating common sense or domain-specific knowledge into question generation. Using the previous sample sentence, knowing that Mexico City is a city could transform the generated question from: *What has many interesting archaeological sites?* to *What city has many interesting archaeological sites?* which is less vague.

3.2.6. Design Approach: SRL parsing and Transformation Rules

An alternative to the PSG parse is the Predicate-Argument parse which identifies for each predicate in a sentence the arguments and modifiers associated with the predicate, and specifies their semantic roles. This parse is also known as the SRL (Semantic Role Label Parse) and is also sometimes called a shallow semantic parse.

3.2.6.1. Mannem, Prasad and Joshi: 2010

The QGSTEC2010 entry by Mannem et al. [74] was the only system that entered Task A - Generating Questions from Paragraphs, and appears to be the first system to use the SRL parse as the foundation for a QG system. Their system had three stages: content selection, question formation, and ranking. In the content selection stage, the SRL parse for a paragraph is input, identifying the mandatory arguments (A0 - A5) and select optional arguments (MNR, PNC, CAU, TMP, LOC, DIS) as potential targets for QG.

After potential targets are identified, a series of transformation heuristics are applied to the source content. These include: replacing the target with the appropriate wh-word (who, why, when, what, how, where), depending upon the semantic role and any named entity tags; modifying verb and auxiliary forms as needed; and moving optional arguments to the end of the formed question, as the authors state this makes for more well-formed
questions.

Discourse analysis is not performed on the paragraph in order to determine the most important content. Rather, the system generates all possible questions and then selects the top 6. Questions are ranked in two ways. First, questions generated from the main predicate of the sentence are given higher rank than those generated from other predicates in a sentence. Second, questions are ranked by the number of pronouns in the question since there was no coreference resolution.

The authors note that source text written in first or second person created problems for the parsing and other NLP tools since these were trained on the Penn TreeBank (WSJ corpus).

3.2.6.2. Chali and Hassan: 2015

Chali and Hassan [22] used SRL parsing and transformation rules along with additional features not previously used in QG systems. The QG system consists of 4 steps. First, sentence simplification is performed in a manner similar to Heilman and Smith. Second, they use the Illinois Named Entity Tagger. At this point they use a set of rules to generate questions without consideration of whether they are answerable from the text. For example, if Apple Inc. is tagged as an organization, the system will generate questions such as: Where is Apple Inc. located? The third step involves parsing the source text with an SRL system and use this to generate questions using manually created transformation rules. The types of generated questions are similar to those created in other QG systems [74, 66, 75]. The fourth step involves using Latent Dirichlet Allocation (LDA) to measure the importance of the generated questions. From the topics identified by LDA, the most frequent word tokens were chosen as subtopics. Once these subtopics are identified, Extended String Subsequence Kernel (ESSK) is applied to measure similarity of these subtopics with the generated questions. Finally, the system judges the syntactic correctness by parsing the question and source sentence with the Charniak parser and then calculating the similarity between the two corresponding trees using a tree kernel method.

They compared their system to the Heilman and Smith system. Specifically they
took the top 20% of the ranked questions generated by Heilman and Smith for comparison with their own questions. Syntactic correctness of their system showed a 3% improvement over the Heilman and Smith system. Using the metric we used in our comparison in 2014 of our question generator at the time [75] with the Heilman and Smith system, their reduction in the error rate was 6% compared to ours, which was 44% (see below). Oddly, their topic relevance showed only a 4% improvement over the Heilman and Smith system, which uses no relevance features in its generation system. The novelty of their approach (in Step 4) is appreciated, but one wonders if simpler methods could be employed to identify important questions, and with more impressive results. Later, this dissertation will describe techniques along these lines.

3.2.7. Design Approach: SRL parsing and Templates

This sections describes QG systems that use the SRL parse and templates to form questions.

3.2.7.1. Lindberg, Popowich, Nesbit and Winne: 2013

Lindberg et al. [66, 67] chose a template-based approach in order to be able to generate questions that are not merely syntactic transformations. Templates were manually constructed based on patterns detected in their source text: a corpus of 25 documents representing a high-school level unit on the environment. Templates matched numbered arguments, A0 - A5, of the SRL parse as well as AM-modifiers. The system did not generate answers. This, plus the domain-specific nature of their source text, allowed them to ask questions such as: **Summarize the influence of a glacial age on the environment.** The phrase a *glacial age* was supplied from a source sentence and the remainder of the question was supplied by a template. One advantage of using the SRL parse is that sentences with the same semantic structure will map to the same SRL parse even if they have very different syntactic structures, as seen in the following.

Source sentence: **Because of this trapped heat, the temperature of the Earth increases.**
Source sentence 2: The temperature of the Earth increases due to this trapped heat.

Generated question: Describe the factor(s) that contribute to the temperature of the Earth.

In both sentences, for the SRL parse, the predicate is increases, its A1 argument is the temperature of the Earth and the AM-CAU modifier is the because or due to phrase. Therefore, one template could match these two sentences whereas a PSG approach would need two patterns, but more significantly, it would be more difficult to generate a causation question from the syntactic patterns of the because phrase and the due to phrase which are represented as a PP and an ADJP, respectively in the PSG parse. Notice also, that the generated question above has relatively little overlap with the source sentences. In contrast, the questions generated from the Heilman and Smith system, below, are more tightly coupled to the source text.

Generated questions from the Heilman and Smith system:
- What does the temperature of the Earth increase due to?
- What increases due to this trapped heat?
- Does the temperature of the earth increase due to this trapped heat?

An evaluation of the Lindberg system by a graduate student in Education determined that 85% of the questions were grammatical, 66% made sense, and 17% had learning value.

3.2.7.2. Mazidi: 2014

In 2014 I developed my first question generation system as a class project for two different classes. The system was a template-based SRL system [75] inspired by the Lindberg system. A design decision was made to only generate questions that could be answered from the text so that both questions and answers could be generated, as in the Heilman and Smith system. Another design decision was to develop a system that would work with any expository text. All of the templates were domain independent and tested on source texts from various science and social studies texts. Although the Lindberg et al. system did have a few filters, the system still generated questions such as: What can it emit? and How often
can this occur? To prevent vague questions, the Mazidi 2014 system had many filters to greatly limit the generation of these types of extremely vague questions. In a comparison of generated questions, the system achieved a 44% reduction in the error rate compared to both the Lindberg et al. system and the Heilman and Smith system.

Source Sentence: If the atoms are pulled apart, potential energy goes up because you are separating particles that attract each other.

Generated questions:
- What happens if the atoms are pulled apart?
- Why does potential energy go up if the atoms are pulled apart?

3.2.8. Design Approach: Discourse Analysis

Discourse analysis has the potential to be very helpful in question generation, by identifying key points within a passage and the relationships between them, as well as providing context for sentences.

3.2.8.1. Agarwal, Shah and Mannem: 2011

A unique approach by Agarwal et al. [3] explored the use of discourse connectives in QG. They use the discourse connectives because, since, when, although, as a result, for example, and for instance, which are paired with appropriate question types. The general approach used is a template and SRL approach similar to Mannem et al. [74]. This system generates questions of type: why, when, give an example, and yes/no. The significance of the Agarwal et al. approach is that they explore both inter-sential and intra-sential discourse connectives; however, they made the assumption that a discourse connective at the beginning of a sentence referred to the immediately preceding sentence. This assumption turned out to not be reliable, leading to many confusing generated questions.

3.2.9. Design Approach: Dependency Parse and Templates

The third type of parse available to NLP researchers is the dependency parse, which identifies the root of a sentence and links all the words in a sentence in a graphical structure, labelling each connection according to the relation between the words. Nivre [90] observes
that dependency relations provide a relatively direct encoding of the semantic relations between predicates and their arguments.

3.2.9.1. Mazidi: 2015

Although the dependency parse had been used as an ancilliary tool for tasks such as sentence simplification, no system prior to my 2015 system [77] had fully explored the dependency parse for question generation. The system was a template-based system that built on the dependency parse, paired with information from semantic role labels and discourse cues. Whereas my first QG system [75] achieved a 44% reduction in the error rate compare to state-of-the-art question generation systems, this system achieved a further 17% reduction in the error rate. In the sample question below, we see an example of how template-based systems can generate a question that is not so tightly bound to the text of the source sentence.

Source Sentence: Cytoplasm is made up of cytosol, a watery fluid that contains dissolved particles and organelles.

Generated questions:
- Describe the composition of cytoplasm.

This system was the prototype of the question generation system presented in Chapter 4 which expanded upon this work.

3.3. One-Off Approaches

Some novel approaches to question generation have proven interesting, but for some reason were never replicated or extended by other QG researchers. These “one-off” approaches discussed below include patterns gleaned from the Web, and an approach using minimal recursion semantics. Since the approaches discussed below achieved reasonable results, it seems likely that some researchers would try to improve on these methods at some point.

3.3.1. Kunichika, Katayama, Hirashima and Takeuchi: 2004

Kunichika et al. [62] extracted syntactic and semantic information from stories using Definite Clause Grammar (DCG). The syntactic information included POS, and the syntac-
tic structure of a sentence, as well as features of constituents. Semantic information included information similar to what would be extracted in a predicate-argument parse. Several different approaches were used to generate questions. The first method asks about the content within one sentence. These are typically wh-questions. A second method of generation is to use synonyms or antonyms. For example, the system could ask *Is Jane free?* from the sentence *Jane is busy.* A third method is to use modifiers of the same noun across multiple sentences. For example, the system could generate the question *Did John sit on a small white bench?* from two sentences: *John sat on a white bench* and *There was a red bicycle near the small bench.* A fourth method could generate *Was there a red bicycle near the small bench which John sat on?* by combining the above two sentences, making one a subordinate clause. Finally, time and space questions are asked based on the semantic information extracted by the DCG. These questions are designed to probe the reading comprehension of English language learners.

3.3.2. Curto, Mendes and Coheur: 2012

Curto et al. [32] created a system to learn patterns from the web based on seed question/answer pairs. The question/answer pairs were sent to Altavista and a set of documents that contained both terms were kept. An example of a pattern is: *<NAME> was born in <ANSWER>*. Patterns such as this can be used to generate new questions from input sentences. The authors reports that their results are competitive with systems in QGSTEC2010.

3.3.3. Yao and Zhang: 2010

Minimal recursion semantics (MRS) is a computational semantics framework for representing language. It can be seen as an alternative to predicate calculus representations in that it encodes the same information but it has a flatter semantic structure than predicate calculus when represented in tree form. MRS has received a great deal of interest from the computational linguistics community but with little emphasis on its utility for QG. A proof-of-concept paper by Copestake et al. [30] was followed by a slightly expanded prototype by Yao and Zhang [114] for QG2010STEC. Yao and Zhang conclude that their work involved
heavy machinery but is theoretically interesting and that engineering and research challenges remain. While the MRS approach is appealing in its attempt to get to the semantics of a sentence, the amount of work needed to produce simple questions from simplified sentences perhaps is perhaps one reason this approach has not been replicated and expanded.

3.4. Special-Purpose Question Generation

Within educational applications there is often a need to create special questions designed to guide students further along towards desired learning goals. Eugenio and Green [33] discuss educational applications as one of the most exciting arenas of Natural Language Generation in the field. The following examples of these special-purpose question generation systems are listed by the name of the systems for which they were designed.

3.4.1. Project Listen: 1996-present

Project Listen is an on-going project from Carnegie Mellon University which listens to children read stories and provides feedback to assist skills involved in reading such as expanding their vocabulary. Project Listen has used questions to assess and assist comprehension, assess engagement, teach and model self-questioning techniques, and help with vocabulary. One use of QG in the system is to directly teach students a self-questioning strategy which has been shown to improve reading comprehension. In teaching this skill, a canned instructional sentence is followed by an automatically generated question based on the text that illustrates the technique of self-questioning, such as: *Right now the question I’m thinking about is, why was the country mouse surprised?* The system then scaffolds the child in creating their own question by having them select a character and a question type (why, what or how) from menu options. The tutor then gives the child feedback on the question. [87]

Another innovative way that Project Listen used automatically generated questions was to help classify children’s responses to a reading tutor that was teaching them to ask their own questions. The automatically generated questions were created by filling in a template such as Template 1 below with text from an SRL parse of children’s stories. An
A0 argument would fill the THING slot, and the predicate and A1 argument would fill the VERB-PHRASE slot.

Template 1:
I wonder | I’m wondering how|why|if|when <THING> <VERB-PHRASE>.

Example:
I wonder how wind makes electricity.

These automatically generated questions formed a synthetic corpus that was the basis of a language model and PFSG (probabilistic finite state grammar) used in classifying children’s responses [27].

3.4.2. AutoTutor: 1997-present

AutoTutor is a long-term project from the Institute for Intelligent Systems at the University of Memphis. AutoTutor is a natural language tutoring system that has been implemented in the domains of computer literacy, physics and critical thinking. AutoTutor has been an effective learning technology, producing learning gains about 0.8 standard deviations above controls who read text alone for an equivalent amount of time [91]. AutoTutor guides the dialogue with the following moves: pump, prompt, hint, assertion, correction, summary, and short feedback that can be positive, negative or neutral.

