

Biology Question Generation from a Semantic Network

by

Lishan Zhang

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved November 2015 by the
Graduate Supervisory Committee:

Kurt VanLehn, Chair
Ihan Hsiao
Chitta Baral
Christian Wright

ARIZONA STATE UNIVERSITY

December 2015

ABSTRACT

Science instructors need questions for use in exams, homework assignments, class discussions, reviews, and other instructional activities. Textbooks never have enough questions, so instructors must find them from other sources or generate their own questions. In order to supply instructors with biology questions, a semantic network approach was developed for generating open response biology questions. The generated questions were compared to professional authorized questions.

To boost students' learning experience, adaptive selection was built on the generated questions. Bayesian Knowledge Tracing was used as embedded assessment of the student's current competence so that a suitable question could be selected based on the student's previous performance. A between-subjects experiment with 42 participants was performed, where half of the participants studied with adaptive selected questions and the rest studied with mal-adaptive order of questions. Both groups significantly improved their test scores, and the participants in adaptive group registered larger learning gains than participants in the control group.

To explore the possibility of generating rich instructional feedback for machine-generated questions, a question-paragraph mapping task was identified. Given a set of questions and a list of paragraphs for a textbook, the goal of the task was to map the related paragraphs to each question. An algorithm was developed whose performance was comparable to human annotators.

A multiple-choice question with high quality distractors (incorrect answers) can be pedagogically valuable as well as being much easier to grade than open-response questions. Thus, an algorithm was developed to generate good distractors for multiple-choice

questions. The machine-generated multiple-choice questions were compared to human-generated questions in terms of three measures: question difficulty, question discrimination and distractor usefulness. By recruiting 200 participants from Amazon Mechanical Turk, it turned out that the two types of questions performed very closely on all the three measures.

TABLE OF CONTENTS

	Page
LIST OF TABLES	vi
LIST OF FIGURES	viii
CHAPTER	
1 INTRODUCTION	1
2 HOW DO MACHINE-GENERATED QUESTIONS COMPARE TO HUMAN- GENERATED QUESTIONS?	5
Introduction	5
Generating Questions from a Semantic Knowledge Base.....	11
Collecting Questions by Google.....	29
Coverage of the Machine-generated Questions.....	33
Main Study	37
Discussion.....	50
Conclusion.....	52
3 ADAPTIVELY SELECTING BIOLOGY QUESTIONS GENERATED FROM A SEMANTIC NETWORK.....	55
Introduction	55
Prior Works in Adaptive Task Selection	58
Description of the Tutoring System.....	73

CHAPTER	Page
Evaluation.....	85
Conclusion.....	91
4 MAPPING QUESTIONS TO THEIR RELATED PARAGRAPHS IN THE TEXTBOOK.....	93
Task Definition.....	93
Dataset.....	94
Methodology.....	95
Evaluation.....	102
Discussion.....	108
Related Works.....	109
Conclusion.....	111
5 EVALUATION OF AUTO-GENERATED DISTRACTORS IN MULTIPLE CHOICE QUESTIONS.....	112
Introduction.....	112
Related Works.....	114
Multiple-choice Question Generation.....	115
Evaluation.....	123
Discussion.....	132
Conclusion.....	134

CHAPTER	Page
6 CONCLUSION.....	135
REFERENCES	138
APPENDIX	
A HUMAN-MACHINE QUESTIONS MAPPING	152
B THE SOURCE IN THE SEMANTIC NETWORK FOR SOME MACHINE GENERATED QUESTIONS	164
C MANUALLY CREATED SYNONYMS.....	167

LIST OF TABLES

Table	Page
1. Keywords for Google Search.....	31
2. Overlap of Machine and Human-generated Questions.....	36
3. Mean (and Standard Deviation) of the Measures-original Scores	41
4. Mean (and Standard Deviation) of the Measures-normalized Scores	41
5. ANOVA of Means from Table 4	42
6. Pair by Pair T-tests of Means from Table 3 and Table 4.....	42
7. The Comparison of Pedagogy after Removing Questions with High Inter-rater Variability.	44
8. The Comparison of Depth after Removing Questions with High Inter-rater Variability	44
9. Histograms and Median Value of the Measures	45
10. Quantile Regression Comparisons.....	46
11. Pearson’s Correlation of the Different Features of the Questions.	48
12. Machine vs. Human Confusion Matrix.....	78
13. Joint Probability Distribution for Answering the Questions.....	82
14. Procedure of the Experiment	87
15. Comparison Between Experimental Group and Control Group.....	88
16. Result of Question Mapping (N=54)	104
17. Result of Mapping on Subset (N=33) of Questions.....	106
18. Result of Question Mapping with Different Gold Standards.....	106

Table	Page
19. Result of Question Mapping with Aggregation by the Index in Textbook (N=33)	107
20. Result of Question Mapping with Random Aggregation for Auto-matched Questions (N=33).....	108
21. Summary of Usefulness of Two Example Distractors.....	132
22. Comparison of Machine-generated Distractors to Human-generated Distractors	132

LIST OF FIGURES

Figure	Page
1. Fragment of the Knowledge Base.....	14
2. Piece of the Semantic Network for “What Is” Questions.....	24
3. Piece of the Semantic Network for “Where” Questions.....	26
4. Piece of the Semantic Network for Direct Input Questions.....	27
5. Piece of the Semantic Network for Complicated “Input&output” Questions	28
6. Piece of the Semantic Network for Generating “Function” Questions.....	28
7. Flow of Collecting Questions	30
8. Screenshot of the Experiment	38
9. Mean of Depth for Each Type of Annotations.....	50
10. The System Feedback on Student’s Answer.....	74
11. Analysis of the Student’s Response.....	77
12. Probabilities of Mastery Calculated by BKT.....	83
13. Utility of a Question vs. Probability of Mastery.....	84
14. Sample Test Question	86
15. The Learning Gain of Adaptive treatment Group vs. The Learning Gain of Mal- adaptive Treatment Group	89
16. The Adjusted Post-test Score of Adaptive Treatment Group vs. That of Mal- adaptive Treatment Group.	90
17. Relationships Among Questions, Knowledge Components and Keywords.....	97
18. Entity1 Can Be a Distractor, If Entity2 Is the Correct Answer.	117
19. Entity2 Can Be a Distractor, If Entity1 Is the Correct Answer.	117

Figure	Page
20. A Distractor for Where Questions.	118
21. Process1 and process2 Are Related Processes.....	118
22. Distractors for Connection Questions.....	119
23. Human-generated Multiple-choice Question with Complex Alternatives.....	120
24. Pre-processed Machine-generated Multiple-choice Question	121
25. Final Machine-generated Multiple-choice Question	121
26. Human-generated Multiple-choice Question with Simple Alternatives.....	122
27. Force Participants to Read Questions Carefully	128

CHAPTER 1

INTRODUCTION

In mathematics and similar topics, question generation is widely used perhaps because it is so simple to implement. For instance, an infinite number of questions can be generated from the template “*Solve for x: [num1]x+[num2]=[num3].*” where [num1], [num2] and [num3] are variables in the template that should be assigned to numbers. Such question generators are used for generating homework exercises, quizzes, exams, questions to be asked in class, and many other purposes. In subjects where questions must be answered in natural language, instructors still need a large supply of questions for their homework assignments, quizzes, exams, etc. The goal of the dissertation is to demonstrate how to automatically generate questions whose answers are in natural language and the benefits of having questions be generated in this way.

Previous work disclosed two major methods in question generation in terms of the inputs: from a formal representation of knowledge and plain text. This dissertation focuses on generating questions from a formal representation of knowledge and evaluating the benefits of utilizing this generation method. Photosynthesis is selected as my study domain because a knowledge base content has already been built and available from Baral & Liang (2012). The formal representation of knowledge specifically in this dissertation is a semantic network for photosynthesis.

To use machine-generated questions in student’s learning, the quality of the generated questions should be as good as those from human. So my first study was to compare the qualities of three types of questions: human authored questions from textbooks, questions mined from the web and machine-generated questions from a semantic network. Although

machine-generated questions did not have good coverage of human-generated deep questions, overall there was no significant difference in quality of the three types of questions. The results suggested that machine-generated questions could be potentially used as the replacement of human-generated shallow questions. Shallow questions were defined as those that were answering them only required to remember or understand factual knowledge. This result has practical importance in that it is likely that students need to master shallow questions before being able to profitably tackle deep questions (I-H Hsiao, Sosnovsky, & Brusilovsky, 2010).

My second study demonstrated that machine-generated method facilitates adaptive question selection. The Q-matrix (Tatsuoka, 1996) and student model are two essential components for building adaptive question selection. The Q-matrix records the associations of questions to knowledge components. The student model maintains a record of each student's mastery levels of each knowledge component. A knowledge component is a unit of knowledge in the teaching domain. (Koedinger, Corbett, & Perfetti, 2012) Identifying knowledge components and associating them to their corresponding questions require domain experts to put in a considerable amount of time. The semantic network used for question generation can also construct the set of knowledge components and the Q-matrix. However, it is not clear whether adaptive question selection would be effective when the knowledge components and Q-matrix were constructed by machine. Thus, my second study measured the effectiveness of adaptive question selection using machine-generated questions from the semantic network.

The advantage of machine-generated questions is not limited to adaptive question selection. It also lies in forming a feedback for a question. A good answer to an open

response question should be precise and accurate. Although students may not give a complex answer to a simple question, the feedback on the student's answer is not necessarily limited to the scope of the specific question because the moment right after a student answers a question, especially the question was answered incorrectly, may be an especially good "time for telling" (Schwartz & Bransford, 1998). Indeed, if only minimum feedback is given, this may impede learning. For example, a precise answer to the question "What does photosynthesis need?" should be "light, water and carbon dioxide", while a paragraph containing the answer to this question would also explain how these reactants are used in photosynthesis. Do students bother reading the related paragraphs? According to Kluger and DeNisi (1996), when a student has a clear learning goal and receives a negative feedback, the student tends to put more effort into making up the gaps between the knowledge and the goal. So students could be motivated to read and learn the related paragraphs to the questions that they answered incorrectly. My third study was conducted to explore how questions annotated with corresponding knowledge components can facilitate question-paragraph mapping.