3.4.3. Comprehension SEEDING: 2011-2014

The Comprehension SEEDING project is a classroom technology system designed to support student learning through Self-Explanation, Enhanced Discussion, and INquiry Generation. A teacher initiates a session with a question that is shown both on a classroom display and on each student’s tablet computer. Students construct their own response to the question and the system provides the teacher with the responses, which are also clustered into four groups based on text similarity measures. Teachers were trained in QtA (Questioning the Author) [10] techniques which encourage learners to be more actively engaged with text. In 2014 I created a component in the system that suggested follow-up discussion questions to the teacher that were automatically generated based on QtA stems and the text of the
original question. Examples are shown below; CONCEPT is the noun phrase of the original teacher question and reference answer identified via NLP techniques to be the most relevant [93].

Sample Question Stems based on identified concept:
- Why do we need to understand CONCEPT?
- List concepts that are related to CONCEPT. How are they different?
- Can you describe an example of CONCEPT?

3.5. Conclusions

Becker et al. [12] advocate for a view of question generation as a three part task: (1) target concept identification, (2) question type determination, and (3) question realization. Some might observe that this is actually an algorithm for how a human would create an assessment question, and does not take into consideration the practical realities of NLG. Note that none of the systems described in this chapter take this approach, even the ones that cite it as a goal. Instead, these systems all take the approach of exploring the source material and generating as many good questions as possible. A more practical realization of this three part goal would be to separate the generation and selection tasks. That is, have a QG component generate questions, and another component select the appropriate content and question type, given the pedagogical goal at that moment, from available questions. It is not feasible with the current state of NLG to imagine that a system could identify the optimal concept with which to engage a learner, determine the best type of question to ask, retrieve appropriate material from some source from which to generate a question, and somehow generate that question, all in real time. Only the best human tutors have this ability, so the goal of implementing these skills in an ITS remains a distant one.

The above survey of QG approaches can additionally be seen as QG in varying levels of specificity of the source text: words and their meaning, words and/or phrases based on their importance in the text, combinations of key words in an ontology, sentences, and one system that, in some instances, was able to span more than one sentence, albeit not reliably.

The above review of the QG literature shows that generating questions automatically
is a vibrant, ongoing research area with many creative approaches to solving this challenge. What is even more challenging than QG, however, is generating mostly questions of quality and knowing when a generated question is not a quality question. This dissertation addresses these two issues.

Table 3.2 lists the major stand-alone QG systems described in this chapter in chronological order and includes authors, approach and types of questions generated (see Table 3.1).
<table>
<thead>
<tr>
<th>Year</th>
<th>Authors</th>
<th>Approach</th>
<th>Q Types</th>
</tr>
</thead>
<tbody>
<tr>
<td>1976</td>
<td>Wolfe [110]</td>
<td>lexical patterns</td>
<td>SA</td>
</tr>
<tr>
<td>2004</td>
<td>Kunichika, Katayama, Hirashma, Takeuchi</td>
<td>Definite Clause Grammar</td>
<td>FC</td>
</tr>
<tr>
<td>2005</td>
<td>Brown, Frishkoff, Eskenazi</td>
<td>WordNet</td>
<td>MC, FC</td>
</tr>
<tr>
<td>2007</td>
<td>Rus, Cai, Graesser</td>
<td>PSG parse, templates</td>
<td>SA</td>
</tr>
<tr>
<td>2008</td>
<td>Gates</td>
<td>PSG parse, transformation</td>
<td>LB</td>
</tr>
<tr>
<td>2009</td>
<td>Wyse, Piwek</td>
<td>PSG parse, templates</td>
<td>SA</td>
</tr>
<tr>
<td>2010</td>
<td>Liu, Calvo, Rus</td>
<td>PSG parse, transformation</td>
<td>PHP</td>
</tr>
<tr>
<td>2010</td>
<td>Kalady, Ellikkottil, Das</td>
<td>PSG parse, transformation</td>
<td>SA</td>
</tr>
<tr>
<td>2010</td>
<td>Ali, Chali, Hassan</td>
<td>PSG parse, transformation</td>
<td>SA</td>
</tr>
<tr>
<td>2010</td>
<td>Mannem, Prasad, Joshi</td>
<td>SRL parse, transformation</td>
<td>SA, CR</td>
</tr>
<tr>
<td>2010</td>
<td>Yao, Zhang</td>
<td>Minimal recursion semantics</td>
<td>SA</td>
</tr>
<tr>
<td>2011</td>
<td>Heilman, Smith</td>
<td>PSG parse, transformation</td>
<td>SA</td>
</tr>
<tr>
<td>2011</td>
<td>Agarwal, Mannem</td>
<td>Key Terms</td>
<td>GF</td>
</tr>
<tr>
<td>2011</td>
<td>Agarwal, Shah, Mannem</td>
<td>Discourse connective, template</td>
<td>SA, CR</td>
</tr>
<tr>
<td>2012</td>
<td>Becker, Basu, Vanderwende</td>
<td>Key Terms</td>
<td>GF</td>
</tr>
<tr>
<td>2012</td>
<td>Olney, Graesser, Person</td>
<td>Ontology</td>
<td>PHP</td>
</tr>
<tr>
<td>2012</td>
<td>Curto, Mendes, Coheur</td>
<td>web pattern extraction</td>
<td>factual</td>
</tr>
<tr>
<td>2013</td>
<td>Lindberg, Popowich, Nesbit, Winne</td>
<td>SRL parse, templates</td>
<td>SA, CR</td>
</tr>
<tr>
<td>2014</td>
<td>Mazidi, Nielsen</td>
<td>SRL parse, templates</td>
<td>SA</td>
</tr>
<tr>
<td>2014</td>
<td>Afzal, Mitkov</td>
<td>Ontology</td>
<td>MC</td>
</tr>
<tr>
<td>2015</td>
<td>Jouault, Seta</td>
<td>Ontology</td>
<td>SA, CR</td>
</tr>
<tr>
<td>2015</td>
<td>Chaudhri, Clark, Overholtzer, Spaulding</td>
<td>Ontology</td>
<td>SA, CR</td>
</tr>
<tr>
<td>2015</td>
<td>Chali, Hassan</td>
<td>SRL parse, transformation</td>
<td>SA</td>
</tr>
<tr>
<td>2015</td>
<td>Mazidi, Nielsen</td>
<td>Dependency parse, templates</td>
<td>SA</td>
</tr>
</tbody>
</table>

Note: Year indicates the most recent system paper on the given system.
MARGE (MARGE automatically reads generates and evaluates) is the question generation software at the heart of this dissertation. This chapter describes MARGE system components that generate questions from sentences (MARGE-S) and the next chapter describes MARGE system components that generate questions from the passage as a whole (MARGE-P).

4.1. MARGE

MARGE was developed in Python 3 and tested on a MacBook (OS X Yosemite) and a desktop running Ubuntu 12.40. MARGE consists of approximately 3000 lines of code that are internally documented and also have an accompanying set of markdown/html pages created with Mou.\(^1\) The markdown/html pages describe how to run MARGE and provide detailed explanation of system functions for developers wishing to modify and improve MARGE.\(^2\)

MARGE expects a preprocessed text input in which each sentence is placed on its own line. Section headings are also placed on their own lines and begin with \#\) to distinguish them from sentences. Python scripts that automate preprocessing are included with MARGE.

4.2. The Passage Class

MARGE reads in the preprocessed text file which is considered to be a passage. A passage is one conceptual unit, which may be one section of one textbook chapter, or one file of text extracted from a Wikipedia or other encyclopedic article. MARGE creates a Passage object which is a container for everything MARGE learns about this text. The driver program that runs MARGE first reads the input text, creates the passage object and then performs some passage analysis that is useful for MARGE-S. This passage analysis includes functions to find acronyms in the text, as well as words that are always capitalized, and phrases with capitalization. These items are stored in the passage object and are helpful in making capitalization decisions in generating questions. In addition, the TextRank algorithm is run on the passage and the top 25 words are stored in the

\(^1\)http://25.io/mou/
\(^2\)http://www.karenmazidi.com/projects.html
Passage object to be used to rank the generated questions for importance. There are more fields in the Passage object that will be discussed in the next chapter.

4.3. The Section Class and the Sentence Class

MARGE takes the input passage text and divides it into sections based on section headings. Within each section, sentence objects are created via the DeconStructure algorithm, described in the next section. The sentence object is a container for everything MARGE gleans about a sentence from various parses as well as NLP heuristics.

MARGE utilizes the SPLAT³ parser from Microsoft Research. This parser was chosen because it provides both the dependency parse and SRL parse. In earlier work [77], I combined the Stanford dependency parse with a SENNA SRL parse, but found that sometimes the two parsers tokenized the text differently which introduced errors. In addition to the parsing information, SPLAT also provides tokenization, POS tags, lemmas, and other information through a JSON (JavaScript Object Notation) request. Although the SPLAT parser was used, using another parser would only require minor changes in the code.

4.4. The DeconStructure Algorithm

The DeconStructure algorithm has one major objective: a sentence is taken apart to be restructured in such a way that reveals what it is trying to communicate. This involves two major phases: deconstruction, then structure formation. In the deconstruction phase, the sentence is parsed with both a dependency parse and an SRL parse. Additionally, word lemmas and parts of speech are gathered, along with named entity information. In the structure formation phase, the algorithm first divides the sentence into one or more independent clauses, then utilizes information from SPLAT to identify clause components, assigning each a label that represents its function within the clause. Before delving into the specifics of these two phases, we justify the approach with theoretical foundations.

4.4.1. Theoretical Foundations

The Cambridge Grammar of the English Language [53] identifies three essential concepts in the analysis of sentences: (1) Sentences have parts, which may themselves have parts, (2) The parts of sentences belong to a limited range of types, and (3) The parts have specific roles or functions

within the larger parts they belong to. Kroeger [60] identifies three aspects of sentence structure: (1) argument structure, (2) constituent structure, and (3) functional structure. With these concepts in mind, the DeconStructure algorithm was designed with three desiderata: (1) Identify sentence constituents in a manner that is intuitive yet consistent with linguistic foundations, (2) Classify constituents from a set of types indicating the semantic function of constituents within sentences, and (3) Determine the sentence pattern: a sequence consisting of the root predicate, its complements and adjuncts.

4.4.2. Parser Comparisons

In prior work, we determined that no one parse tells us everything we would like to know about a sentence, as each of the three parser types gives its own particular viewpoint. Table 4.1 compares parser outputs. The PSG (phrase structure grammar) parse identifies sentence constituents and labels phrases with the appropriate phrase label such as VP, NP, and so forth. The SRL parse (semantic role label parse, also called predicate-argument parse) identifies numbered arguments of the predicate as well as modifiers. The dependency parse provides a representation of the grammatical relations between individual words in a sentence. Table 4.2 shows the front end of the DeconStructure created by the algorithm, which is called the Meaning Analysis Representation of the sentence. The DeconStructure algorithm gleans the most important aspects from each of the parsers and combines them into a structure that is both intuitive and practical, thus making sentence elements readily available for downstream NLP applications, such as the question generation system presented in this paper. Although Table 4.2 shows the MAR, it is important to note that all of the parsing information from Table 4.1, as well as generated information such as sentence type, is available in the DeconStructure sentence object.

A central observation of this work is that sentence structure is a key aspect of natural language understanding and that advances in NLU at the sentence level can benefit question generation systems. To the degree that computer applications can “understand” natural language, it is with the aid of lexical resources such as WordNet and syntax parses. These different types of parses follow different linguistic traditions. The dependency grammar tradition is probably the oldest, having roots dating back to ancient Greek and Indian linguistic traditions. Analysis of semantic roles could be traced back to the Indian karaka theory of the 7th Century [57].

The question that arises in looking at these different grammar theory approaches is: How well
do any of these types of parses correspond to how humans parse sentences as we listen to a speaker or read text? Chomsky [28] proposed that we have an internal grammar in our minds that allows us to make sense of language. This idea has been controversial since its publication, but recent research in neuroscience has found some evidence that Chomsky was on the right track, although the research provides no insight as to whether these structures are innate or developed through experience. Using magnetoencephalography, researchers at NYU were able to identify distinct cortical activity that concurrently tracked auditory input (stripped of acoustic cues) at different hierarchical levels: words, phrases, sentences [34]. In other words, a hierarchy of neural processing underlies grammar-based internal construction of language. This exciting research may in the future be able to tease out what information these hierarchical processes actually encode. As that occurs, perhaps parsers could be developed in which sentence structure corresponds to our internally encoded structure. For example, it’s doubtful that we hear the beginning of sentence and think: that’s an NP. Rather, we think: that’s what we are talking about, i.e., the subject. And the rest of the sentence is just telling us what action surrounded that subject and possibly other entities. While we await further advances from neuroscience, it seems practicable to continue research in sentence representation forms that correspond to an intuitive understanding of sentence meaning, such as the sentence representation discussed in this work, rather than a particular linguistic tradition.

4.4.3. MAR versus other Representations

The MAR is not the first attempt by a researcher to create a representation of a sentence, and it will not be the last. An active research topic at the moment is combining word vectors, such as from word2vec or Glove, in order to represent a sentence, possibly in conjunction with deep learning. This is an exciting research area worth watching.

Another current representation scheme is AMR (Abstract Meaning Representation) which seeks to create a semantic representation for a sentence. The SPLAT parser does provide an AMR representation but it did not provide all the information needed for question generation, therefore this approach used the MAR output from the DeconStructure algorithm. Appendix E provides a comparison of AMR and MAR.
<table>
<thead>
<tr>
<th>Token</th>
<th>PSG</th>
<th>SRL</th>
<th>Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 The</td>
<td>(S(NP*)</td>
<td>B-A0</td>
<td>det(algorithm-3,the-1)</td>
</tr>
<tr>
<td>2 DeconStructure</td>
<td>*</td>
<td>I-A0</td>
<td>compmod(algorithm-3,DeconStructure-2)</td>
</tr>
<tr>
<td>3 algorithm</td>
<td>*</td>
<td>E-A0</td>
<td>nsubj(creates-4,algorithm-3)</td>
</tr>
<tr>
<td>4 creates</td>
<td>(VP*)</td>
<td>S-V</td>
<td>ROOT(root-0,creates-4)</td>
</tr>
<tr>
<td>5 a</td>
<td>(NP(NP*)</td>
<td>B-A1</td>
<td>det(creation-4,a-5)</td>
</tr>
<tr>
<td>6 functional-semantic</td>
<td>*</td>
<td>I-A1</td>
<td>amod(creation-4,functional-semantic-6)</td>
</tr>
<tr>
<td>7 representation</td>
<td>*</td>
<td>I-A1</td>
<td>dobj(creation-4,representation-7)</td>
</tr>
<tr>
<td>8 of</td>
<td>(PP*)</td>
<td>I-A1</td>
<td>adpmod(creation-4,of-8)</td>
</tr>
<tr>
<td>9 a</td>
<td>(NP*)</td>
<td>I-A1</td>
<td>det(sentence-10,a-9)</td>
</tr>
<tr>
<td>10 sentence</td>
<td>*))</td>
<td>E-A1</td>
<td>adpobj(of-8,sentence-10)</td>
</tr>
<tr>
<td>11 by</td>
<td>(PP*)</td>
<td>B-AM-MNR</td>
<td>adpmod(creates-4,by-11)</td>
</tr>
<tr>
<td>12 leveraging</td>
<td>(S(VP*)</td>
<td>I-AM-MNR</td>
<td>adpcomp(by-11,leveraging-12)</td>
</tr>
<tr>
<td>13 multiple</td>
<td>(NP*</td>
<td>I-AM-MNR</td>
<td>amod(parses-14,multiple-13)</td>
</tr>
<tr>
<td>14 parses</td>
<td>*))</td>
<td>E-AM-MNR</td>
<td>dobj(leveraging-12,parses-14)</td>
</tr>
</tbody>
</table>

**Table 4.1. Comparing Parser Outputs: Phrase Structure Grammar, Semantic Role Label, Dependency**

<table>
<thead>
<tr>
<th>Constituent</th>
<th>Text</th>
<th>Head</th>
<th>Governor</th>
</tr>
</thead>
<tbody>
<tr>
<td>predicate</td>
<td>creates</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>subject</td>
<td>the DeconStructure algorithm</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>dobj</td>
<td>a functional-semantic representation of a sentence</td>
<td>7</td>
<td>4</td>
</tr>
<tr>
<td>MNR</td>
<td>by leveraging multiple parses</td>
<td>11</td>
<td>4</td>
</tr>
</tbody>
</table>

**Table 4.2. MAR (Meaning Analysis Representation) for Sentence in Table 4.1**
4.4.4. Practical Advantages of Multiple Parsers

The DeconStructure algorithm (see Algorithm 1) exploits synergies between the SRL and dependency parses. For example, a prepositional phrase that is dependent on the verb can be an argument or an adjunct. Knowing what role the PP is playing is crucial for NLP applications but the dependency parse does not identify this information. However, the SRL will label PPs with numbered arguments if they are arguments of the verb. By checking if a PP dependent on a root verb is also a numbered argument in the SRL parse, the PP can be identified as an argument; otherwise it will be considered to be an adjunct.

Another advantage of multiple parsers is that they can check each other. In the process of analyzing the output of the SRL and dependency parses, it was noted that when a sentence did not parse well for the dependency parse, in over 90% of cases no SRL output was created, indicating a failure of the parsers to "understand" this sentence. By flagging such sentences as having a questionable parse, the system was prevented from generating many bad questions.

4.4.5. Determining Sentence Structure

Complements are words, phrases and clauses that complete the meaning of the verb, including the objects of traditional grammar [20, 53]. The universal dependency label set has six distinct labels that may be internal complements of the VP: direct object, indirect object, attr (attribute), acomp (adjectival complement), ccomp (clausal complement) and xcomp (non-finite clause-like complement) [82]. Including the PP-argument and the case in which there are no internal VP arguments, this gives eight distinct patterns for major constituents in clauses. Table 4.3 provides pattern distribution data observed from collections of expository text. Appendix A provides sample sentences for each structure, along with generated questions. Note that all modifiers and PP that are not core arguments are available in the DeconStructure for placement in generated questions. The pattern distribution in Table 4.3 is a collective distribution from one set of expository text on many subjects from multiple authors. The particular distribution of one author on one subject can vary slightly depending on the topic and the author’s style of writing. This suggests that the analysis of pattern distributions in text could be an interesting feature in authorship attribution work. A final observation about the distribution is that one would expect significantly different distributions in other kinds of text such as narrative text, dialogue, and so forth.
### Table 4.3. Typical Sentence Pattern Distribution in Expository Text

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Meaning</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-V-acomp</td>
<td>adjectival complement that describes the subject</td>
<td>8%</td>
</tr>
<tr>
<td>S-V-attr</td>
<td>nominal predicative complement defining the subject</td>
<td>14%</td>
</tr>
<tr>
<td>S-V-ccomp</td>
<td>clausal complement indicating a proposition of subject</td>
<td>7%</td>
</tr>
<tr>
<td>S-V-dobj</td>
<td>indicates the relation between two entities</td>
<td>28%</td>
</tr>
<tr>
<td>S-V-objj-dobj</td>
<td>indicates the relation between three entities</td>
<td>&lt;1%</td>
</tr>
<tr>
<td>S-V-parg</td>
<td>phrase describing the how/what/where of the action</td>
<td>17%</td>
</tr>
<tr>
<td>S-V-xcomp</td>
<td>non-finite clause-like complement</td>
<td>8%</td>
</tr>
<tr>
<td>S-V</td>
<td>indicates an action of the entity</td>
<td>14%</td>
</tr>
<tr>
<td>other</td>
<td>combinations of constituents</td>
<td>4%</td>
</tr>
</tbody>
</table>

Algorithm 1 DeconStructure Algorithm

\[ S \leftarrow \text{set of parsed sentences} \]

\[ \text{for each sentence } s \in S \text{ do} \]

\[ \text{DIVIDEINDEPCLAUSES}(s) \]

\[ \text{for each indepClause } ic \in s : \text{ do} \]

\[ \text{Step 1: Add predicate complex} \]

\[ ic[pred.label] \leftarrow \text{predicate} \]

\[ icRoot = \text{pred.index} \]

\[ \text{Step 2: Add constituents} \]

\[ \text{for each dep } \in \text{ dependencies do} \]

\[ \text{if dep.gov == icRoot then} \]

\[ ic[const.label] \leftarrow dp \]

\[ \text{Step 3: Add ArgMs to IC} \]

\[ \text{for each AM in ArgMs for icRoot do} \]

\[ ic[AM.label] \leftarrow \text{ArgM} \]

\[ \text{Step 4: Determine pp type} \]

\[ \text{for each pp in PPs do} \]

\[ \text{if pp == ArgN then} \]

\[ pp.label = \text{ppArg} \]

\[ \text{else} \]

\[ pp.label = \text{ppMod} \]

\[ \text{Step 5: Determine ic structure} \]

\[ \text{Determine ic type (passive, active, ...)} \]

\[ \text{Classify ic pattern} \]

\[ \text{Flag sentences with questionable parse} \]

4.5. Question Generation

As seen in Table 4.3, these sentence patterns fall into a surprisingly small number of categories in well-formed expository text. For each sentence, the QG system classifies its sentence pattern prior to the question generation phase. The sentence pattern is key to determining what type of question should be asked about that sentence. This analysis was based on text extracted
from open source textbooks as well as Wikipedia passages, where each text passage consisted of the
text of one chapter section, or Wikipedia text of equivalent length. In order to identify patterns
to be included in the QG system, the following criteria was used: (1) Does the sentence pattern
occur frequently across passages in different domains? (2) Is the semantic information conveyed
by the sentence pattern consistent across different instances? and (3) Does the sentence pattern
identify important content in source sentences so that generated questions will be meaningful and
not trivial?

An independent clause can be viewed as a proposition, and the predicate identifies the rela-
tionship, property or state of the entities participating in the proposition. The predicate determines
the number of participants, or arguments, that are allowed [61]. In the S-V-iobj-dobj pattern, for
example, there must be 3 entities identified in the sentence. The predicate is often the main verb but
there are other constructions in which the predicate can be found in other syntactic categories. The
acomp constituent follows a copula verb which has negligible semantic content in this construction.
The meaning is carried by the acomp, which may be an adjective or a noun. Linguists often used
the term xcomp to denote predicate complements of various syntactic categories [61]. In contrast,
the universal dependency relations divide the complements into acomp for AP, attr for NP, ccomp
for subordinate clauses, leaving xcomp for VP. It matters what syntactic category a complement
belongs to because this provides important semantic indications of what the clause is saying. Take
for instance a ccomp compared to a dobj. They differ syntactically in that the ccomp is a clause
whereas the dobj is a phrase. Semantically, the dobj identifies the second entity in the predicate
relation whereas the ccomp can be viewed as an independent proposition either indicated by or
about the subject.

4.5.1. Templates

After a sentence object is created for each independent clause of each sentence via the
DeconStructure algorithm, the sentence pattern is compared against approximately 70 templates.
If a template matches the pattern, a question can be generated. Templates are designed to ask
questions related to the major point of the sentence as identified in the pattern (see Appendices
A). Templates also contain filter conditions which are checked. Filter conditions may check for the
presence or absence of particular verbs (particularly be, do and have), whether the sentence is in
active or passive voice, and other conditions that are documented in the template file. A list of
templates is provided in Appendix B.

4.5.2. Ranking Question Importance

A question generation system can increase its utility by ranking the output questions in order to identify which questions are more likely to be acceptable. Heilman and Smith used a logistic regression question ranker which focused on linguistic quality. The ranker more than doubled the percentage of acceptable questions in the top 20% of generated questions, from 23% to 49% [48]. The logistic regression approach has also attempted by others, but with less success. One system [66] was able to identify with 86% precision that a question was not acceptable; however, their annotator considered 83% of the questions to be unacceptable questions so the utility of the classifier is unclear.

Given that our system typically outputs questions that are grammatically correct, we decided to evaluate the question importance, an often overlooked criterion [109]. To that end we employed the TextRank algorithm [84] for keyword extraction. For a given input passage, the top 25 nouns were identified by TextRank. Then each generated question was given a score based on the percentage of top TextRank words it contained, with a penalty for very short questions such as What is keyword? Our evaluation demonstrated that outputting important questions also increases their acceptability scores.

TextRank is useful for identifying key words. It links words that co-occur and recursively ranks words based on the importance of words with which they co-occur. The TextRank algorithm for key word extraction first builds a graph of words where each word is a vertex. For our application, we chose to only include nouns. The weighted score of a given word (vertex) is determined recursively by:

\[
WS(V_i) = (1 - d) + d \sum_{V_j \in In(V_i)} \frac{w_{j,i}}{\sum_{V_k \in Out(V_j)} w_{j,k}} WS(V_j)
\]

where \(d\) is a damping factor set in our application to 0.85. The sum of In vertices is the sum of words that co-occur with the given vertex and the sum of weights this word (vertex) co-occurs with is denoted by Out.
4.6. Summary

MARGE-S consists of several major steps: create the MAR for each sentence, match sentence patterns to templates, generate questions, evaluate questions. However, the run time is still fast. Given a sample input file (previously parsed) with 133 sentences, the system ran the DeconStructure algorithm on the sentences, generated and ranked 115 questions, saving the results to a file in less than 17 seconds.

The system is fast and generates quality questions. In Evaluation 1 (see Chapter 7), the system generated 72% acceptable questions compared to 42% acceptable of the most frequently cited state-of-the-art QG system. This is a 71% increase in the percentage of acceptable questions. This advance in the state of the art was achieved by using the DeconStructure algorithm to provide internal NLU analysis of what the sentence is communicating and to the application of the TextRank algorithm to identify the most important questions.

A key contribution of MARGE-S is that sentence structure in expository text scaffolds semantics. A sentence conveys meaning not only by the words that are chosen but also by the way in which the words are arranged. When we speak or write we do not spontaneously invent new sentence structures with each sentence. We project our thoughts into familiar structures so that they can be communicated, analyzed, and remembered. The application of the DeconStructure algorithm to expository text as discussed above reveals that the overwhelming majority of sentences in academic texts fit into a handful of familiar patterns.