Although open response questions, which require students to type in their answers, have advantages in lowering the probability of guessing and pushing students to think hard, auto-evaluating students' answers to this type of questions often face many technical difficulties and can be inaccurate. Multiple choice questions are more widely used when high accurate auto-grading are required. However, the quality of a multiple-choice question depends strongly on which incorrect answers are supplied as choices. Such answers are called *distractors* or foils. A good distractor may articulate a latent or vague misconception, and thus seduce the student into an incorrect answer that leads to an especially productive

learning opportunity. So the last study evaluates the quality of multiple-choice questions generated from a semantic network. In particular, it compares the machine-generated distractors to human-generated distractors.

The rest of the dissertation is divided into five chapters. Chapter 2 to Chapter 5 describe the four studies respectively, and Chapter 6 provides a summary and conclusions. More specifically, Chapter 2 evaluates machine-generated questions from a semantic network by comparing the questions to web-mined questions and professional authorized questions. Chapter 3 demonstrates how to take the advantage of using machine-generated questions from a semantic network to implement adaptive question selection. Chapter 4 describes a question-paragraph mapping task, and evaluates how well the machine-generated questions annotated with corresponding knowledge components can facilitate this task. Chapter 5 describes how to generate distractors from a semantic network for multiple choice questions, and compares these machine-generated distractors to human-generated distractors in terms of three measures: question difficulty, question discrimination, and distractor usefulness. Chapter 6 draws conclusions from these studies.

CHAPTER 2

HOW DO MACHINE-GENERATED QUESTIONS COMPARE TO HUMAN-GENERATED QUESTIONS?

2.1 Introduction

2.1.1 Techniques of question generation

A recent review of educational systems with automatic question generation (Le, Kojiri, & Pinkwart, 2014) noted that there are two major techniques in terms of sources: generating from plain text or from a formal representation of knowledge.

Automatic generation of questions from plain text converts declarative sentences in a text into questions. One of the first systems of this type, AUTOQUEST (Wolfe, 1976), helped students check their understanding of readings by generating questions on each sentence. The sentence was first parsed using a natural language parser, the parsed result was matched to a predefined pattern, a template was selected, and a question was generated by filling in variable in the template with values from the pattern. For example, suppose the original sentence was “John bought some fruits”. The corresponding parse tree was:

```
(ROOT
  (S
    (NP (NNP John))
    (VP (VBD bought)
      (NP (DT some) (NNS fruits))))))
```

When the subject of the sentence is being questioned, the question “Who bought some fruits?” was generated. When the object of the verb phrase was being questioned, the question “What did John buy?” was generated. This general procedure has been used many times (Ali, Chali, & Hasan, 2010; Heilman & Smith, 2009; Kalady, Elikkottil, & Das,

2010; Varga; Wyse & Piwek, 2009). Patterns and templates can be hand authored (Ali et al., 2010; Kalady et al., 2010) or learned from given question-answer pairs (Curto, Mendes, & Coheur, 2012).

The second technique generates questions from a formal representation of knowledge. Perhaps the first system of this type, the SCHOLAR system (Carbonell, 1970), represented knowledge as a semantic network and used heuristics for generating questions from it. Later work focused on providing better assessments of students (Lazarinis, Green, & Pearson, 2010) and adaptively selecting questions (Dillenbourg & Self, 1992). The most recent work (Jouault & Seta, 2014) was a history tutoring system for reviewing the chronology of important events.

Hybrids of the two basic approaches have been explored as well. For instance, Liu and Calvo's (2012) system first identified key concepts, which were essentially a set of noun phrases, in a student's essay. Next, three types of relationships (Different-to, Similar-to, Kind-of) among the key concepts were extracted from Wikipedia by using Tregex pattern matching rules on Wikipedia articles. Finally those relations were used to generate questions based via expert-made templates. Although the system dealt with concepts and relationships, it didn't reason with them. The main effort lay in identifying the key concepts and finding the corresponding relations.

Olney, Graesser and Person (2012) also used a hybrid method to generate questions in Biology. Their system first extracted a concept map from texts, where a concept map is a particularly simple type of semantic net. They then used templates to generate questions either from single links in the concept map or from small clusters of specific links. Once

again, the emphasis was on extracting knowledge from the text rather than reasoning with the knowledge once it was extracted.

A third hybrid system (Chen, 2009) reasoned about knowledge it extracted from text in order to generate questions about the mental states of characters in stories. The story was first transformed to a set of semantic relations in the Scone language (Fahlman, 2006). New relations were then inferred by using everyday knowledge (e.g., the definition of “pretend”) already stored in Scone knowledge base, so that the mental states could be described more accurately. Finally, yes/no questions were generated to help check readers’ understanding of the characters’ mental states in the story.

Instead of generating questions, another way to quickly obtain a large number of questions is to harvest them from the web. The process is to first get more than enough questions and then rule out the low quality ones. Section 3 describes how we harvested questions using Google’s key word search.

2.1.2 Evaluation of question generation systems

Now that the feasibility of machine-generated questions is established, the next issue is evaluating their quality compared to human-generated questions. Only a few studies have addressed this issue. Liu and Calvo (2012) compared their machine-generated questions to human questions in terms of 5 quality measures (i.e. correctness, clarity, relevancy, useful for learning concepts and useful to improve documents) rated by 23 human students. The students were required to write essays at first, and then rate the questions that helped to improve their essays. The machine-generated questions were specific to the students’ essays, but the human questions were generic to all the essays. Therefore, this comparison confounded the machine vs. human contrast with the specific vs. generic contrast.

Although comparison of machine-generated questions to human-generated questions are rare, several studies have compared different kinds of machine-generated questions to each other, and an evaluation method has come to be widely accepted.

In order to compare questions produced by different question generation teams, (Rus et al., 2010) identified five criteria for humans to use in judging the quality of questions: syntactic correctness, ambiguity, relevance, question type, and variety. Although the first two criteria, *syntactic correctness* and *ambiguity*, are probably clear, the other three criteria need explanation. Because the generation task required generating questions from given texts, *relevance* measured the match of the question to its source in the text. Both *question type* and *variety* rewarded systems that generated questions in a variety of formats. For instance, a system that merely prepended “Is it the case that” to every sentence would get low marks for both question type and variety.

Heilman and Smith (2009) came up with similar criteria for ranking the questions generated from Wikipedia and also discussed an important additional issue: over-generation. They listed 7 deficiencies that an over-generated question could have. Questions that contained any of them were treated as over-generated questions. Judgments on 644 machine-generated questions were done by three researchers. On average, 87.2% of the questions were over-generated, but there was a low inter-rater agreement (.42) according to Fleiss’s κ . This work treated over-generation more from a syntactic perspective than a semantic perspective, in part because the machine-generated questions were not assumed to be used for instruction.

The rating systems discussed so far did not directly address the utility of the questions for learning. A pedagogically useful question should prompt deep thought and learning

about the domain. For example, most people can correctly answer “Yes” when asked “Is a vacuole wall a part of a vacuole?” even when they have never heard of a vacuole. Thus, assessing the learning caused by questions should be viewed as a separate measure from question quality.

Learning gains have sometimes been used to evaluate questions. For instance, Beck, Mostow and Bey (2004) found statistically reliable improvements in reading comprehension after wh-questions (what/where/when questions). Their study used a within-subject design and logistic regression to predict student’s performance in reading comprehension as a function of the number of wh-questions.

2.1.3 Research question

Our primary interest was in finding ways to help students learn declarative knowledge domains like Biology. Such domains require understanding many concepts and relationships, and answering questions is a common way to exercise and assess such understanding. Unfortunately, the number of questions in textbooks is extremely limited, and teachers usually have little time to invent their own questions. So there is also a need for machine-generated questions in this domain.

Our research question was simple: Are machine-generated biology questions as good as human-generated biology questions according to human judges? We focused on questions for a target population of college students in introductory biology classes.

We assume that questions that are not grammatically correct, ambiguous, confusing or are poor at communicating in some other way are probably not going to help students learn biology. However, even if the questions are high quality on these measures, they may or may not help students learn biology. Although we did not measure learning gains in this

experiment, we did ask judges to rate the pedagogical benefits of the question. This was intended to be a weak proxy for measuring learning gains.

To do the comparison, we first implemented a method for machine-generating questions from a biology knowledge base. A second set of questions was formed by selecting questions retrieved from the web. The authors of most web questions were probably students and not biology experts. Our third set of questions was comprised of professional human-generated questions from textbooks and a biology study website¹. University biology students were recruited to act as judges. They rated the three sets of questions on four measures. Our primary question was whether the machine-generated questions were worse than the human-generated ones. We were also interested in whether the two types of human-authored questions were of different quality.

In order to keep the project feasible, all questions concerned photosynthesis at an introductory college level. Although the methods could easily have generated more questions, evaluation of the questions was made easier for the human judges by using just one topic and allowing them to review the topic before judging the questions.

The rest of the chapter is organized as follow. First, we describe how questions were generated from the biology knowledge base. Second, we describe how questions were collected via web search and how the irrelevant questions were filtered out. Lastly, we describe the comparison of these two types of questions to the professional human-generated questions.

¹ <http://www.biology-questions-and-answers.com>

2.2 Generating questions from a semantic knowledge base

Although recent work on question generation has generated semantic knowledge bases from text (A. Olney, A. C. Graesser, & N. Person, 2012), we chose to use an existing knowledge base corpus. Although few such knowledge bases are currently available, the technology and the market for such knowledge bases are increasing steadily ("ConceptNet," ; Foxvog, 2010; "Wikidata,"). Thus, we believe that it is only a matter of time before knowledge bases are available that are sufficient for high school basic sciences.

There is no uniform way of structuring a semantic knowledge base, so our methods are limited somewhat to the particular knowledge base used for experimentation. The semantic knowledge base we chose was from Baral and Liang (2012). They used the knowledge base to develop a question-answering system for biology (Baral, Vo, & Liang, 2012). For the purpose of evaluating our generated questions, we limited our copy of the knowledge base to photosynthesis.

We also cleaned up the knowledge base manually by taking out redundant information and removing inconsistencies. More specifically, the predicate *cloned_from*, which was used to clone the attributes from one instance to another, was removed from the original knowledge base, because the predicate was potential to introduce redundant attributes for an instance. Rather than simply removing all the relations that contained this predicate, we manually went over each of these relations and copied over the attributes to make sure each instance was complete and concise. The hierarchical information was also cleaned up. For instance, *light reaction*, *light reaction involving noncyclic electron flow* and *light reaction involving cyclic electron flow* were used to be three sub-events of *photosynthesis*, whereas

the latter two processes were also the sub-events of the first process. The latter two processes were removed from the subevents of *photosynthesis* to make the hierarchy clear.