The fact that the structure of sentences in expository text is predictable has two advantages: structure is generative, and structure scaffolds understanding. If an author understands that the structure subject, verb, direct object identifies a relationship between two entities, there are endless possible sentences that can be constructed with that structure. The structure then gives semantic clues to the reader who can readily identify the two entities and the relation that holds between them. These two advantages of structure apply even when a sentence has no real-world meaning. Chomsky’s Colorless green ideas sleep furiously [28] fits the pattern subject, verb, Mod:Manner, as does a more sensible Fat lazy dogs sleep soundly. In fact, the structure of the nonsense sentence is so familiar that we may try to make sense of it metaphorically once we realize it has no meaning in reality. Our minds attempt to join the structure with the meaning of the individual words and our knowledge of the world. In contrast, Chomsky’s other arrangement of
these 5 words: *Furiously sleep ideas green colorless* can only be viewed as a poem, not a sentence.

This work explores the sentence structures revealed by the DeconStructure algorithm applied to expository text, for the purpose of question generation. However, the Meaning Analysis Representation created by the DeconStructure algorithm should also be helpful for question answering systems and other NLP applications.

MARGE-S has pushed the boundaries of the state of the art of question generation. The next chapter describes how MARGE-P pushes the boundaries further by breaking past the sentence barrier in automatic question generation.
CHAPTER 5

MARGE-P

This chapter describes MARGE system components that generate questions from the passage as a whole. These passage-level questions typically require a student to synthesize information from multiple sentences and therefore would be level 2 questions in the taxonomy presented in Chapter 2. In contrast to the questions generated in MARGE-S, the answers are not generated for these passage-level questions because that is beyond the state of the art of current AI techniques. Also in contrast to the MARGE-S questions, passage questions are generated in the MARGE-P functions and not via templates.

Each of the sections below describe sources of information gathered for passage question generation, as well as how these sources of information are used for question generation. Figure 5.1 gives an overview of the sources of information and types of questions generated.

The Passage object was described in the previous chapter as a container for everything MARGE learns about the passage. Most of the fields of the Passage object are filled in in the MARGE-P functions which utilize several techniques for question generation.

5.1. Topic Modeling

The R topic modeling package was used\(^1\) to determine the top topics for each passage section, as well as the words for each topic. Topic modeling is run on the sections of the passage, so that each section is considered a document in the topic modeling paradigm.

Topic modeling is a generative bag-of-words model which learns topics and topic words from frequency measures in the corpus documents. Topic modeling assumes that some probabilistic generative process created the documents. There are actually two levels of probabilistic generation: (1) each topic is a distribution of words \(w = (w_1, \ldots w_N)\) from the corpus vocabulary \(V\), and (2) each document is a distribution of topics. So we must find the term distribution \(\beta\) for each topic, as well as the proportions \(\theta\) of the topics for document \(w\). A Dirichlet model gives us a mathematical way to assign prior probabilities to all possible models that could have generated the observed process. In other words, we see the result of the distribution in the documents and the Dirichlet

\(^1\)https://cran.r-project.org/web/packages/topicmodels/vignettes/topicmodels.pdf
analysis lets us discover the distributions that resulted in the observed results. Mathematically this is expressed as follows:

$$\beta \sim \text{Dirichlet}(\delta)$$

$$\theta \sim \text{Dirichlet}(\alpha)$$

For a given topic $$z_i$$, chosen from

$$z_i \sim \text{Multinomial}(\theta)$$

and a word $$w_i$$ is chosen from a multinomial distribution conditioned on the topic

$$z_i : p(w_i | z_i, \beta)$$

an iterative two-step Expectation-Maximization process is used to improve the estimates until a certain threshold is reached. In the maximization phase, the sum over the log-likelihoods of the documents is maximized with respect to parameters $$\alpha$$ and $$\beta$$. For the expectation step, for each document optimal values are found for parameters in an LDA (Latent Dirichlet Allocation) model. Also, a Gibbs sampling technique is used for estimating the unseen processes creating the documents. For our application we set the burnin parameter to 4000, the number of iterations to 2000, and the number of starts to 5, letting the system pick the best of the 5.

The results of topic modeling are very sensitive to the number of topics, $$k$$. This parameter is set in MARGE’s configuration file `config.py`, which allows the user to select $$k$$ based on their observations of the data and topic modeling results. If the number of topics, $$k$$, is set to $$X$$, then the system will determine an appropriate value of $$k$$ as follows. In an initial run of the R topic modeling script, $$k = 2 \times |\text{sections}|$$ since each section will have at most 2 topics used for question generation. Then, if there are $$n$$ unused topics in the top 2 topics for all sections, $$k$$ will be reduced by $$n$$ and topic modeling will be rerun with this new $$k$$.

The configuration file also sets the number of terms per topic to display to 6. The top 6 stemmed words of the top two topics are stored as `topic_words`. In addition, a dictionary is created of topic modeling words and their counts in the section. Appendix D gives an example of input text from the evaluation set and the resulting topic modeling words.

5.1.1. Topic Modeling Questions

Phrases are gathered from the most important topic modeling words of a section to generate summary questions such as: *Explain what you learned about topic phrase in this passage.*
Figure 5.1. Sources of Passage-Level Question Generation.
The potential topic phrases are gathered by starting with the top 6 topic terms. These 6 terms are then taken in every possible combination and if they are found in the section text, are kept for question generation.

A "hint" feature is enabled that looks through the section sentences, extracting sentence constituents that contain the phrase. An example follows:

Explain what you learned about brain waves in this passage. Your discussion may include the following phrases:
- higher amplitude brain waves than alpha waves;
- a rapid burst of higher frequency brain waves that may be important for learning and memory.

5.2. TextBlob

Noun phrases are extracted using the Python package TextBlob\(^2\) which is built on top of the better-known NLTK package. The list section_nps contains a list of nps for a section.

5.2.1. Noun Phrase Questions

For noun phrases that have the same head word, a compare question is generated. Example: *Differentiate between alpha waves and theta waves.*

For leftover noun phrases that were not matched with another noun phrase, a description question is generated which asks the student to relate the noun phrase to the section topic. Example: *Describe the relation between brain waves and stages of sleep.*

5.3. Head Words of Subjects and Direct Objects

The Meaning Analysis Representation of each sentence in the section is searched to find head words of subject or dobj constituents but not in the always_capitalized list (described in the previous chapter). The field unnamed_entities is a list of these nouns. The field candidate_phrases is a list of noun phrases that include unnamed entities. The field entity is a dictionary of unnamed entities where the key is the entity and the value is its part of speech.

The unnamed entities are categorized into one of 5 types: people, living, physical, event, and abstract by working through the WordNet\(^85\) hierarchy until the appropriate top-level category is found.

The list possible_terms is all unique entities that are not common words appearing in a
dictionary of 100K common English words.\(^3\)

The above information is used to generate terminology questions, described in the next
section.

5.3.1. Terminology Questions

If a possible term is a person according to the WordNet analysis described above, a describe
question is generated. Example: *Describe the role of neuroscientists as described in this section.*

From all possible_terms that are not people, pairs of similar terms are gathered where
similarity is a Levenshtein distance. The Levenshtein distance measures how similar two items are
by the number of insertions, deletions and substitutions required to change one item into another,
normalized by item length. For example, the terms mesoderm and ectoderm meet the threshold
of similarity (0.4) and so one of three compare questions will be generated, the pairs are rotated
through the three wordings to provide variety. Examples:

- Compare and contrast term1 and term2.
- Describe the difference(s) between term1 and term2.
- Is there a relationship between term1 and term2? Explain.

Any item in possible_terms that did not find a close term is used to form a question:
*Provide a definition for term and discuss its relation to section_topic.* For example: *Provide a
definition for epithelium, and discuss its relation to epithelial tissue.*

5.4. Section Topic

The section_topic is the noun phrase in the section heading that occurs most frequently
in the section text. The first section heading is used for subsequent sections if the system cannot
find a noun phrase in the heading.

5.4.1. Summary Question

Finally, a summary question is asked about the section topic, for example: *Summarize what
you learned about epithelial tissues in this section.*

\(^3\)https://gist.github.com/h3xx/1976236/forks
5.5. Discussion

In the discussion for MARGE-S we noted the speed of a sample input file which showed that MARGE-S components were fast. Timing MARGE-P components indicated that these components added a negligible amount of time to the total run time, an average of about 0.02 seconds. Running 3 files with an average length of 100 sentences through the entire MARGE system including topic modeling, averaged 0.199 seconds per input sentence, so that an input file of 100 sentences would run in less than 20 seconds. This figure is assuming that the input files had previously been parsed. Parsing time seems to vary with the response time of Microsoft SPLAT JSON requests, but averages around one second a sentence. The config.py file has a field which indicates whether or not SPLAT needs to be run or if the previously parsed and pickled files can be used.

The question generation percent of MARGE-S varies with the passage and the number of templates used. With the current 70+ templates in the system, the generation rate is about 70%, or 7 questions per 10 input sentences. As anticipated, MARGE-P produces fewer questions. The generation rate varies by passage but was observed to be as low as 40% of the number of MARGE-S questions and as high as 100%. Future work could involve a ranking component to select among MARGE-S and MARGE-P questions using information already gathered such as topic modeling words and TextRank words.
CHAPTER 6

EVALUATING AUTOMATICALLY GENERATED QUESTIONS

This chapter explores methods of evaluating the quality of automatically generated questions, discusses the relative merits of these methods, and outlines proposed methods and metrics for evaluating questions generated by MARGE. Most, but not all, of the QG systems discussed in Chapter 3 included an evaluation component. These are discussed in this chapter, as well as other evaluation frameworks encountered in the literature.

There is no standard way to evaluate automatically generated questions. Evaluations methods found in the literature include:

- IR metrics: precision, recall, F-measure
- Item test theory
- Acceptability determined by selected judges
- Acceptability determined by crowd sourcing
- Pedagogical utility
- Turing-type evaluations
- Ranking programmatically

6.1. What are we Measuring?

The primary question when considering how to evaluate questions is: What are we measuring? Various evaluations methods, discussed below, may seek to measure:

(1) Syntactic quality of questions: grammaticality, naturalness, etc.
(2) Semantic quality of questions: clarity, meaningfulness, etc.
(3) Variety of question types
(4) Utility of questions for learners

As noted by Rus et al. [103], two core aspects of a question (aside from its linguistic quality) are the goal of the question and its importance. This suggests that evaluation of questions outside of an authentic educational context may be problematic. A question ultimately may be "good" only if it helps a given student at a given point in their learning.
6.2. IR Metrics: Precision, Recall and F Measure

One method of evaluating questions is to compare them to a set of human-authored questions that were manually constructed over the same source material. Then, measures of precision, recall and F-measure could be used to compare the two sets of questions. This method was used by Kalady et al. [58]. They asked 5 independent judges to generate all possible questions from 20 sentences from the Brown corpus. Then they compared the output of their system to these human authored questions. The recall measure was the number of questions generated by the system that were also in the human-authored set, divided by the number of questions in the human-authored set. Precision was defined to be the number of questions generated that were also in the human-authored set, divided by the total number of questions generated. Note that this measure would penalize a system for generating good questions that were not contained in the human-authored set.

Jouault et al. [56] used the same concept as precision but called it coverage. In their case they compared their generated questions to a set of SparkNotes questions over the same topic. Chaudhri et al.[25] also used coverage by comparing questions suggested by their system to those generated by teachers of the subject. Another system using coverage for evaluation was Kunichika et al. [62] which compared their generated questions with those from a reading comprehension problem set.

This method has not been widely used because it would penalize a system for generating a question that the human authors did not think of, and because it would be costly to have humans try to generate all possible questions for a large corpus.

6.3. Pedagogical Metrics: Item Test Theory

Linn and Gronlund [68] outline a simplified item-analysis procedure for evaluating assessment questions. Three statistics commonly associated with item analysis are discussed next: item difficulty, item discriminating power, and effectiveness of distractors.

Item difficulty is measured by the percentage of students, \( P \), who correctly answered a question, and can be calculated as shown in Equation 1. The number of students who answered the item correctly, \( R \) is divided by the total number of students, \( T \), and multiplied by 100 to obtain a percentage.
To calculate item discriminating power, and effectiveness of distractors, a set of upper scores, $U$, and a set of lower scores, $L$ must be created. Tests are scored and ordered from highest to lowest. The upper group and lower group of tests are selected from the top/lower 25 to 30%. The middle tests are not considered.

Discriminating power compares the number of students in the upper and lower groups who answered a question correctly. A question has good discriminating power if students in the upper group tend to answer it correctly and those in the lower group do not. Equation 2 indicates how discriminating power, $D$, is calculated. The difference between the number of correct answers from the upper and lower groups is divided by $\frac{1}{2}$ of the total number of students in both groups. A positive value for $D$ indicates that it does have discriminating power. A value of 1.0 would mean perfect discriminating power, 0 would mean no discriminating power, and a negative number would mean that more students in the lower group answered correctly than the upper group.

\[
D = \frac{(RU - RL)}{0.5T}
\]

For multiple choice questions, it is possible to calculate effectiveness of distracters. Equation 2 can be used to calculate the effectiveness of each discriminator, but the results should be interpreted in the opposite way. That is, we don’t want distractors that attract more students from the top group than from the bottom group.