The rest of the section first introduces the knowledge base and then explains how we generated questions from it.

2.2.1 The knowledge base

The knowledge base distinguishes classes and instances. Classes define an ontology, that is, a hierarchical classification of all that exists. For example, *photosynthesis* is a sub-class of *chemical process*, and *chemical process* is a sub-class of *event*. *Event* is one of the root classes, which doesn't have any parent classes. Another root class is *entity*. For instance, *chloroplast* is a distant sub-class of *entity*. On the other hand, instances represent specific objects or specific events. A triple composed of two instances and one predicate form a relation, which is written as the following format:

```
has(photosynthesis001, result, oxygen001)
has(photosynthesis001, instance_of, photosynthesis)
has(oxygen001, instance_of, oxygen)
```

In the first line, *result* is a predicate, and *photosynthesis001* and *oxygen001* are instances. The second and third lines indicate that they are instances of the classes *photosynthesis* and *oxygen*, respectively.

There are features common to every instance of a class. For example, any instance of *photosynthesis* can produce oxygen. These are not properties of the class *per se*, so a special instance named a prototype instance is used to represent relations common to all members of a class. Any other instance of a class is assumed to have the relations involving

the prototype instance of the class as well as the relations in the prototype instances of the super-classes. A class can only have one prototype instance.

The relations directly stored in the knowledge base are called atomic relations. Some relations are inferred during the question generation process, and those are called secondary relations. Similarly, the pre-existed predicates in the knowledge base are called as atomic predicates, and those that are introduced by reasoning only are called secondary predicates. The meanings of atomic predicates are defined in KM (Clark, Porter, & Works, 2004).

According to the most recent version of Bloom's Taxonomy (Krathwohl, 2002), because the atomic relations in the knowledge base only recorded basic elements in photosynthesis, they essentially represented the required factual knowledge in the domain. The inferred secondary relations somehow represented pieces of conceptual knowledge.

2.2.2 Question generation from seed questions

Our first attempt was to generate questions from a user given "seed" question of the form "what is the difference between *A* and *B*?" The strategy was to design an algorithm that would input a seed question, generate a set of constraints that characterized the relationship between concepts *A* and *B*, then output all pairs of concepts that met the constraints, thus generating more questions of the form "what is the difference between *A* and *B*." Although we tested the algorithm in biology, the process was domain independent in principle. The rest of this section first introduces the details of the algorithm and then discusses its performance.

Preparation

In the knowledge base, concepts are represented as classes and the relations of the concepts are represented in terms of prototype instances, so we extracted all the relations of the prototype instances and directly attached them to the concepts. Figure 1 illustrates the resulting knowledge base.

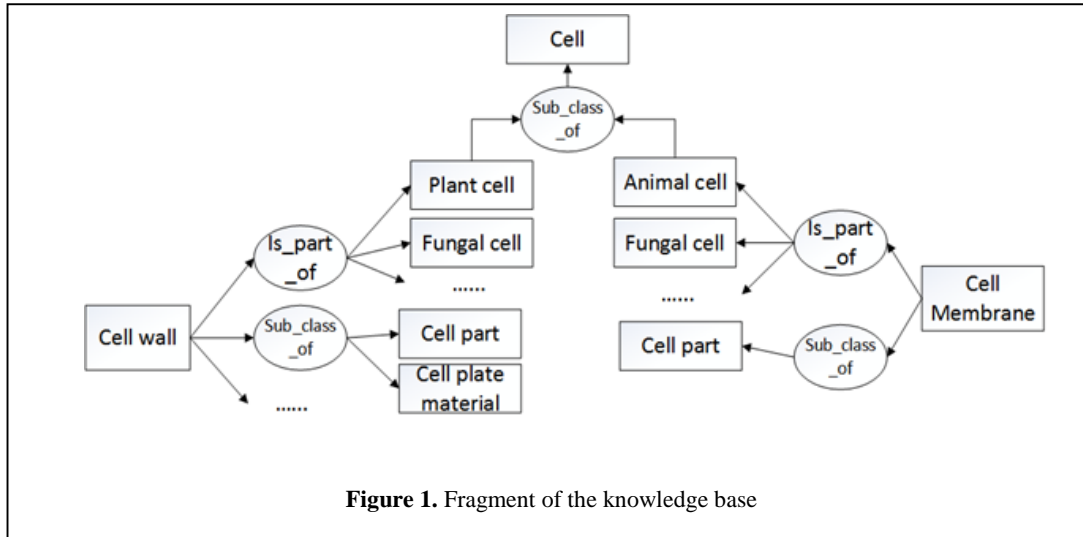


Figure 1. Fragment of the knowledge base

Constraints induction

Our basic approach was based on similarity. Given a user-supplied seed question, such as “What is the difference between *cell walls* and *cell membranes*?”, A and B comprised a good question if their similarity score (A:B) was larger than the similarity score of the human-provided seed question concepts (cell wall: cell membrane). Note that there has been many methods developed for calculating semantic similarity, often based on the ontology in WordNet (Banerjee & Pedersen, 2002; Budanitsky & Hirst, 2001; Jiang & Conrath, 1997; Leacock & Chodorow, 1998; D. Lin, 1998; Resnik, 1995). Although the score we calculated was also called similarity, it was a completely different measure. For instance, *dog* and *hunting dog* are highly semantical related, but the pair was clearly not a good fit for our question template.

We defined two measures. One measured direct similarity and the other measured indirect similarity. As an illustration of these two kinds of similarity, consider Figure 1. It shows that both of *cell wall* and *cell membrane* are part of *Fungal cell*. The direct similarity counts such connections. As an illustration of indirect similarity, note that *Cell wall* is part of *Plant cell* and *Cell membrane* is part of *Animal cell*, and both *Plant cell* and *Animal cell* are sub-classes of *Cell*. Indirect similarity counts these kinds of relations.

Define the direct similarity

The most straightforward way to measure the similarity between two concepts is counting the number of features they have in common and dividing by the average number of features they each have. However, a feature contains both a predicate and an object, and some predicates appear frequently connecting many different objects whereas other predicates appear seldomly. Thus, we calculated similarity on a per-predicate basis, and used the resulting vector of predicate-number pairs in the subsequent question generation algorithm.

The formula below is to used to calculate the per-predicate similarity:

$$similarity(concept1, concept2, pred) = \frac{2 \times \text{shared}}{\text{total}}$$

$$\text{shared} = |\text{neighbors}(\text{concept1}, \text{pred}) \cap \text{neighbors}(\text{concept2}, \text{pred})|$$

$$\text{total} = |\text{neighbors}(\text{concept1}, \text{pred})| + |\text{neighbors}(\text{concept2}, \text{pred})|$$

$$\text{neighbors}(\text{concept}, \text{pred}) = \{x | \text{pred}(x, \text{concept})\}$$

The similarity value is in the range [0,1], where 1 means that the two concepts are exactly the same in terms of the predicate, and 0 means that they have nothing in common.

Define the indirect similarity

Indirect similarity means that two concepts are connected by the same predicate to two different objects (e.g., *plant cell* and *animal cell* in Figure 1) that have the same ancestor (*cell* in Figure 1). Again, we calculated the portion of shared objects for each shared predicate, and stored a vector of predicate-number pairs to represent their indirect similarity. The equation below defines the per-predicate indirect similarity measure:

$$\text{indirect_similarity}(\text{concept1}, \text{concept2}, \text{pred}) = \frac{|(m,n)|}{\text{pairs_total}}$$

$$(m,n) = \{m,n | m \in (\text{neighbors}(\text{concept1}, \text{pred}), n \in$$

$$(\text{neighbors}(\text{concept2}, \text{pred}), m \neq n, \exists p \text{ subclass_of}(m, p) \ \& \ \text{subclass_of}(n, p))\}$$

$$\text{pairs_total} = |\text{neighbors}(\text{obj1}, \text{pred})| \times |\text{neighbors}(\text{obj2}, \text{pred})|,$$

$$\text{neighbors}(\text{concept}, \text{pred}) = \{x | \text{pred}(x, \text{concept})\}$$

However, calculating $|(m,n)|$ is not trivial. A brute force way is for each shared predicate of the two concepts, for each pair of different objects connected to the concepts by that predicate, enumerate all the ancestors of the two objects and see whether they have an ancestor in common. If they do, a counter is increased by 1, otherwise the counter stays the same as before. When all pairs for the predicate are explored, the value of the counter is $|(m,n)|$.

This algorithm is inefficient because it repeatedly finds all the ancestors for an object. If the same object is visited x times, the ancestors set will be generated x times. Thus, we cached the ancestor set calculation: When the ancestors of an object are first generated, the

set is stored as a *s* property of the object. The ancestor sets turned out to be small enough that this was tractible, so caching reduced the computation from about 17 hours to 15 minutes for generating questions for one seed question.

Definition of the constraints

Given two vectors of predicate-number pairs--one that measures the direct similarity of a pair of concepts and the other that measures the indirect similarity--the constraints for “What is the difference between A and B?” is defined below:

$Question(A, B) \equiv$

$$\forall p \text{ similarity}(A, B, p) > \text{similarity}(seed1, seed2, p)$$

and

$$\text{indirect_similarity}(A, B, p) > \text{indirect_similarity}(seed1, seed2, p)$$

In principle, this calculation is run for every possible pair of concepts, A and B. However, if there are predicates *p* such that *similarity(A,B,p)* is zero but *similarity(seed1, seed2,p)* is non-zero, the calculation does not need to be run because it will be false for this A,B pair.

2.2.3 Schema induction needs human intervention

We used 8 seed questions to test the algorithm, one for each of these 8 pairs: anabolism & catabolism, genotype & phenotype, gill & lung, mitosis & meiosis, nematode & annelid, plasma membrane & cell wall, spore & gamete, transcription & translation

The algorithm generated an average of 842 questions per seed questions, and the standard deviation was very high: 1721. Several seed question will be discussed in turn.

The first seed pair was cell wall & plasma membrane. This pair caused generation of 8 new questions plus the original seed question. Of the 8 new questions, 6 were reasonable biology questions based on our (limited) experience. Moreover, 2 of the 6 questions were asked by someone else before, as confirmed using Google search. The other 4 did not appear in a Google search, which was interesting. Thus, cell wall & plasma membrane is an example of a good seed question.

The second seed pair was anabolism & catabolism. The generator found no new questions; it generated only the seed question. This occurred because the restrictions generated from the seed question contained some uncommon predicates.

The third seed pair was gill & lung. The generator's output was 4750 questions. Most of the questions would not make good comparisons. This seed question led to few restrictions, which caused too many pair of concepts be output.

The second and third seed pairs are representative of 7 out of the 8 questions. Except for the cell wall and cell membrane seed pair, all the seed pairs led to either too many or too few generated questions.

These results indicate that the number and the quality of the questions generated from the constraints varied substantially. The variance could be due to the algorithm, the seed questions, and/or the knowledge base. Every seed question is a good comparison question itself. But the knowledge base sometimes didn't have adequate descriptions for the concepts that appeared in the seed questions, and this led to over-generation. Some concepts in the seed question were attached to uncommon predicates in the knowledge

base, and it led to under-generation. It suggested that our method of inducing constraints from seed questions was unstable in part because it depended too strongly on the detailed content of the knowledge base.

These results suggested that any technique that relied on traversing large parts of the knowledge base in an attempt to generate deep, thought-provoking questions was likely to be as fragile as our method. Knowledge bases are just very large and very complex. Nonetheless, instructors still need questions to help their students exercise their knowledge. We decided that some help was probably better than no help, so we decided to focus on generating questions that tapped only small amounts of knowledge. In surveying textbooks, we had noticed that about half their questions were shallow, factual ones (this was confirmed later, as discussed below). This suggested that although instructors probably put a high value on deep questions, they probably also want students to have enough factual knowledge to be able to tackle deep questions, so they probably ask shallow questions as preparation for the deeper ones. If high quality shallow questions could be generated, then this would remove a load from the instructors and authors, thus allowing them to focus on generating deeper questions. As the remainder of this document demonstrates, even the comparatively modest goal of generating moderately shallow questions proved challenging.

2.2.4 Preparing for the generation

As mentioned earlier, we choose to study questions about a single topic so that the human judges could more easily refresh their biology knowledge before judging the questions. The chosen topic was photosynthesis, which is a type of event. Thus, we created a subset of the knowledge base that had only knowledge about photosynthesis, its sub-events and

concepts that were directly related to such events. For instances, photosynthesis occurs in chloroplasts, so the concept *chloroplast* is included in our reduced knowledge base. A chloroplast is a part of a plant cell, but the concept of plant cell is not in the knowledge base because it is not directly related to photosynthesis or one of its sub-events.

Also, the knowledge base was biologically accurate, so it represented details that are irrelevant to students in our target population. For instance, the process of photosynthesis is somewhat different in plants, algae and photosynthetic prokaryotes. The original knowledge base contained information about all three types of photosynthesis, and used three different class names to distinguish them; all were subclasses of the *photosynthesis* class. Some facts about photosynthesis are general to all three types, so the knowledge base uses *photosynthesis* for them. Other facts are specific to plants, so the knowledge base uses the subclass, *photosynthesis_by_plants*. The facts about algae and photosynthetic prokaryotes were removed from the knowledge base so that the knowledge base remains inside the focus of introductory biology textbooks,

In order to simplify the generation of questions, information about instances that would normally be inferred via inheritance from distal prototypes and classes was copied onto the instances themselves. First, features of the prototype instance of a class were copied to all the instances of the class. Second, the transitive closure of the sub-class relationship was represented. For instance, given

```
sub-class(photosynthesis, chemical process)
sub-class(chemical_process, event)
```

the secondary relations

```
ancestor(event, photosynthesis)
```

was added. Other secondary relations were also added, and will be described later.

2.2.5 Question schemas

Questions were generated from question schemas, where a schema is a template with extra information. A question schema consisted of variables, constraints and question templates. The constraints were matched against the knowledge base in order to bind template variables to concepts in the knowledge base and thus generate a question.

The question schemas depended on the structure of the semantic network we used. As mentioned earlier, the semantic network was extracted from a knowledge base that was used for question answering purpose, where the entire knowledge base was described in terms of class, entity, event and relations among these concepts. Since generating questions about classes (e.g., Is photosynthesis an event?) making no sense in our teaching domain, we focused on generating questions about entities, events and their relations. Briefly put, we generated “What is” questions to have students learn compositional relations between an event and its sub events. We generated “Input and Output” questions to have students learn relationships between events belonging to the same parent event. We generated “Where” questions to have students learn locational relations between events and entities. We generated “Function” questions to have students learn how entities participant in events. This subsection discusses general design issues for the question schemas. Subsequent subsections discuss issues that are specific to the question types just mentioned.

One general issue is that generating all questions of a certain type from the same question schema can be boring to the student. Thus, each question schema had several versions that generated different but synonymous questions, such as “What are the 2 stages of photosynthesis?” and “Could you describe the two sub-processes of photosynthesis?”

Also, as mentioned earlier, only facts about plant photosynthesis remained after removing knowledge about two other kinds of photosynthesis. Thus, “photosynthesis” and “photosynthesis by plants” both appear in questions and they have the same meaning. Thus, the knowledge base itself introduces some more-or-less random variation in the generated output that helps prevent boredom.

A difficult problem in natural language generation is generating a referring expression that is both succinct and unambiguous (Carenini & Moore, 1993). In our system, this problem arises when generating expressions that refer to an instance. When an instance fills a slot in a question schema, the name of an instance (e.g., `inner_membrane001`) cannot be directly used to fill the blank in the question’s template. Although the class of the instance (e.g. `inner_membrane`) can be used with the underscores removed, and the referring expression is succinct, it is sometimes ambiguous (Carenini & Moore, 1993). For instance, the prototype instance `inner_membrane001` represented the inner membrane of a cell whereas `inner_membrane002` represented the inner membrane of a chloroplast. Both of them were instances of the class named `inner_membrane`. If the generator merely used “inner membrane” as the referring expression in the question, then student may not know which inner membrane was referred to.

Our solution to the problem is simple but far from perfect. In the knowledge base, instances that belong to the same class were distinguishable because they participated in different relations. Including all the relations in the referring expressing would certainly distinguish two instances of the same class, but such referring expressions would be too large. Thus, the referring expression generator used only one relation, namely the “part-of” relation. That is, when instance X was a part of instance Y, the question generator used

“the [class of X] of the [class of Y]” to refer to X. If the instance to be referred to was not a part of anything according to the knowledge base, then the question generator used only the name of the class as the referring expression. Moreover, rather than implement this policy as a general subroutine, which inputs an instance and outputs some text, the question templates themselves implemented the policy, and some implemented it slightly differently from others in order to increase the variability of the output. For instance, some templates used “in” instead of “of” (e.g., “the [class of X] in the [class of Y]”). This solution is clearly not perfect. For instance, it sometimes generates spurious contextualization, such as “the calvin cycle of photosynthesis.” It also is strongly affected by the reduction of the knowledge base. For instance, it generates “chloroplast” rather than “the chloroplast of the plant cell” because the reduction eliminated the concept plant cell from the knowledge base. Despite the simplicity of this policy, it does an adequate job on a natural language generation problem that is known to be very difficult to solve perfectly (Rus et al., 2010).

Similarly, generating the determiner for a noun phrase is also a difficult problem (Rus et al., 2010). That is, should the referring expression say “chloroplast”, “the chloroplast”, “a chloroplast” or “any chloroplast”? Our solution is again simple but imperfect. When there is a part-of relationship, then “the” is used twice as the determiner (e.g., “the inner membrane of the chloroplast”). When such a relationship is absent, then the noun phrase is generated without a determiner (e.g., “__chloroplast”). However, the policy is deliberately implemented slightly different in different question templates in order to increase the variability of the language.

Capitalization (e.g., “the Calvin cycle” vs. “the calvin cycle”) is not handled at all. Our assumption is that capitalization would not affect learning. Indeed, many of the minor

disfluencies in the generated questions probably would not affect learning, which is confirmed in the later analysis.

The rest of the section explains how the questions were generated type by type. Appendix B also provides the source in the semantic network for several machine-generated questions.

Generating “what is” questions

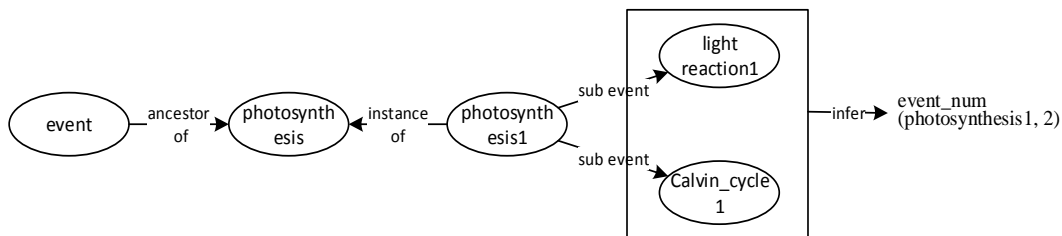


Figure 2 Piece of the semantic network for “what is” questions

“What is” questions ask students to explain a concept directly. A simple example is “What is photosynthesis?” Such questions can be answered in many ways, such as explaining the chemical equation of photosynthesis in the student’s own words and pointing out the different stages of the photosynthesis process. The chemical equation actually describes the input and output of the process, which belong to the behavior part of a system. Pointing out the stages refers to the structure part. We decided to have our “what is” questions focus on the structure instead of behavior, because behavior could be handled by other types of questions. Because photosynthesis is an event, all its parts and subparts are also events. Because the evaluation focuses on photosynthesis, rather than cell anatomy, the part-of relations between entities (e.g., a chloroplast is a part of a plant cell) were removed. Thus, our “what is” questions all ask students to describe the different

stages of a process. Figure 2 shows the corresponding piece of the semantic network for the question “What are the 2 stages of photosynthesis?”

The key to generating this type of questions is identifying the appropriate concept, which is “photosynthesis” in the example. In principle, any class in the knowledge base can become the questioning concept, but many of them are too general or too specific. For example, it doesn’t make sense to ask “What is chemical process?” As it turns out, neither the too-general nor too-specific concepts have sub-events, so generating “what is” questions based only on the part-of relationship rules out those inappropriate concepts.