Mitkov et al. [86] used item test theory in their analysis of multiple choice questions generated by their system. Specifically, they performed item analysis for the purpose of comparing manually constructed questions to those created within their system, which generated questions and then provided an environment for post-editing of questions.

6.4. Acceptability Determined by Selected Judges

Determining what percentage of generated questions are acceptable according to human judges is an approach that was used prior to the emerging popularity of crowd-sourcing. Typically
judges are asked to rate questions along various criteria either on a Likert-type ordinal scale or on a binary pass/fail basis. Gates [40, 39] had one judge rate each question as: perfect, ok, bad, or failure. The question was considered to be acceptable if the judge rated it as perfect or ok. Heilman and Smith expanded this approach by using 15 undergraduate raters, who were to indicate any of 8 possible deficiencies: ungrammatical, does not make sense, vague, obvious answer, missing answer, wrong WH word, formatting, other. A question was considered acceptable if it had none of these deficiencies. For the test set, three raters were used for each question, and a majority vote determined if a question was acceptable. The inter-rater agreement was Fleiss’s $\kappa = 0.42$, which suggests that getting agreement with this approach is challenging. Chali and Hassan [22] used a similar approach to that of Heilman and Smith with average agreement Fleiss’s $\kappa = 0.45$.

The QGSTEC2010 challenge asked judges to evaluate questions generated from paragraphs for specificity, syntax, semantics, question type correctness and diversity. For questions generated from sentences, judges evaluated relevance, question type, syntactic correctness and fluency, ambiguity and variety. One suggestion that came out of QGSTEC2010 follow-up discussions [103] is to replace absolute ratings with preference judgments. In such a scenario, judges would be shown several questions at one time, submitted by different participants, and asked to rank them based on the above criteria. To our knowledge, no evaluation has reported this approach.

Lindberg et al. [67] wanted to evaluate the educational value of generated questions as well as their linguistic quality. For this reason, they had an Graduate Education student give binary judgments for grammaticality, semantic validity, vagueness, answerability and learning value.

Olney et al. [92] had judges evaluate on a 1 - 4 scale: relevance, fluency, ambiguity, and pedagogy, plus a binary value for correct question type.

In a prior evaluation [75] of linguistic quality of generated questions, we asked university students in a teacher preparation program to evaluate questions on a 5-point scale for grammaticality, clarity, and naturalness. The average inter-annotator agreement after allowing a difference of one between the annotators’ ratings was Pearson’s $r = 0.47$. The low agreement between annotators highlights one of the advantages of crowd sources. Of the university students participating in the evaluation, one gave high ratings to almost every question, whereas another gave low rating to any question that was lengthy. Thus, individual personalities and preferences influenced the evaluation. In contrast, crowd sourcing approaches damp down individual variation because many workers are
used. This, plus the lower cost of crowd sourcing, are the principal reasons that I switched to crowd-sourcing evaluations.

Curto et al. [32] use an approach adapted from Chen, Aist and Mostow [26]. They asked judges to categorize questions into one of three groups: (1) plausible and non-anaphoric, meaning that the question was well formulated at lexical, syntactic and semantic levels, and makes sense in the context of the sentence that generated it, (2) implausible, meaning it failed any of the criteria listed in the plausible group, and (3) plausible given a context, meaning that the sentence would meet the criteria for group 1 but contains pronouns such that the reader would need the source sentence to understand to what the pronoun refers. Using this approach they achieved a very impressive $\kappa = 0.82$.

6.5. Acceptability Determined by Crowd Sourcing

An alternative to using selected judges is to use crowd sourcing sites like Amazon’s Mechanical Turk service. Snow et al. demonstrated that MTurk provides NLP annotations that approach the quality of experts for lower cost and considerably less time. Their recommendation over all NLP annotations that they tested was an average of 4 workers per HIT (Human Intelligence Task) [105]. Heilman and Smith [51] used MTurk ratings to train a question ranker, and also to evaluate the final top-ranked output of the system. Workers were shown an excerpt of sentences including the source sentence, and a generated question. Workers scored each question on a 5-point scale (bad, unacceptable, borderline, acceptable, good). Each question was rated by 5 workers, with the final rating of each question being the average of the 5 scores. They assured worker quality by using workers with a 95% acceptance rate or higher for their previous work, submitting the work in small batches, and manually and programatically monitoring the results. As a check, the work was compared against the ratings of a computational linguist and the first author. Pearson’s correlation coefficients were $r = 0.74$ and $r = 0.79$, respectively, showing good agreement. Interestingly the agreement between the computational linguist and the first author was only $r = 0.65$. This is consistent with the findings of Snow et al. [105] who speculated that aggregate scores from MTurk workers can overcome individual bias issues compared to even expert annotation work.

We also used MTurk workers in prior evaluations [76] and found the work to be satisfactory. In an evaluation of pedagogical quality, workers were given the source sentence and a generated question, and were asked to rate them on a 1 - 3 scale for linguistic quality, namely is the question
grammatical and clear. In separate HITs, workers were asked to consider whether or not the question would help them remember and understand the meaning of the source text. Two workers evaluated each question. Questions for both the linguistic and the pedagogical evaluations were considered acceptable if either they received a 3 by both workers or a 3 by one and at least 2 by the other. Pearson’s correlation coefficient averaged $r = 0.52$ using the MTurk workers which was higher than we had achieved previously with undergraduate student workers. It may be the case that a larger number of MTurk workers than 2 leads to better results. In later work, we used 4 workers per HIT.

6.6. Pedagogical Utility

This section explores ways that researchers have tried to measure the pedagogical utility of questions. Human judges can say what they will about a question, but if the question was generated for educational purposes, it seems logical that the ultimate evaluation of a question would be whether or not it achieved its purpose pedagogically. This is a difficult and costly thing to measure, and frankly, out of the range of expertise of computer science researchers. These kinds of evaluations are more likely to be performed in the context of large educational applications with a multidisciplinary team of researchers, not to mention a significant funding source.

6.6.1. Pedagogical Utility compared to Human-authored Questions

In 1977 Wolfe [111] was able to conduct a detailed evaluation of the pedagogical utility of questions automatically generated with his AUTOQUEST system because he had a boundless supply of naval recruits to work with. Naval recruits were assigned to one of four experimental groups and one control group. All groups read material previously developed by educational psychologists Richard C. Anderson and David Myrow, which was an 11-page passage describing cultural practices of a fictitious African tribe. The control group received no questions in their reading material. All other groups had to answer questions after each page of reading material. There were a total of 20 questions for each group. Two groups received cloze questions, one received AUTOQUEST questions, and the fourth group received questions authored by Anderson and Myrow. Two methods were used to create cloze questions, and each method was given to a different group. One method omitted every 5th word, and the other omitted every-other long word. In order for the time for the control group to match that of the experimental groups, the control group was instructed to read the material two more times.
After reading the material and answering the interspersed questions (except for the control group), the recruits were given a nonverbal aptitude test (Hidden Figures) in order to provide a delay between learning and retention. Then they were given an 80-item multiple choice criterion test. Of the 80 questions, 20 were over material that was not included in any of the questions, and the remaining 60 were similar to questions in the control groups (human-authored, AUTOQUEST, and long-cloze), but reworded to be in a multiple-choice format. The order of the questions was randomized.

The study concluded that the groups receiving human-authored and AUTOQUEST questions both scored half a standard deviation better on questions relating to content on which they had answered questions, but their retention on material for which they did not answer questions was no better than the control group. Further findings were that cloze questions likely interfered with learning. The author speculated that the reason for the relatively poor performance of the cloze groups was that the questions caused the recruits to focus on low-level details to the detriment of higher processing.

6.6.2. Pedagogical Utility Given a Tutoring Scenario

Becker et al. [13] describe the on-going development of a methodology for evaluating human-authored questions, given a state in a tutorial dialogue. They chose to use human-authored questions rather than automatically generated ones so as not to confound issues of grammaticality, naturalness, and so forth with question selection. A linguist was trained to author questions of the target type that an automatic system would be generating, namely QtA (Questioning the Author) [10] type questions. Essentially, their goal was to train a ranker to select the question that would provide the best dialogue move at a given point. Data for this experiment was from log files created during a WoZ (Wizard-of-Oz) study in which a human tutor is participating in the dialogue of the MyST (My Science Tutor) system, an ITS for elementary school science. Amazon’s Mechanical Turk service was used to evaluate the questions. Workers were given text of the preceding section of the tutorial dialogue, a list of the learning goals, and 6 candidate questions which they were to rate on a 1 - 10 scale for the appropriateness of each question at that point. This data was used to build an SVM regression model. In evaluating the SVM ranker, they treated it as one of the raters and compared its rankings to those of the MT workers. Results were encouraging that the system could rank as well as the novice MT workers. However, issues that need future work include dealing with the lack
of experience of the MT workers with the QtA approach, and including more dialogue act features. This was a very preliminary experiment on a very complicated task. One further question that is unaddressed in this work, is that even if annotators agree that a given question is the optimal one at a given point in the dialogue, does this mean anything in terms of actual learning gains?

6.6.3. Pedagogical Evaluation of ITS

The most important evaluation for educational applications such as Intelligent Tutoring Systems is learning gains, which seeks to compare scores obtained by students who used the system versus those who did not. This is ascertained by comparing scores on a pretest and a posttest. Learning gains are measured by the difference between the pre- and posttests, sometimes normalized by the maximum possible gain. Persistence of learning is measured by administering a posttest after a delay of days or weeks. Effect size seeks to measure how much more one condition is with respect to the other. One means of calculating effect size is Cohen’s $d$, which is the difference between the means of the pretest and posttest, divided by the standard deviation of either. [33]

Eugenio and Green [33] discuss Intelligent Tutoring Systems as one of the most exciting applications of Natural Language Generation in the field. It is not yet known which features of a tutorial dialogue engender learning. If this were known, more resources could be devoted to automatically generating those dialogue acts. The AutoTutor system [91] described in Chapter 2 has achieved learning gains about 0.8 standard deviations above controls who read text alone for an equivalent amount of time. AutoTutor guides the dialogue with the following moves: pump, prompt, hint, assertion, correction, summary, and short feedback that can be positive, negative or neutral. Which of these types of questioning techniques are most effective at various points in a tutorial dialogue remains an open question for evaluation techniques.

6.7. Turing-type Evaluations

The ultimate goal of any computer system within the AI field is to pass the Turing Test. The original Turing Test was derived from an imitation game in which an interviewer, who cannot see or hear two participants, asks a series of questions to determine which of the participants is a man and which is a woman. Turing proposed that if we replace one of the participants with a computer, and the interviewer could not determine which was human, then we could say that that computer could “think”, it would pass what would become known as the Turing Test. What if the
computer performs a task, comparable to human thinking, but not in the same way? Can we still say that the computer can think? Turing was unconcerned by this distinction of methods. Turing himself believed that by the year 2000 computers would have about a 70% chance of passing the test after 5 minutes of play. [108]

In 2002, Person and Graesser conducted a modified Turing Test they called a Bystander Turing Test, in which participants rated whether dialogue moves in the AutoTutor system were generated by AutoTutor or a human tutor. The reason for the modification, in which participants are given printed transcripts, is that it was observed that human tutors don’t respond as quickly as AutoTutor, and humans make typos. These factors could provide clues to the participants about the human tutor responses. The BTT was constructed as follows. From AutoTutor conversations, 282 were randomly selected. Within each conversation, an AutoTutor dialogue move was deleted (along with subsequent material dependent upon it) and replaced by human tutor responses, which were collected by asking skilled tutors what they would say at that point given the prior conversation. Packets containing 36 conversations were assembled, 18 with AutoTutor dialogue only and 18 with the embedded human dialogue move. Study participants (college undergraduate students) were asked to rate the last dialogue move on a 1 - 6 scale, which 1 being definitely human and 6 being definitely computer. They also evaluated the dialogue move on 1 - 6 scales for appropriateness and effectiveness. Statistical analyses indicated that the participants could not distinguish between AutoTutor and human tutor dialogue moves. Participants also found the AutoTutor moves to be as appropriate as human ones. The AutoTutor mean was 4.07 and the human tutor mean was 4.13. Similarly, effectiveness scores were similar between AutoTutor and human dialogue moves. When the effectiveness scores were grouped by dialogue move (pump, prompt, hint, assertion, correction, summary, feedback) it appeared that there was a preference for the dialogue moves that did not require as much of a cognitive load for students but resulted in more output from the tutors. The authors state that this is a reflection of students’ preference for passive instructional methods because they seem easier. [94]

Liu, Calvo and Rus [70, 71] used a similar approach in evaluating their QG system to support student academic writing. One major difference is that the participants were the student authors of the papers for which the system generated questions. Therefore, the participants were in a good position to evaluate the accuracy and the utility of the generated content. On the negative side
however, there were only 6 participants, which is a small sample size. For each student paper, a total of 20 questions were gathered for evaluation, 5 each from: the system, a lecturer familiar with the subject matter, a human tutor, and generic questions. Students were asked to rate according to 5 quality measures: (1) the question is correctly written, (2) the question is clear, (3) the question is appropriate to the content, (4) this question makes me reflect about what I have written, and (5) this is a useful question. Each was rated on a Likert-type 1 - 5 scale where 1 was ‘strongly disagree’ and 5 was ‘strongly agree.’ Results showed that the system produced questions that were as helpful as human-authored questions. The system questions outscored Generic Questions and surprisingly outscored the Lecturer-authored questions. The authors speculate that this may be due to the complexity for the Lecturer in reading 6 papers and generating 30 questions in all. This points to an interesting advantage for automatically generated questions: computers don’t get tired, overwhelmed and distracted.

6.8. Ranking Generated Questions Programatically

Heilman and Smith [49] were the first to create a statistical ranker as part of a QG system. A logistic regression model of question acceptability was developed using an extensive set of over 200 features based on the question, answer, source sentence, internal manipulations that were performed, and n-gram language model features. Of all test set questions, 27% were acceptable. By using the statistical ranker and selecting the top 20%, the percentage of acceptable questions nearly doubled, to 52%. A later experiment [51] using linear regression on data that had been evaluated on an ordinal scale achieved similar results.

Lindberg et al. [67] also created a logistic regression classifier, but which sought to classify on learning value. Features included length, language model, SRL, named entity, glossary and syntax features. Their best measure was 0.86 precision for questions with no learning value. The authors state that if they generate a bad question, they have a high probability of knowing that it is a poor question and can thus discard it.

Niraula and Rus [89] used active learning to train a classifier to judge question quality. Active learning was chosen because it requires less annotated data and therefore is less expensive. They used the data set from the Becker et al. [13] evaluation described above, but binarized the classifications to good or bad from the original good, okay, bad judgments by labeling a question good when at most one rating was okay or bad, and labeling all others as bad. Classifiers achieved
up to 74% accuracy, a result that shows promise for this approach.

6.9. Discussion of the Merits of Evaluation Schemes

In comparing the above approaches we make the following observations and recommendations for evaluating the quality of automatically generated questions such as for the MARGE system described in this dissertation.

- The limitations of IR measures was discussed above, chief of which is that a system would be penalized for generating a good question that human authors simply didn’t think of.

- A large learning gains evaluation for a stand-alone question generator such as MARGE is both beyond the scope of this proposed research, and beyond the range of Computer Science topics of interest.

- Although the questions generated by MARGE will be for educational purposes, it is the goal of this research to experiment with different techniques for generating quality questions over important content, not discovering which types of questions lead to better learning. While it is hoped that these two goals would converge, this work focuses on technical details of generating.

- Forty years ago, Wolfe established that automatically generated questions were as effective as human-authored ones in assisting with retention of new material; therefore, replicating this type of experiment may have little research value.

- In terms of question quality, it appears that crowd sourcing provides similar quality to selected judges, at a much lower cost and time required.

- Ranking questions programmatically has been shown to increase question quality and should be a part of a fully developed question generation system.
CHAPTER 7

MARGE EVALUATIONS

This chapter describes evaluations performed on the generated questions from MARGE-S and MARGE-P. The evaluations seek to answer our research question: Can infusing NLU techniques into QG lead to higher quality questions compared to approaches that employ syntactic manipulation alone, and can they approach the quality of human-authored questions?

In the following sections, two evaluations of MARGE-S questions are presented. In Evaluation 1, all MARGE-S questions are evaluated. In Evaluation 2, the TextRank algorithm is used to rank MARGE-S questions, and the top questions are compared to the current state-of-the-art system, as well as to human authored questions.

MARGE-P questions are first evaluated programmatically in terms of the number of sentences to which a student should refer in order to answer the question, and the results are discussed in terms of our two-level question taxonomy presented in Chapter 2. We also conducted an evaluation of MARGE-P questions using crowd sourcing.

7.1. Evaluation 1: All MARGE-S Sentence-Generated Questions

As discussed in Chapter 6, there is no standard way to evaluate automatically generated questions. Recent work in QG and other NLP applications favors evaluation by crowdsourcing which has proven to be both cost and time efficient and to achieve results comparable to human evaluators [105, 51]. In Evaluation 1, all questions generated from MARGE-S were evaluated using Amazon’s Mechanical Turk Service. A sample HIT (Human Intelligence Task) is shown in Figure 7.1. Workers were shown a generated question as well as the source sentence from which it was generated, and asked to rate the question. Workers were selected with at least 90% approval rating on their prior work and who were located in the US and proficient in US English. To monitor quality, work was submitted in small batches, manually inspected, and run through software to detect workers whose ratings did not correspond well with fellow workers. Each question was rated on a 1-5 scale by 4 workers. The four scores were averaged and a mean over 3.5 was considered acceptable. In addition, I gave a binary acceptable rating to each question. Questions were counted as acceptable if they received an acceptable rating from me and an average acceptable rating from the MTurk workers. Agreement between this rating and the MTurk workers’ acceptability score was $\kappa = 0.67$. 

75
7.1.1. Test Data and Results

Input evaluation data was gathered from open source textbooks as well as Wikipedia on a diversity of topics in science and the humanities. Each text passage consisted of the text of one chapter section, or Wikipedia text of equivalent length. Table 7.1 provides information about the test data and the percentage of acceptable questions generated from each text passage.

<table>
<thead>
<tr>
<th>No.</th>
<th>Topic</th>
<th>No. Sentences</th>
<th>% Generated</th>
<th>Grade Level</th>
<th>% Accept.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1.</td>
<td>Lymphatic System</td>
<td>136</td>
<td>65%</td>
<td>14</td>
<td>62%</td>
</tr>
<tr>
<td>A2.</td>
<td>Eukaryotic Cells</td>
<td>165</td>
<td>43%</td>
<td>14</td>
<td>52%</td>
</tr>
<tr>
<td>A3.</td>
<td>Federalism</td>
<td>117</td>
<td>70%</td>
<td>14</td>
<td>46%</td>
</tr>
<tr>
<td>A4.</td>
<td>International Trade</td>
<td>83</td>
<td>65%</td>
<td>12</td>
<td>56%</td>
</tr>
<tr>
<td>B1.</td>
<td>Chemical Bonds</td>
<td>138</td>
<td>46%</td>
<td>14</td>
<td>54%</td>
</tr>
<tr>
<td>B2.</td>
<td>Planned Economies</td>
<td>67</td>
<td>50%</td>
<td>15</td>
<td>70%</td>
</tr>
<tr>
<td>B3.</td>
<td>Toledo War</td>
<td>83</td>
<td>63%</td>
<td>13</td>
<td>50%</td>
</tr>
<tr>
<td>B4.</td>
<td>Tornadoes</td>
<td>151</td>
<td>38%</td>
<td>13</td>
<td>48%</td>
</tr>
<tr>
<td></td>
<td>Average</td>
<td>118</td>
<td>53%</td>
<td>14</td>
<td>55%</td>
</tr>
</tbody>
</table>

The QG system output 55% acceptable questions without any ranking component. The most frequently cited state-of-the-art QG system, Heilman and Smith [49], achieved 52% acceptable questions after a comparable evaluation, when considering only the top 20% of their automatically ranked questions. Analysis of questions that received unacceptable ratings reveal that the majority were unacceptable due to vagueness. Evaluation 2 shows the efficacy of using the TextRank algorithm to down-rank vague questions.

7.2. Evaluation 2: Highest Ranked MARGE-S Questions

In this evaluation MARGE-S questions were compared to the most-frequently cited prior question generation system by Heilman and Smith [48]. The evaluation framework was identical to that in Evaluation 1, except the score for each question was not a binary acceptability rating but an average 1-5 score from the 4 Amazon Turk workers. Agreement between each set of workers and the average had a Pearson’s correlation $r = .71$, showing high agreement.

7.2.1. Test Data

Test data consists of 10 science and humanities passages, one each from 10 open source textbooks from OpenStax and Saylor. All text sources are written at an early college reading level with an average of 83 sentences per passage. Each passage represents the text of one textbook chapter section, chosen at random. Table 7.2 lists the topics in the test data set, along with the number
of sentences in each file and the number of questions generated by the Heilman & Smith system and our system. The H&S system takes an overgenerate-and-rank approach, generating almost 5 questions per input sentence. In contrast, our system generates an average closer 0.7 question per input sentences by focusing on the important content in each sentence but not generating questions when conditions are not favorable for generating a good question.

### Table 7.2. Test Data and Questions Generated

<table>
<thead>
<tr>
<th>Topic</th>
<th>Sents</th>
<th>H&amp;S</th>
<th>M&amp;T</th>
</tr>
</thead>
<tbody>
<tr>
<td>Epithelial Tissue</td>
<td>148</td>
<td>600</td>
<td>77</td>
</tr>
<tr>
<td>Protists</td>
<td>118</td>
<td>545</td>
<td>76</td>
</tr>
<tr>
<td>Bankruptcy</td>
<td>37</td>
<td>159</td>
<td>23</td>
</tr>
<tr>
<td>Network Layers</td>
<td>79</td>
<td>267</td>
<td>55</td>
</tr>
<tr>
<td>Monetary Policy</td>
<td>90</td>
<td>431</td>
<td>37</td>
</tr>
<tr>
<td>Uzbekistan</td>
<td>71</td>
<td>351</td>
<td>52</td>
</tr>
<tr>
<td>Legislature</td>
<td>73</td>
<td>375</td>
<td>55</td>
</tr>
<tr>
<td>Jackson Era</td>
<td>46</td>
<td>279</td>
<td>24</td>
</tr>
<tr>
<td>Stages of Sleep</td>
<td>72</td>
<td>339</td>
<td>44</td>
</tr>
<tr>
<td>Education</td>
<td>103</td>
<td>715</td>
<td>50</td>
</tr>
<tr>
<td>Average</td>
<td>83</td>
<td>406</td>
<td>50</td>
</tr>
<tr>
<td>Generation Percents</td>
<td>488%</td>
<td>60%</td>
<td></td>
</tr>
</tbody>
</table>

#### 7.2.2. Results

The evaluation looked at the top 20 questions output from each system for each input file, with each system performing its own internal ranking. Table 7.3 compares the average MTurk worker
Table 7.3. Average Scores for Top 20 Questions

<table>
<thead>
<tr>
<th>Topic</th>
<th>H&amp;S</th>
<th>M&amp;T</th>
</tr>
</thead>
<tbody>
<tr>
<td>Epithelial Tissue</td>
<td>2.6</td>
<td>3.9</td>
</tr>
<tr>
<td>Protists</td>
<td>2.6</td>
<td>4.1</td>
</tr>
<tr>
<td>Bankruptcy</td>
<td>2.7</td>
<td>3.5</td>
</tr>
<tr>
<td>Network Layers</td>
<td>3.0</td>
<td>3.9</td>
</tr>
<tr>
<td>Monetary Policy</td>
<td>2.8</td>
<td>3.8</td>
</tr>
<tr>
<td>Uzbekistan</td>
<td>3.3</td>
<td>3.6</td>
</tr>
<tr>
<td>Legislature</td>
<td>3.0</td>
<td>3.1</td>
</tr>
<tr>
<td>Jackson Era</td>
<td>3.4</td>
<td>3.7</td>
</tr>
<tr>
<td>Stages of Sleep</td>
<td>3.0</td>
<td>4.0</td>
</tr>
<tr>
<td>Education</td>
<td>2.6</td>
<td>3.1</td>
</tr>
<tr>
<td>Average</td>
<td>2.9</td>
<td>3.7</td>
</tr>
</tbody>
</table>

Figure 7.2. Score Distributions. Light:H&S, Dark:M&T

ratings for each file for the two systems. Our system had a higher rating for every topic. When averaging all 200 questions, the Heilman & Smith system had an average rating of 2.9. Our system had an average rating of 3.7. The results are statistically significant, $p < 0.001$, as determined by the Student’s t-Test. Figure 2 shows a side-by-side histogram of the score distributions between the two systems. The histogram demonstrates that the majority of the Heilman and Smith system questions are below the mid-point of 3.0 and that the majority of our questions are above this mid-point. Using $> 3.0$ as the acceptability threshold, 72% of our questions are acceptable whereas only 42% of the Heilman and Smith questions pass this threshold. This is an increase in the acceptability percentage of the top questions of 71%. Interestingly, the Heilman and Smith percentage of 42% found in our evaluation of their top 20 questions is close to the 49% acceptable percentage they
found in their analysis of the top 20 percent of their generated questions.

Figure 7.3 shows the average scores of the 2 systems, along with scores of human-authored questions. The average score over all questions for the Heilman and Smith system was 2.9 whereas the average score for the Mazidi and Tarau system was 3.7. Our average is still below the human average of 4.4 but shows a significant gain in the state of the art, approaching human performance.