In order to make it clear that the sub-events are to be described, the question schema used the template, such as “What is [the number of sub-events] stages of [the concept]”. In order to generate such questions, it is necessary to know the number of subevents of each event. This is pre-computed during the preparation phase and stored as a secondary relationship.

Some events have too many (5 or more) sub-events defined in the knowledge base. Pointing out the number of events in this case may confuse the students, because different people may have different criteria for distinguishing a sub-event. In this case, the question schema doesn’t mention the number, and instead uses the following template: “Please explain the process of [concept].”

Generating “where” questions

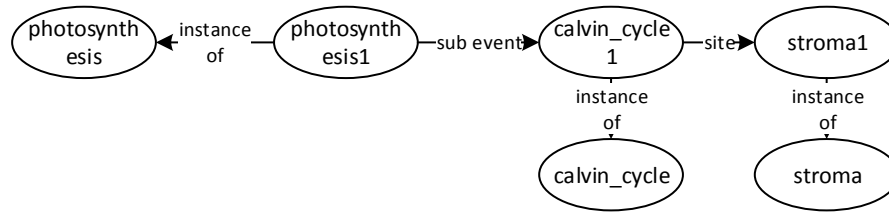


Figure 3 Piece of the semantic network for “where” questions

“Where” questions are classified into two types. The first one is to question the site of an event. For example, “Where does the Calvin cycle occur?” Generating this type of questions is fairly straightforward. Since *site* is a predefined atomic predicate, the question can be composed by generating a question about its object. As mentioned earlier, simple policies were used for generating referring expressions, so spurious context phrases and articles were sometimes generated. In fact, instead of “where does the Calvin cycle occur?” the system actually generated “Where does the calvin cycle of photosynthesis occur?” Figure 3 shows the piece of the semantic work for generating the question.

The second type of “where” question is questioning the origin or the destination of a movement. The knowledge base describes a movement in terms of three atomic predicates. They separately specify the origin, the destination, and the object that is moving. A movement must happen under some circumstance, which means that it must belong to a parent event. So the two corresponding question templates are “In [parent event], where does the [object of the movement] in [destination] come from?” and “In [parent event], where does the [object of the movement] go after it leaves [origin]?”

Generating “input & output” questions

An “input & output” question focuses on the behavior of a process. The most straightforward “input & output” questions ask what a single process needs for starting and what products it produces. These questions can be easily generated from the knowledge base predicates *raw_material* and *result*. Just as their names indicate, the predicate *raw_material* describes the inputs of a chemical process, and the predicate *result* describes its outputs. Again, if the attached process to the predicate is a sub-event of another one, the parent event needs to be included to provide the context. Examples of the templates for the input and output questions are respectively “What does the [process] require?” and “What does the [process] produce?” Figure 4 shows the corresponding piece of the semantic network for generating “What does the light reaction in photosynthesis require?”

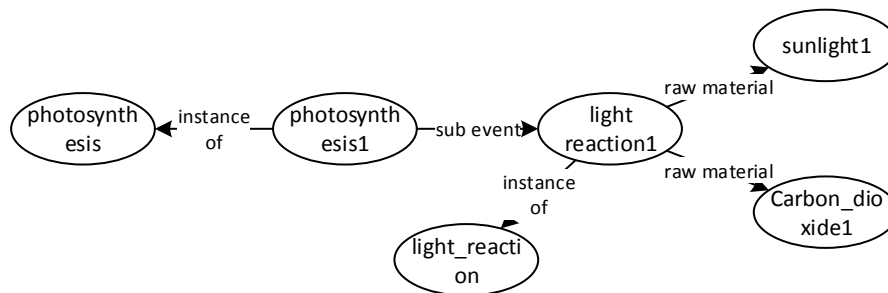


Figure 4 Piece of the semantic network for direct input questions

Because the products of one process often serve as the raw materials of another one, we devised questions to help students review these connections. In generating such a question, the generator finds processes that have the connecting relations and then puts them into the templates. If the two processes are the sub events of the same process, the parent process will also be included in the question to give the context. So one question template is “How does [process1] support [process2] in [parent process]?” This question template involves

somewhat more knowledge than the others. Figure 5 shows the corresponding piece of the semantic network for generating “How does light reaction support Calvin cycle?”

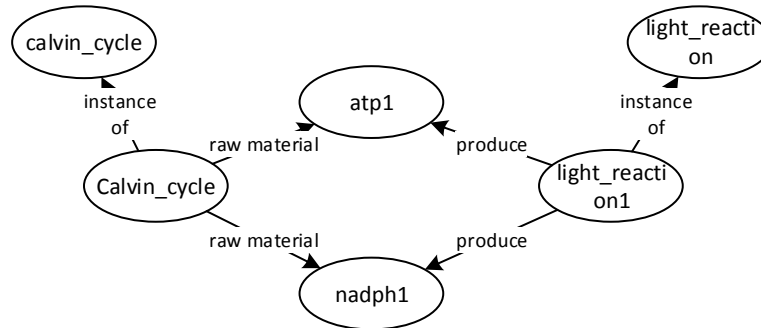


Figure 5 Piece of the semantic network for complicated “input&output” questions

Generating “function” questions

“Function” questions ask about the role of an entity or event in another event. The predicates in the knowledge base corresponding to the role concept are: *enables*, *inhibits*, *causes*, *agent*, *site*, and *raw_material*. Because all these predicates describe how the subject of the predicate (an object) can affect an event, they can all be used to generate a function question. For instance, one question template is: “What is the role of [subject] in [event]?” Figure 6 shows the piece of the semantic network for generating this types of questions.

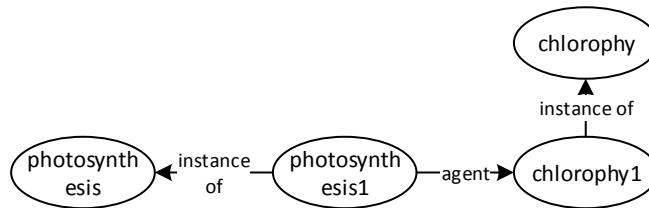


Figure 6 piece of the semantic network for generating “function” questions

Generate the questions in natural language

When the constraints in a question schema are matched against the knowledge base, they bind the variables of the schema. This produces an instance of the schema, such as these:

`What_is(photosynthesis, 2)`. A “what is” question, with two sub-events.

`Site_of(calvin_cycle)`. A “where” question.

`Raw_material(photosynthesis)`. An “input” question.

`Role_of(sunlight, light_reaction, photosynthesis)`. A “function” question.

Now that the “question” is written as a relation, we can have several different natural language templates for each type of relation, so that students wouldn’t get bored when they were answering the same type of questions multiple times. In the process of transforming the relations into the natural language questions, the program randomly selects one natural language template from several predefined ones.

Because the knowledge base was limited to photosynthesis, the generator produced 56 questions: 6 were “what is” questions; 17 were “where” questions; 17 were “input & output” questions; and 16 were “function” questions.

2.3 Collecting questions by Google

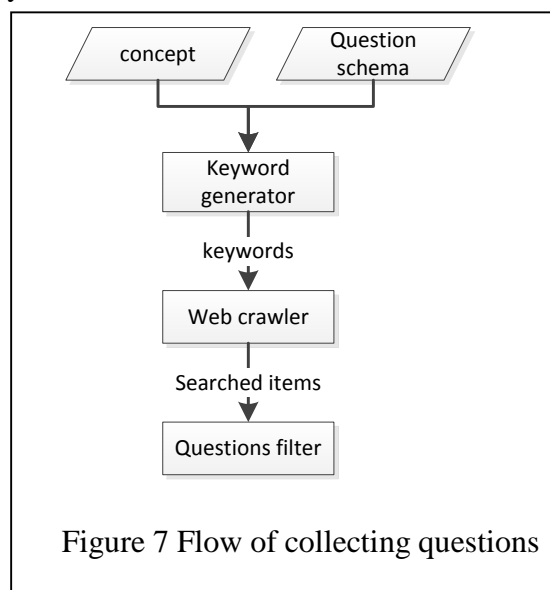
For a topic like biology, which is widely taught, standard question-answering websites, such as ask.com and yahoo answer, can be used to find human-generated questions. We used a Google custom search engine to search the entire Internet with these question-answering websites prioritized. Figure 2 described how the questions were collected. In order to generate keywords, an algorithm described in section 3.1 was repeatedly given a concept and one question schema as the inputs. The keywords were then sent to the Google engine, which returned a set of web links. The generator visited those web pages and

crawled the relevant text. Questions were selected from the text by the filter described in section 3.2. In the following, the two main steps were described in detail: generating keywords and filtering out irrelevant items.

2.3.1 Generating the keywords

Since Google's search is driven by a list of keywords, it is crucial to choose the keywords properly. To make the collected questions comparable with the machine-generated questions, the keywords need to specify both the concept to be questioned and the type of question. Let us first discuss how keywords for concepts were generated.

An initial list of concepts was extracted from the chapter on photosynthesis in a biology textbook and an article on photosynthesis in Wikipedia. However, the same concept may have several different names that are used by different people and websites. For example, a cell membrane is also called a plasma membrane. Since a keyword search doesn't know about synonymous terms, the problem needs to be resolved when keywords are generated. To cover all the variations, we used WordNet (Fellbaum, 1998) to augment the initial list of concepts with synonyms.



For generating keywords corresponding to the type of question, we used the keywords shown in Table 1. Any combination of the word in question type table and the word in the list of concepts makes one set of keywords. For example, while collecting “output” questions for the concept “photosynthesis”, a possible keywords set is {produce, photosynthesis}. However, a short set of keywords like this often make Google search return too many non-question sentences. In order to increase the portions of questions retrieved and release part of burden of filtering process, we added “what”, “how”, “where” and “when” to the keywords set respectively to form the entire set of keywords. Thus, there would be four queries that were actually issued for the keywords set {produce, photosynthesis}. They were {what, produce, photosynthesis}, {how, produce, photosynthesis}, {where, produce, photosynthesis}, and {when, produce, photosynthesis}

Table 1 . keywords for Google search

Type	Keywords
what is	what, explain
where	where, place, site
Input & output	produce, product, result, material, cause, need
function	function, use, usage, inhibit, enable

2.3.2 Filtering mechanism

The module web-crawler takes the keywords set by set and issues queries via the Google API. The returned search results are in JSON format. Each item contains three properties of the webpage. They are “Title”, “Snippet” and “Address”. Both the Title and Snippet can contain a question or parts of a question, so the contents of these two properties are stored as candidates of the final returned questions. The program then goes to the actual

web page to get the content in HTML format by the link provided in “Address”. The questions may be deeply embedded in the content so we wrote a parser in terms of HTML tags to extract the possible text and store them as the list of candidates. Specifically, the parser considered the content of every HTML tag as one candidate and left the final decisions to the filters.