7.3. Evaluation 3: MARGE-P Question Breadth and Depth

MARGE-P questions are evaluated programmatically for question breadth as follows. Each MARGE-P question either asks about one key phrase, or the comparison of two key phrases. In order to answer the question from the section text, a student will need to re-read the relevant sentences and construct an answer synthesizing information from these sentences. This can be considered question breadth. Programmatically, a count of relevant sentences was calculated for each MARGE-P question by searching for the key phrase(s) in the sentences. Table C shows the average counts of the number of relevant sentences, as well as the percentage of sentences in the section that this count represents, for three evaluation texts.

A student would need to re-read an average of nearly 5 sentences in the section to adequately construct an answer to MARGE-P questions. This is about 1/3 of the sentences.
Table 7.4. MARGE-P Question Breadth

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Epithelial Tissue</td>
<td>146</td>
<td>87</td>
<td>4</td>
<td>28%</td>
</tr>
<tr>
<td>Monetary Policy</td>
<td>87</td>
<td>41</td>
<td>6</td>
<td>47%</td>
</tr>
<tr>
<td>Stages of Sleep</td>
<td>71</td>
<td>21</td>
<td>4</td>
<td>25%</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td></td>
<td>4.7</td>
<td>33.4</td>
</tr>
</tbody>
</table>

7.3.1. Question Depth

The two-level question taxonomy presented in Chapter 2 divided questions into factual comprehension questions and conceptual comprehension questions. MARGE-S questions are considered to be in the factual comprehension category because the answer is typically one phrase from one sentence. In contrast, MARGE-P questions are considered to be conceptual comprehension questions because they require the student to synthesize material from multiple sentences. MARGE is the first automatic question generation system to successfully generate questions spanning multiple input sentences and to quantify this question breadth.

7.4. Evaluation 4: MARGE-P Questions

This evaluation is conducted in a similar manner to Evaluation 2, with two differences. The first difference is that each worker is presented a one-paragraph summary of the section (from the end of the textbook chapter), rather than a single sentence. The second difference is the scale, which is a 1-4 scale where 4 is best. This scale is designed to rate questions on their semantic quality, or meaningfulness. A sample HIT for this task can be seen in Figure 7.2.

The first input file from Table 7.1 was chosen for this evaluation: Epithelial Tissue. The same data from MARGE-S, the Heilman & Smith system, human-authored questions was used but with the addition of 20 randomly selected MARGE-P questions. The results are shown in Table 7.5.

Table 7.5. Average Scores per Source

<table>
<thead>
<tr>
<th>MARGE-S</th>
<th>H&amp;S</th>
<th>Human</th>
<th>MARGE-P</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.85</td>
<td>2.05</td>
<td>2.7</td>
<td>3.08</td>
</tr>
</tbody>
</table>

In this evaluation of the meaningfulness of questions on a 1-4 scale, we can consider questions that score 3 or 4 to be meaningful questions. The Heilman and Smith questions had the lowest average score which is not surprising since their system is not designed to evaluate whether questions are meaningful or not. Both MARGE-S and MARGE-P scored higher than the human authored
questions, which were end-of-chapter questions. By infusing NLU techniques into both MARGE-S and MARGE-P, the generated questions are meaningful questions with high semantic quality. The questions generated by MARGE-P were the only ones that scored above 3.0 on average. The number of end-of-chapter human-authored questions was limited in the source texts used. A more robust evaluation could involve a large set of author-provided questions, if available, which could serve as a gold standard.
CHAPTER 8

SUMMARY OF THIS WORK AND LOOKING FORWARD

The research question that drives this research is: *Can infusing NLU techniques into QG lead to higher quality questions compared to approaches that employ syntactic manipulation alone, and can these techniques move the state of the art closer to the quality of human-authored questions?*

Our evaluations demonstrated that MARGE questions do indeed approach the quality of human-authored questions, and we attribute this success to the infusing of NLU techniques into the QG approach. MARGE has advanced the state of the art of question generation to the point that the majority of output questions are of high linguistic quality and also meaningful questions with high semantic quality. Specifically:

- MARGE-S outputs 55% acceptable questions without ranking compared to 52% for the top 20% of the ranked questions in the current state-of-the-art system.
- MARGE-S produces 72% acceptable questions after ranking, a 71% increase over the prior state-of-the-art system.
- MARGE-S questions on average scored 3.7 on a 5-point scale which is a significant improvement over the prior state-of-the-art 2.9 average, and moves close to the 4.4 human-authored average.
- MARGE-P is the first QG system to successfully break through the sentence barrier. In prior QG systems, including MARGE-S, the answer to a question is found within one sentence. In contrast, the answer for MARGE-P questions must be synthesized from an average of 33% of the passage sentences.
- When evaluating question meaningfulness and importance on a 4-point scale, MARGE-S questions scored 2.85 compared to the prior state-of-the-art 2.05. MARGE-P questions proved competitive with human-authored end-of-chapter questions.

This research focused on the generation of quality factual and conceptual questions. Even though the system does not overgenerate in the manner that prior work does, it is still prolific. This brings forth an interesting area of future research: how to select questions from the pool of available questions. This delves into an area combining educational psychology and cognitive science with computer science so an interdisciplinary approach would be warranted.
APPENDIX A

SAMPLE MARGE-S SENTENCE-GENERATED QUESTIONS
Table A.1 shows a sample source sentence and generated question for the sentence patterns commonly found in expository text.

<table>
<thead>
<tr>
<th>Pattern and Sample</th>
<th>Text sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. <strong>S-V-acomp</strong></td>
<td>Epithelial tissues provide the body’s first line of protection from physical, chemical, and biological wear and tear.</td>
</tr>
<tr>
<td>Adjectival complement that describes the subject.</td>
<td>The cells of an epithelium act as gatekeepers of the body controlling permeability and allowing selective transfer of materials across a physical barrier.</td>
</tr>
<tr>
<td><strong>S:</strong> Brain waves during REM sleep appear similar to brain waves during wakefulness.</td>
<td>All substances that enter the body must cross an epithelium.</td>
</tr>
<tr>
<td><strong>Q:</strong> Indicate characteristics of brain waves during REM sleep.</td>
<td>Some epithelia often include structural features that allow the selective transport of molecules and ions across their cell membranes.</td>
</tr>
<tr>
<td>2. <strong>S-V-attr</strong></td>
<td>Many epithelial cells are capable of secretion and release mucous and specific chemical compounds onto their apical surfaces.</td>
</tr>
<tr>
<td>Nominal predicative complement following copula, often defining the subject.</td>
<td>The epithelium of the small intestine releases digestive enzymes, for example.</td>
</tr>
<tr>
<td><strong>S:</strong> The entire eastern portion of the Aral sea has become a sand desert, complete with the deteriorating hulls of abandoned fishing vessels.</td>
<td>Cells lining the respiratory tract secrete mucous that traps incoming microorganisms and particles.</td>
</tr>
<tr>
<td><strong>Q:</strong> How would you describe the entire eastern portion of the Aral sea?</td>
<td>A glandular epithelium contains many secretory cells.</td>
</tr>
<tr>
<td>3. <strong>S-V-ccomp</strong></td>
<td>Irrigation systems have been updated to reduce the loss of water.</td>
</tr>
<tr>
<td>Clausal complement indicates a proposition of or about the subject.</td>
<td>A glandular epithelium contains many secretory cells.</td>
</tr>
<tr>
<td><strong>S:</strong> Monetary policy should be countercyclical to counterbalance the business cycles of economic downturns and upswings.</td>
<td></td>
</tr>
<tr>
<td><strong>Q:</strong> What evidence could support the notion that monetary policy should be countercyclical?</td>
<td></td>
</tr>
<tr>
<td>4. <strong>S-V-dobj</strong></td>
<td>The early portion of stage 1 sleep produces alpha waves.</td>
</tr>
<tr>
<td>Indicates the relation between two entities.</td>
<td>The 1828 campaign was unique because of the party organization that promoted Jackson.</td>
</tr>
<tr>
<td><strong>S:</strong> The early portion of stage 1 sleep produces alpha waves.</td>
<td><strong>Q:</strong> Why was the 1828 campaign unique?</td>
</tr>
<tr>
<td><strong>Q:</strong> What does the early portion of stage 1 sleep produce?</td>
<td></td>
</tr>
<tr>
<td>5. <strong>S-V-iobj-dobj</strong></td>
<td>The Bill of Rights gave the new federal government greater legitimacy.</td>
</tr>
<tr>
<td>Indicates the relation between three entities.</td>
<td></td>
</tr>
<tr>
<td><strong>S:</strong> The Bill of Rights gave the new federal government greater legitimacy.</td>
<td></td>
</tr>
<tr>
<td><strong>Q:</strong> What gave the new federal government greater legitimacy?</td>
<td></td>
</tr>
<tr>
<td>6. <strong>S-V-pparg</strong></td>
<td>REM sleep is characterized by darting movement of closed eyes.</td>
</tr>
<tr>
<td>Prepositional phrase that is required to complete the meaning.</td>
<td></td>
</tr>
<tr>
<td><strong>S:</strong> REM sleep is characterized by darting movement of closed eyes.</td>
<td></td>
</tr>
<tr>
<td><strong>Q:</strong> What is REM sleep characterized by?</td>
<td></td>
</tr>
<tr>
<td>7. <strong>S-V-xcomp</strong></td>
<td>Irrigation systems have been updated to reduce the loss of water.</td>
</tr>
<tr>
<td>Non-finite clause-like complement.</td>
<td>The 1828 campaign was unique because of the party organization that promoted Jackson.</td>
</tr>
<tr>
<td><strong>S:</strong> Irrigation systems have been updated to reduce the loss of water.</td>
<td><strong>Q:</strong> Why was the 1828 campaign unique?</td>
</tr>
<tr>
<td><strong>A:</strong> For what purpose have the irrigation systems been updated?</td>
<td></td>
</tr>
<tr>
<td>8. <strong>S-V</strong></td>
<td>The 1828 campaign was unique because of the party organization that promoted Jackson.</td>
</tr>
<tr>
<td>May contain phrases that are not considered arguments such as ArgMs.</td>
<td></td>
</tr>
<tr>
<td><strong>S:</strong> The 1828 campaign was unique because of the party organization that promoted Jackson.</td>
<td></td>
</tr>
<tr>
<td><strong>Q:</strong> Why was the 1828 campaign unique?</td>
<td></td>
</tr>
</tbody>
</table>
Ranked MARGE-S Questions

Each question below is preceded by the number of the sentence from which it was generated, and its TextRank score.

26 0.5 What does a glandular epithelium contain?  
19 0.4 What do epithelial tissues provide?  
23 0.375 What are capable of secretion and release mucous and specific chemical compounds onto their apical surfaces?  
23 0.34 Indicate characteristics of many epithelial cells.  
22 0.0 List what some epithelia often include.

Comments

Note that not every sentence resulted in a generated question. In Sentence 20, the parse misidentified the main verb as controlling instead of acts but MARGE detected a questionable parse and did not generate a question from this parse. A similar problem occurred in Sentences 21 and 25. In Sentence 24, the parser could not identify the subject. MARGE detected this problem as well and did not generate a question from this sentence. Sentence 23 has two independent clauses and therefore a question was generated for each independent clause (branch in MARGE terminology).

In this small sample, 4 of the 8 input sentences did not parse correctly. The average for SPLAT is closer to 25% in our observations. Since SPLAT is a relatively new parser it is hoped that improvements over time will be seen.

Parsing and Meaning Analysis Representation Output

MARGE produces a detailed output file of results for development purposes. For every sentence, the SRL and dependency parse is shown, along with error conditions, branch patterns, branch constituents and generated questions. The following is the output for Sentence 26.

Sentence 26: section:2 A glandular epithelium contains many secretory cells.

Error conditions:

SRL data:
branch1->{'A1': [[5, 7]], 'A0': [[1, 3]], 'pred:0': ['contains:3']}
Branch root indices, cc, types = [[1, 7]], [], ['regular']

------------------ Branch:
Branch pattern: pred|dobj
Branch subject info [plural, head word, head pos, vague]: False, epithelium, NN, False

subject:3 = a glandular epithelium [1,3],4
dobj:7 = many secretory cells [5,7],4
pred:4 = contains [4,4],0

Generated questions
  template: 17 dobj1a
What does a glandular epithelium contain?
Many secretory cells
  Question rank: 0.5
APPENDIX B

TEMPLATES AND DESCRIPTIONS
This appendix provides the full list of templates used for MARGE-S question generation, along with explanations. Documentation is provided within the template file, a summary of this documentation is given here. Templates have 6 fields, delimited by a semi-colon:

**Field 1: label**
Each template is given a unique label so that statistics can be gathered for development and evaluation purposes.

**Field 2: sentence types**
This is a list of sentence types that match the template. Types of sentences include: regular, passive, perfect, progressive, and existential. If this field contains '*', then it matches any sentence type.

**Field 3: pattern**
This is the sentence pattern described in Chapter 4. An example of a pattern is S-V-acomp, meaning that the subject and verb are followed by an acomp constituent. Templates were designed to ask about the most important content conveyed by each pattern. A sentence is composed of one or more independent clauses, and each clause has its own pattern. The templates listed on the following pages are organized by sentence pattern.

**Field 4: requirements and filters**
These are conditions that must, or cannot, be present in the source sentence. The requirements and filters options are numerous, and are documented in the template file as well as in the matcher.py program that is responsible for matching templates to sentences. An example is subject=VBG which means that the head word of the subject constituents must have POS VBG. In contrast, subject!VBG means that the subject head word POS cannot be VBG.

**Field 5: surface form**
The surface form of the template consists of literal text and constituents, possible modified. Any text within pipe symbols is a constituent. A simple example is |init_phrase| describe |subject|. The subject constituent is placed in the appropriate spot, as is any
optional initial phrase. Adding initial phrases was shown to give more context to generated questions, as in: *In contrast with the tight and anchoring junctions, what does a gap junction form?*

**Field 6: answer**

This field identifies the constituent that provides the answer to the question.

The templates on the following pages have been broken into two or more lines for readability purposes, with the secondary lines indented. In the templates file each template would be on one line.
# -- S-V-acomp
acomp1a;regular;pred|acomp=JJ;subject!vague,subject!VBG,acomp+NN,!aux;
   |init_phrase| indicate characteristics of |subject|pp|.;acomp
acomp1b;regular;pred|acomp=JJ;subject=VBG,acomp+NN,!aux;
   |init_phrase| indicate a function or purpose of |subject|pp|.;acomp
acomp1c;regular;pred|acomp=JJ;!aux;
   |init_phrase| what-who|verb|acomp|pp|?;subject
acomp2;regular;pred|acomp=JJR;
   |init_phrase| subject|verb|acomp@|than what?;subject
acomp3;regular;pred|acomp=VBN;subject+NN;
   |init_phrase| describe |subject|.;acomp
acomp4;passive;pred|acomp=VBN;subject+NN;
   |init_phrase| describe |subject|.;acomp
acompMNR1;regular;pred|acomp=JJ;MNR,!CAU;
   |init_phrase| what |subject@|verb|neg|acomp|?;subject
acompCAU1;regular;pred|acomp=JJ;CAU;
   |init_phrase| why |verb|subject|neg|acomp<CAU|?;CAU
acompPNC1;regular;pred|acomp=JJ;PNC;
   |init_phrase| for what purpose |verb|subject|neg|acomp<PNC|?;PNC
# make acomp>xcomp Why verb+ subject acomp? ans=split off xcomp
#
# -- S-V-attr
attr1;regular,perfect;pred|attr;subject!vague;
   |init_phrase| how would you describe |subject|?;attr
attr2;regular;pred|attr;subject=vague;
   what |verb|attr|?;X
attr2per;perfect;pred|attr;subject=vague;
   what |aux|verb|attr|?;X
attr2pas;passive;pred|attr;subject=vague;
   what |auxpass|verb|attr|?;X
attrCAU1;regular;pred|attr,subject!vague;CAU;
   why |verb|subject|neg|attr<CAU|?;CAU
#
# -- S-V-ccomp
ccomp1;regular,perfect;pred|ccomp;V=be,V=let,V=light;
   what evidence could support the notion |ccomp|?;X
ccomp2;regular;pred|ccomp;V=be,subject+NN,subject!LOC;
   |init_phrase| what |verb|subject|?;ccomp
#
# -- S-V-dobj
dobj1a;regular;pred|dobj;dobj!CD,V=light,V=describe,
   V=include,V=叫,!MNR,!CAU,!PNC,subject!vague,!pp>verb;
   |init_phrase| what-who|do|subject|vroot|?;dobj
dobj1b;regular;pred|dobj;dobj!CD,V=call,!MNR,!CAU,!PNC,
   subject+NN,!pp>verb;|init_phrase|define |dobj|.;subject
dobjjc;regular;pred|dobj;dobj!CD,V=include,!MNR,subject+NN,
   !pp>verb;
   |list what |subject|verb+|.|;dobj
CAU1;regular;pred;V!be,CAU,subject!vague;
    |init_phrase| why |do|subject|vroot|pp|?;CAU
CAU2;passive;pred;V!be,CAU,subject!vague,!aux;
    |init_phrase| why |auxpass|subject|verb|pp|?;CAU
PNC1;regular;pred;V!be,PNC,subject!vague;
    |init_phrase| for what purpose |do|subject|vroot|pp|?;PNC
PNC2;passive;pred;V!be,PNC,!aux,subject!vague;
    |init_phrase| for what purpose |auxpass|subject|verb|pp|?;PNC
#
# -- advcl_
#when1;*;*;advcl_when;what can you conclude |advcl_when|?;S
# possibly add req for nsubj dep on head of advcl_when
when1;*;*;advcl_when;
    what can happen to |subject-object|advcl_when|?;S
if1;*;*;advcl_if,!dobj,!aux;
    what |happen|advcl_if|?;S
as1;*;*;advcl_as;
    what |happen|advcl_as|?;S
whereas1;*;*;advcl_whereas;
    |subject|can be contrasted with what|?;S
#
# two-branch questions
# The text states: <branch1>. However, s verb+ what? for dobj
#
# existential - 33
# Provide a detailed discussion of the statement: S?
exist1;existential;pred;;
    provide details supporting the assertion that there |verb|subject|.;X
#
# questions in text - just repeat them as long as 'you' is not the subject
#
# two-branch questions
#
but1;*;*;BR-but;
    What qualification would you add to the statement: |branch1|?;S
#

APPENDIX C

SAMPLE MARGE-P PASSAGE QUESTIONS
This appendix lists MARGE-P questions for the file *Stages of Sleep* in the evaluation corpus. Column 1 indicates the section from which it was generated, and column 2 lists the question. The passage sections are as follows:

(1) Stages of Sleep
(2) NREM Stages of Sleep
(3) REM Sleep
(4) Dreams

**Table C.1. MARGE-P Questions**

<table>
<thead>
<tr>
<th>Section</th>
<th>Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Describe the relation between brain wave activity and stages of sleep.</td>
</tr>
<tr>
<td>1</td>
<td>Describe the relation between brain waves and stages of sleep.</td>
</tr>
<tr>
<td>1</td>
<td>Explain what you learned about brain waves in this passage.</td>
</tr>
<tr>
<td>1</td>
<td>Explain what you learned about wave activity in this passage.</td>
</tr>
<tr>
<td>1</td>
<td>Summarize what you learned about stages of sleep in this section.</td>
</tr>
<tr>
<td>2</td>
<td>Provide a definition for hz, and discuss its relation to nrem stages of sleep.</td>
</tr>
<tr>
<td>2</td>
<td>Differentiate between brain waves and alpha waves.</td>
</tr>
<tr>
<td>2</td>
<td>Describe the relation between brain wave activity and nrem stages of sleep.</td>
</tr>
<tr>
<td>2</td>
<td>Explain what you learned about brain activity in this passage.</td>
</tr>
<tr>
<td>2</td>
<td>Summarize what you learned about nrem stages of sleep in this section.</td>
</tr>
<tr>
<td>3</td>
<td>Describe the relation between emotional processing and rem sleep.</td>
</tr>
<tr>
<td>3</td>
<td>Describe the relation between sleep deprivation and rem sleep.</td>
</tr>
<tr>
<td>3</td>
<td>Explain what you learned about rem sleep in this passage.</td>
</tr>
<tr>
<td>3</td>
<td>Explain what you learned about sleep deprivation in this passage.</td>
</tr>
<tr>
<td>3</td>
<td>Explain what you learned about rem deprivation in this passage.</td>
</tr>
<tr>
<td>3</td>
<td>Summarize what you learned about rem sleep in this section.</td>
</tr>
<tr>
<td>4</td>
<td>Describe the role of a psychiatrist as discussed in this section.</td>
</tr>
<tr>
<td>4</td>
<td>Describe the role of a researcher as discussed in this section.</td>
</tr>
<tr>
<td>4</td>
<td>Describe the role of neuroscientists as discussed in this section.</td>
</tr>
<tr>
<td>4</td>
<td>Describe the relation between collective unconscious and dreams.</td>
</tr>
<tr>
<td>4</td>
<td>Explain what you learned about dream content in this passage.</td>
</tr>
<tr>
<td>4</td>
<td>Summarize what you learned about dreams in this section.</td>
</tr>
</tbody>
</table>
APPENDIX D

SAMPLE TOPIC MODELING RESULTS
This appendix will give an example of topic modeling results. The first section of the file *Stages of Sleep* from the evaluation corpus will be listed, along with the topic keywords.

**Section 1 Text**

Sleep is not a uniform state of being. Instead, sleep is composed of several different stages that can be differentiated from one another by the patterns of brain wave activity that occur during each stage. These changes in brain wave activity can be visualized using EEG and are distinguished from one another by both the frequency and amplitude of brain waves. Sleep can be divided into two different general phases: REM sleep and non-REM (NREM) sleep. Rapid eye movement (REM) sleep is characterized by darting movements of the eyes under closed eyelids. Brain waves during REM sleep appear very similar to brain waves during wakefulness. In contrast, non-REM (NREM) sleep is subdivided into four stages distinguished from each other and from wakefulness by characteristic patterns of brain waves. The first four stages of sleep are NREM sleep, while the fifth and final stage of sleep is REM sleep.

**Document topics and keywords**

The R script for topic modeling outputs the top 6 terms for each topic in a csv file. The following shows the top terms for the 6 topics.

1. general, report, become, degree, earlier, former
2. appear, wakefulness, different, higher, memory, wake
3. rem, deprivation, suggest, effective, emotional, fact
4. stage, wave, activity, pattern, amplitude, frequency
5. dream, content, freud, jung, time, woman
6. sleep, brain, associate, occur, high, movement

Topic 1 was not among the top two topics for any of the 4 sections. For section 1, the top topic was 4 and the second was 6. These are provided by the R script in a csv file, as are the actual probabilities. The probability of topic 4 in Section 1 is 0.312 and the probability of topic 6 is 0.263.

The topic keywords for topics 4 and 6 within section 1 are:
stage-7  wave-6  activity-2  pattern-2  amplitude-1  frequency-1
sleep-14  brain-6  associate-0  occur-1  high-0  movement-2
APPENDIX E

COMPARISON OF MAR AND AMR
AMR (Abstract Meaning Representation)\(^1\) is a project that seeks to translate sentences into meaning representations, which is the same goal as the MAR (Meaning Analysis Representation) produced by the DeconStructure algorithm. In this appendix, a brief overview of AMR is provided, followed by a comparison of AMR and MAR.

**AMR**

AMR is the result of collaboration from many researchers and universities, including Martha Palmer of the University of Colorado, and researchers from Edinburgh, and USC’s Information Sciences Institute. The project has been funded by NSF and DARPA.

AMR uses the PENMAN notation to represent a sentence as a directed acyclical graph. Penman is a system from USC/ISI that generates sentences from non-linguistic sources.\(^2\) In the AMR we have a root note which is the predicate of the sentence. Other nodes are labeled with variables and with concepts.

Figure E.1 shows a simple AMR Example. The nodes each have a label that is a combination of a variable and the head word. The edges are the SRL relations between the nodes. There are also constants for items like negation words or numbers. All concepts are included in the graph without plurality, articles and tense.

---

\(^1\)http://amr.isi.edu/

\(^2\)http://www.isi.edu/natural-language/penman/penman.html
MAR and AMR Sample Sentence

The following shows the representation for the sample sentence: \textit{A glandular epithelium contains many secretory cells.}

MAR Representation

Branch root indices, cc, types = [[1, 7]], [], ['regular']

------------------ Branch:
Branch pattern: pred|dobj
Branch subject info [plural, head word, head pos, vague]: False, epithelium, NN, False

subject:3 = a glandular epithelium [1,3],4
dobj:7 = many secretory cells [5,7],4
pred:4 = contains [4,4],0

AMR Representation

["c / contain :align "4*8" :ARG0 (c / epithelium :align "1,3" :mod (g / glandular :align "2*")) :ARG1 (c2 / cell :align "7*" :mod (m / many :align "5*")) :mod (s / secretory :align "6*"))")]

Figure E.1. AMR Example

Comparison

The main advantage of MAR over AMR is that MAR creates functional/semantic labels for relations whereas the AMR uses SRL Arg labels. The SRL label \texttt{Arg0} is often but not always the subject. In the MAR we can easily retrieve the subject or any other constituent in the sentence. The MAR also gives us important information about the subject: whether or not it is plural, the head word, the head word POS, and whether or not it is vague. All of these are useful for question generation. Information about plurality, tense and so forth is lost in the AMR representation. The MAR also provides the branch pattern used in this work to match templates for question generation.

AMR is built upon a large bank of manually annotated data. The annotation scheme is quite complex but the end result enables some nice features, including wikification of concepts, and normalized entities (five bucks -> 5 dollars). These features are not present in MAR.


Art Graesser, Yasuhiro Ozuru, and Jeremiah Sullins, *What is a good question?*, (2010).


[99] Henry L Roediger and Mary A Pyc, *Inexpensive techniques to improve education:


[105] Rion Snow, Brendan O’Connor, Daniel Jurafsky, and Andrew Y Ng, Cheap and fast—but is it good?: evaluating non-expert annotations for natural language tasks, Proceedings of the conference on empirical methods in natural language processing, Association for Computational Linguistics, 2008, pp. 254–263.