Every set of keywords generates a list of candidate questions, and all these candidates are concatenated. The next step is to go through each candidate and check whether it is qualified to be returned. The criteria of the filter are listed below:

- length: The number of the words in the candidate must be between 3 and 50. A one or two word candidate usually is a fragment of question. A long one is often a piece of JavaScript code.
- relevance: The candidate must contain one of the concept keywords and one of the question type keywords.
- duplication: The candidate should not be too similar to a candidate that is already on the list of qualified questions.

Duplication is defined in terms of the edit distance between the candidate and the qualified question. Different from the usual definition of edit distance, words were used as the atomic units instead of characters. Because absolute value of edit distance is unfair to the items with many words, we divided the absolute value by the length of the current candidate. If the quotient was below 0.2 for all existing qualified questions and the candidate passed the other two measures as well, it would be added to the list of qualified questions.

To improve quality, the questions were further filtered with the Stanford parser. The Stanford parser tagged each word in a sentence with a syntactic symbol. When the first word's tag was "SBARQ", which is a sign of direct question introduced by wh-word or wh-phase, this candidate was judged to be a question. In total, there were 8 concepts input to the generator, and the generator finally returned 43 questions.

2.4 Coverage of the machine generated questions

Unlike many other applications of question generation, it is important for instructional applications that the set of questions that are generated "cover" the target material. In particular, the machine-generated questions were all based on a small number of predicates such as "site" or "cause." Thus, they could be considered "factual" questions rather than "deep reasoning" questions. Although it would be difficult to precisely characterize the difference between such questions, we can evaluate coverage by counting how many questions are in common between human questions and the machine generated questions. We assume that if there exist "deep questions" on a topic, then some human will probably ask them. In other words, we need to examine the coverage of the machine-generated questions against human-generated questions.

Since we were working on a narrow domain, "photosynthesis" in entry level biology, we assumed that the union of the questions collected from textbooks and the questions gathered from the web would serve as an adequately large set of human questions. Given this assumption, we calculated the coverage of the machine-generated questions by matching them against human questions. Some different human questions were actually asking the same thing with different expressions. These questions were combined together. On other hand, 4 special "what is" questions such as "What is light reaction process?" were

split into several questions, because each of them could be treated from different angles. These “angles” correspond to annotations, which will be described in a moment.

After splitting and merging, there were 59 distinct human questions. Out of the 59 human questions, 24 (41%) were covered by machine-generated questions, and 35 were not. Because one human question sometimes covered more than one machine-generated question, 28 machine-generated questions were covered by the 24 human made questions. Therefore, of the 56 machine-generated questions, 28 (50%) matched human questions and 28 did not. The latter figure is important, because it suggests that humans may be failing to generate enough questions to completely cover the knowledge in the knowledge base.

In order to understand the differences between the two sets of questions, we first annotated all the questions according to their topic. We classified topics as structure, behavior or function, as this three-way distinction is common (Goel, Rugaber, & Vattam, 2009) and can be viewed as a related to the depth of the question.

Questions asking compositional relations of events, which were sometimes considered as behavior questions, were considered as structure questions here. For example, the question “What are the stages into which photosynthesis is divided?” was treated as a structure question.

Questions annotated with “behavior” were further classified into two levels. Questions asking immediate facts were annotated as “behavior-1”, whereas questions whose answers involved inferences of facts were annotated as “behavior-2”. For example, “What does cyclic photophosphorylation produce?” was annotated as “behavior-1”, and “How do the raw materials of photosynthesis reach the chloroplasts of the leaves?” was annotated as “behavior-2”.

Questions like “What is the function of chloroplast membranes?” were annotated as function questions.

As mentioned before, three “what is” questions were split into ten new questions. Two “what is” questions asked about a process, so they were each split into four questions with the four annotations (structure, behavior-1, behavior-2 and function). The remaining “what is” question asked about an entity. The “behavior” annotation was not suitable for that question, so that question was split into only two ones.

Table 2 describes the results for the four categories of question topics. The two columns on the right show how many questions were matched by questions from the other category. The number of matched questions differs slightly in the two columns because in a few cases, one human-generated question matched more than one machine-generated question.

For the structure and behavior-1 categories, there was considerable overlap of the machine- and human-generated questions. These are fairly simple questions, so it is not surprising that the two sources tended to generate the same questions. For the behavior-2 and function categories, which tend to include more complex questions, the overlap was much smaller. In terms of a distribution of question types, the machine generated many fewer behavior-2 questions and many more behavior-1 questions than the humans.

Krathwohl (2002) divided the cognitive process of a student into six categories from simple to complex: *remember*, *understand*, *apply*, *analyze*, *evaluate* and *create*. Mastery of a simpler category would be prerequisite to mastery the next more complex one. Therefore, it is reasonable to classify the questions that only test simpler cognitive categories as shallow questions and classify the questions that test more complex cognitive categories as deep questions. Because remembering and understanding all the basic facts

in photosynthesis would be enough for answering structure and behavior-1 questions, these two types of questions were classified as shallow questions. Answering behavior-2 and function questions would require more a complex cognitive process, so these two types of questions were classified as deep questions. Because a simpler cognitive process is prerequisite to a more complex one, answering shallow questions may also help students have deep learning, which was confirmed by I-H Hsiao, Sosnovsky, & Brusilovsky (2010)

Table 2 Overlap of machine and human-generated questions

Category	Machine	Human	Human matched	Machine matched
Structure	9	16	9/16 (56%)	9/9 (100%)
Behavior-1	30	11	10/11 (91%)	13/30 (43%)
Behavior-2	2	14	1/14 (7%)	1/2 (50%)
Function	15	18	4/18 (22%)	5/15 (33%)
Total	56	59	24/59 (41%)	28/56 (50%)

This analysis seems to be the first one to study the coverage issue of question generation, so there is no previous work or baseline for comparison. Rus et al. (Rus, Cai, & Graesser, 2007) examined the quality of fact questions generated from plain text by the schemas. Their fact questions were similar to “structure” and “behavior-1” questions in our study. All their generated fact questions were annotated by two human judges, and a question was classified as “good” when both of the two annotators agreed on that. They found that only 55% of the machine-generated questions were rated as “good”. Although they were

looking at a different measure from us, their work suggests the difficulty of question generation.

2.5 Main study

The research question we wanted to answer was “What are the relative qualities of questions generated from a knowledge base, from searching the web and from a professional source of questions?” For the evaluation, we randomly selected 40 questions generated from the knowledge base, 20 questions from the web search and 20 from the list of professional questions. These selections formed a photosynthesis question base with 80 questions in total.

The quality of questions was evaluated in terms of four measures: fluency, ambiguity, pedagogy and depth. Every measure had a scale from 1 to 5. *Fluency* had the judges rate the grammatical correctness. *Ambiguity* asked them whether the question was semantic ambiguous or not. After the judge answered the question, he/she was asked whether answering the question helped him/her understand or review any concepts, which was our measure of *pedagogy*. For *depth*, we asked the judges to rate how much thinking was involved in their answering of the question. The first two measures, fluency and ambiguity, were borrowed from a question generation challenge (Rus et al., 2010).

We did not expect significant differences among the three types of questions in the first two measures, but we expected the professional questions to exceed the other two in *pedagogy* and *depth*. In addition to the quality measures, we also asked the judges to rate the relevance of the question to their level of understanding of biology. So there were 5 measures for each question in total.

2.5.1 Procedure

Biology Questions

Pick a question

Question: What are the roles of NADPH and ATP in the chemical stage of photosynthesis?

Are all the concepts in the question covered in your Biology classes?
all of them not at all
 5 4 3 2 1

Is the question grammatically natural?
very natural not at all
 5 4 3 2 1

Is the question ambiguous?
very ambiguous not at all
 5 4 3 2 1

Do you think the question was made by a human or a machine?
 Human Machine

Your answer to the question:

ATP provides the energy to the process of photosynthesis.

Did you learn or strengthen your understanding of any concept by answering the question?
Yes. not at all
 5 4 3 2 1

How much thinking vs. just memorizing did you use in answering the question?
Thinking. Just memorizing
 5 4 3 2 1

Submit

Figure 8 Screenshot of the experiment

All 80 questions were stored in a database, and we implemented a data collection website so that students could answer questions from anywhere and need not come to our lab. The webpage is shown in Figure 8. To serve as judges, we recruited 12 college students from our school's Paid Research Participation System, where students registered for

participating in research experiments. All but one participant reported that they were native speakers of English; the remaining participant was from India and had been speaking English since she began school. We selected students who reported that they took an intro biology course in the recent two semesters. Every participant was required to answer all 80 questions in order to get 20 dollars as the compensation for finishing the experiment. The experiment was IRB approved.

2.5.2 Results

Although the questions asked to a student should have been distinct, a technology issue caused some of the early participants to get duplicates of a few questions instead. There were 12 judges, and 40 machine-generated questions, 20 web-generated questions and 20 professionally written questions. Thus, there should have been $12 \times (40 + 20 + 20) = 960$ data points per measure (e.g., fluency). When the judgments of duplicates were removed, there were 867 data points for each measure.

A typical method for evaluating machine-generated questions is to have two or more people rate every question independently, and the mean of their ratings is treated as the final score of the question. On questions where the scores from the raters were substantially different, the raters discuss and resolve their conflicts. In our experiment, the 12 participants were the raters, but it was impossible for them to discuss and resolve discrepant judgments, so the scores were combined without discussion and adjustment.

We used two different methods for combining scores across judges. The first one assumes judges may be biased, so it normalizes their scores and makes every judge's mean score for a particular measure be zero with a standard deviation of one. This method assumes that the scores given by the judges form a ratio scale, and thus it is sensible to take

means and standard deviations. The second method assumes that the scales are ordinal instead of ratio, so it is not sensible to take the mean and standard deviation. This method uses medians to compare scores.

Normalized comparison using the ratio scale assumption

Although the scale for every judge was 1 to 5, different judges chose to use different part of the scale. For example, some judges used mostly 2, 3 and 4 while other judges used the whole scale. Simply taking the mean would give greater weight to the judgments of participants who used a wider range of scores. Thus, as mentioned earlier, we normalized the judges' scores using a z-transformation:

$$Z = \frac{(\text{score} - \text{mean})}{\text{std.}} + 3$$

where the mean and standard deviation are taken over all 80 scores from a particular student's judgments about a particular measure (e.g., fluency). By subtracting the mean and adding 3, the z-score of every judge/measure combination has a mean of 3, which is the center of the 1-5 scale.

To aggregate across the 12 judges, we simply took the means of their z-scores, as shown in Table 3, and the original scores were as shown in Table 4. Reading across the rows suggests that there were few differences among the three different question types. To test that, we ran ANOVA across all the three groups and pair-wise T-tests on each pair of two groups. The results are shown in Table 5 and Table 6.

Table 3 mean (and standard deviation) of the measures-original scores

	knowledge-base (n=40)	web (n=20)	professional (n=20)
fluency	4.05(0.340)	4.06(0.479)	4.00(0.341)
ambiguity	3.97(0.331)	4.05(0.367)	4.00(0.322)
pedagogy	2.72(0.342)	2.89(0.254)	2.94(0.363)
depth	2.59(0.351)	2.66(0.360)	2.93(0.478)
relevance	3.51(0.460)	3.92 (0.298)	3.65(0.409)

Table 4 mean (and standard deviation) of the measures-normalized scores

	knowledge-base (n=40)	web (n=20)	professional (n=20)
fluency	3.03(0.343)	2.98(0.517)	2.97(0.323)
ambiguity	2.95(0.358)	3.05(0.402)	3.03(0.291)
pedagogy	2.90(0.360)	3.06(0.281)	3.14(0.337)
depth	2.93(0.319)	2.98(0.277)	3.16(0.405)
relevance	2.86(0.481)	3.26(0.334)	3.02(0.471)

The T-tests did find some significant differences. The web-generated questions were more relevant to students' knowledge than the other two types of questions. This difference was not surprising because the web-mining method tended to collect questions from students.

Professionally written questions beat the questions generated from the knowledge base in both pedagogy and depth. The questions mined from the web received marginally higher pedagogy ratings than the questions generated from the knowledge base. This is consistent with the analysis of coverage presented earlier which showed that the more questions from humans than machines were classified as Behavior-2 or Function, which seem intuitively to be deeper and potentially more pedagogically useful types of questions.

Table 5 ANOVA of means from Table 4

Measure	Value
Relevance	p<0.01, F=5.38, power=0.77
Fluency	p=0.84, F=0.17, power=0.94
Ambiguity	p=0.46, F=0.79, power=0.68
Pedagogy	p=0.03, F=3.71, power=0.68
Depth	p=0.04, F=3.32, power=0.65

Table 6 Pair by pair T-tests of means from Table 3 and Table 4

Professional vs. Knowledge-base				
relevance	fluency	ambiguity	pedagogy	depth
p=0.214	p=0.546	p=0.363	p=0.0167**	p=0.0313**
d=0.344	d=-0.163	d=0.235	d=0.669	d=0.669
power=0.507	power=0.612	power=0.517	power=0.495	power=0.594
Knowledge-base vs. web				
relevance	fluency	ambiguity	pedagogy	depth
p<0.01**	p=0.7153	p=0.3481	p=0.0659*	p=0.5095
d=-0.9118	d=0.1153	d=-0.2709	d=-0.4748	d=-0.1735
power=0.360	power=0.738	power=0.546	power=0.448	power=0.587
Professional vs. web				
relevance	Fluency	ambiguity	pedagogy	depth
p=0.0765*	p=0.9538	p=0.8453	p=0.4419	p=0.1088
d=-0.5472	d=-0.0199	d=-0.0658	d=0.2383	d=0.4909
power=0.579	power=0.954	power=0.850	power=0.590	power=0.567
** means p<.05 * means p<.10				

However, even after normalization, the scores from different raters on the same question and measure were often far apart. This suggests that human judges may be unreliable for those particular measures on those particular questions. Because it was impossible to call

back the participants so that they could discuss and agree upon a rating, a second analysis was done discarding the questions/measure combinations with high standard deviation.

For the measure Pedagogy, the results are shown in Table 7. Out of the 80 questions, 4 questions were removed because their standard deviations were much bigger than others (more precisely: They were greater than 2 standard deviations larger than the mean of the 80 standard deviations). Among the 4 questions, there were 2 from the knowledge base, 1 from the web and 1 from professionals. Table 6 reports the mean values and the significances of the comparisons of the remaining 76 questions. There is still a trend for the professionally written questions to be pedagogically better than the questions generated from the knowledge base, but the trend is only marginally reliable perhaps due to the loss of power due to the loss of data points.

For the measure Depth, the results are shown in Table 8. Out of the 80 questions, 2 questions were removed, 1 from the web and 1 from professionals. Removing the two questions didn't lead to a different result.

Although removing question/measure pairs from the sample is one way to cope with unreliability, it reduces power, which makes it harder to tell when two methods of generating questions are truly equal on a measure. Thus, for our second analysis, we used medians instead of means, on the grounds that they are less sensitive to outliers.

Table 7 The comparison of pedagogy after removing questions with high inter-rater variability.

	mean (std.)	vs. Professional	vs. web
knowledge base	2.96 (0.3128)	p=0.0913* d=-0.5005 power=0.525	p=0.1652 d=-0.3859 power=0.490
web	3.07 (0.2822)	p=0.6696 d=-0.1396 power=0.697	
Professional	3.12 (0.3330)		
* means p<.10			

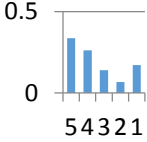
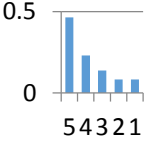
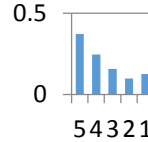
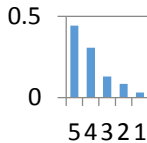
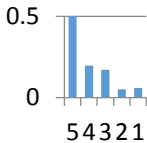
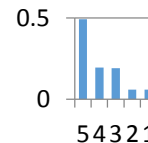
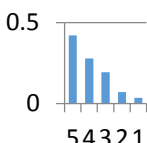
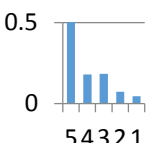
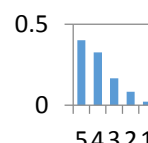
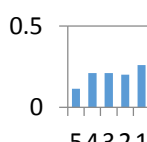
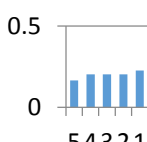
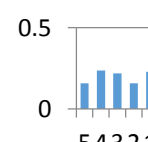
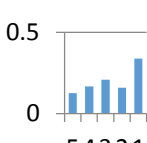
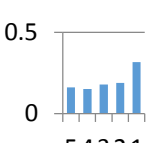
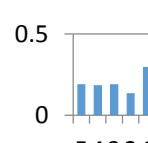
Table 8 The comparison of depth after removing questions with high inter-rater variability

	mean (std.)	vs. Professional	vs. web
knowledge base	2.59 (0.351)	p=0.0429** d=-0.577 power=0.530	p=0.653 d=-0.126 power=0.685
web	2.64 (0.354)	p=0.102 d=0.544 power=0.505	
Professional	2.83 (0.491)		
** means p<0.05			

Compare by responses

Among the 867 responses, there were 435 for machine-generated questions, 217 for Google collected questions and 215 for textbook questions. Table 9 shows the distribution and median for each measurement.

Table 9 Histograms and median value of the measures

measure		knowledge base (n=435)	web (n=217)	professional (n=215)
relevance	Median	4	4	4
	Relative frequency			
fluency	Median	4	5	4
	Relative frequency			
ambiguity	Median	4	5	4
	Relative frequency			
pedagogy	Median	3	3	3
	Relative frequency			
depth	Median	3	2	3
	Relative frequency			

For most measures, the medians were the same. The exception was that the medians of fluency and ambiguity suggest that web-generated questions were better than the other two. However, the underlying distributions for all three question sources on the measures relevance, fluency and ambiguity show that all were similar and all were probably not a major source of pedagogical difficulty.

Pedagogy and depth are the two measures we most care about. The histograms look similar across question sources.

To test whether there are any significant differences existed, we ran Quantile regression (Koenker, 2005), which is an algorithm to predict the median from the training data. A variable I was introduced to represent the identity of the question source, and the significance level of the coefficient of the variable I represents the reliability of the difference. For example, in comparing the pedagogy of knowledge base questions to that of web questions, I equals 1 when the question is from the web, and I equals 0 when the question is from the knowledge base. The significance of the coefficient of I is the reliability of their difference between the two question sources.

Table 10 Quantile regression comparisons.

knowledge base vs. professional				
relevance	Fluency	Ambiguity	pedagogy	Depth
p=1.0	p=1.0	p=1.0	p=1.0	p=1.0
web vs. knowledge base				
relevance	Fluency	ambiguity	pedagogy	Depth
p=1.0	p=0.043**	p=0.042**	p=1.0	p=0.016**
web vs. professional				
Relevance	Fluency	Ambiguity	pedagogy	Depth
p=1.0	p=0.012**	p=0.047**	p=1.0	p=0.064*
** means $p < 0.05$. * means $p < .10$				

The Quantile regression comparisons suggest that the differences in the medians that are apparent in Table 10 can all be trusted. That is, there is no difference between the knowledge-base questions and the professional questions on any measure, but there are three differences between the web-mined questions and the other two: The web mined questions are more fluent, less ambiguous and shallower than the others.

The pattern of median results is not consistent with the pattern of results from the means of the z-scores. The latter showed that the knowledge base questions were shallower and less pedagogically beneficial than the professional questions, and that the web-based questions were in between the other two types on all measures except relevance, where they were ranked higher than both the professional and knowledge-base questions.

These conflicting results leave us with an interpretation challenge. The main issue is how to interpret the conflicting results for pedagogy and depth. The histograms of Figure 7 suggest that the judgments were nearly uniformly distributed, which in turn suggests that medians may not be sufficiently sensitive to detect differences. Thus, we tend to put more confidence in the results about pedagogy and depth that are shown in Tables 7 and Table 8, which factor out the influence of questions with high inter-rater variability. We can probably ignore the conflict about fluency, ambiguity and relevance because those are not the main measures of interest. Moreover, we had informal interviews with some of the participants after the study, and no one claimed that they had trouble in understanding the questions. So fluency and ambiguity probably would have little effects on students' learning. The next analysis addresses this question.

Correlation of the measures

If the goal of generating question is to enhance students' learning, are relevance, fluency, ambiguity or depth of the question going to affect the pedagogy? To answer this and similar questions, we calculated the correlations of the measures. The original 867 responses per measure were used without aggregating or normalizing or dividing by question source.

The result was showed on Table 11. From the table, the depth of the question is highly correlated with the pedagogy. All the other three measures have very low correlations with pedagogy. Relevance, fluency and ambiguity are moderately inter-correlated, and apparently uncorrelated with depth and pedagogy.

These results are consistent with those of Table 10. The web-mined questions received high scores in fluency and ambiguity, but are tied with or even worse than questions from the knowledge base and professionals in pedagogy and depth. This suggests that as long as the question itself is understandable (staying at the relatively high level in fluency and ambiguous), its pedagogical features are unrelated to the syntactic aspects.

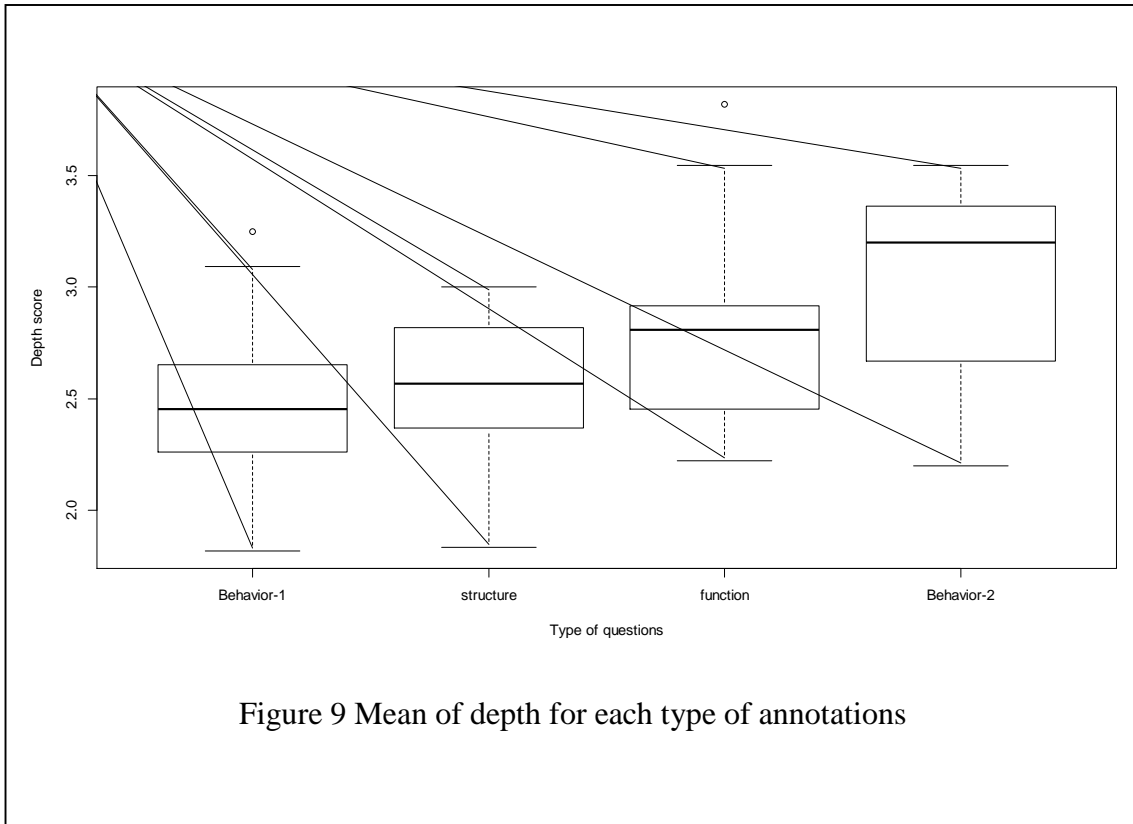
Table 11 Pearson's correlation of the different features of the questions.

	Relevance	fluency	ambiguity	pedagogy	depth
relevance		0.398	0.331	0.168	0.140
fluency			0.328	0.202	0.124
ambiguity				0.061	-0.013
pedagogy					0.800
depth					

When we measured the overlap of the human-generated questions and the machine-generated questions, we used four annotations: structure, behavior-1, behavior-2 and

function. It seems intuitively clear that behavior-2 and function questions are deeper than the other two types. To explore whether students agreed with this intuition, each of the 80 questions used in the experiment were annotated with “behavior-1”, “behavior-2”, “function” or “structure” as described in section 4, except for 2 “what is” questions that could not be annotated with a single flag. So 78 questions were split into 4 groups based on the types of annotations. As Figure 9 shows, questions with different annotations have clearly different depth score means. An ANOVA was conducted to confirm our assumption ($p < 0.001$, $F = 6.414$, $df = 74$). Tukey HSD test was performed to test the group-by-group differences. If behavior-2 and function questions are considered as deep questions, then both should get higher scores than behavior-1 and structure questions. Both of behavior-2 and function questions do have significantly higher scores than behavior-1 questions. But only behavior-2 questions have a significant higher depth score than structure questions. It is probably because some function questions have very obvious answers. E.g. “What do chloroplasts enable plant cells to do?” can be simply answered as “photosynthesis”, whereas behavior-2 questions were defined such that their answers always required inference.

The analysis was repeated with just two categories, shallow and deep. That is, behavior-2 and function questions were aggregated together as the deep group, whose mean depth score was 2.85 (SD=0.41, n=35), and behavior-1 and structure questions were grouped together as the shallow group, whose mean of the depth score is 2.51 (SD=0.32, n=43). The difference is significant according to t-Test ($t = 3.948$, $p < 0.001$), with a large effect size (Cohen’s $d = 0.92$). This suggests again that our coding of questions corresponded to the depth judgments of the 12 raters.



2.6 Discussion

In order to compare questions generated from the knowledge base, mined from web, and collected from textbooks, we first examined the coverage of the human-generated questions (the latter two categories) by the questions generated from the knowledge base. As expected, the machine-generated questions had a much higher coverage of shallow human questions than deep human questions. This is not a surprise because each question generation schema used with the knowledge base only accessed a small number of knowledge base relationship. We tried generating complex question schema using seed questions to infer the pattern of knowledge base relationships for the schema. This turned out to be too sensitive to the details of the knowledge base. Some seed questions generated

far too many question schemas for a human author to check, and other seed questions generated too few candidates. Therefore, when questions are generated from a knowledge base, it seems that a practical method is to have human write the constraints for generation.

In terms of the quality of the machine-generated question, our experiment shows that the questions from the three different sources have only minor differences in their quality ratings. Although some significant differences in their fluency and ambiguity ratings were observed, the differences in pedagogical and depth ratings were unreliable, appearing only with some statistical tests and not others. In particular, we found:

- Knowledge base vs. professional:
 - Z-scores: for **pedagogy** and **depth**, knowledge base < professional
 - Z-scores with removal of questions with high inter-rater variability: no reliable difference on **pedagogy**, reliable difference on **depth**
 - Medians: no reliable differences
- Knowledge base vs. web mining:
 - Z-scores: for relevance, knowledge base < web
 - Z-scores with removal of questions with high inter-rater variability: no reliable differences on **pedagogy and depth**
 - Medians: for fluency and ambiguity, knowledge base < web; for **depth**, knowledge base > web
- Web mining vs. professional:
 - Z-scores: no reliable differences
 - Z-scores with removal of questions with high inter-rater variability: no reliable differences on **pedagogy and depth**

- Medians: for fluency and ambiguity, web > professional

If these ratings from 12 biology students can be trusted as proxies for the pedagogical quality of the questions, it appears that all three sources are equally beneficial on the whole.

However, a different analysis did show a difference in the depth. Unlike the main evaluation, which was conducted on a sample of each type of questions, this analysis compared all the machine-generated questions to all the human generated questions, both professional and web-mined, merged together. When we classified questions as shallow (both structural and behavior-1) versus deep (both behavior-2 and function), we found strong correlations with the judgments of the 12 biology students, but the ratings systems were not identical. Using our classifications, we found that about 50% of the human questions were deep whereas only about 30% of the machine questions were deep. Moreover, only 15% of the human deep questions were also generated by the machine, while 70% of the human shallow questions were generated by the machine. These figures are not terribly surprising, as they are consistent with the design goals for the machine generated questions. However, we were pleasantly surprised to see that the machine generated a very high percentage of the shallow questions that humans generated.

2.7 Conclusion

Given a knowledge base, we developed a rule-based algorithm to generate biology questions, and developed a method to collect biology questions by searching the web with keywords. When 12 biology students judged the pedagogical merit, depth, relevance, fluency and ambiguity of the two different types of questions to professional human made questions, the pattern of results depended on whether we assumed the judgments were a ratio scale or an ordinal scale. If we assumed a ratio scale, then professionally generated

questions were rated as deeper and more pedagogically beneficial than questions generated from the knowledge base, but there were no differences in fluency and ambiguity. On the other hand, if we assumed an ordinal scale, then there were differences in fluency and ambiguity, but no differences in depth or pedagogy.

However, when we classified the questions by their topic, there were differences due to the design of our question schemas, in that about 50% of the human-generated questions addressed deep topics (function and behavior-2) whereas only about 30% of the machine-generated questions did. However, the machine generated questions did cover a large fraction of the shallow human-generated questions. Conversely, only 50% of the machine-generated questions were covered by human-generated questions.

This suggests that machine-generated questions can play an important role in instruction in that they can generate the shallow questions that human authors would otherwise need to generate. This would leave human authors to focus on generating deep questions. Moreover, if our method for collecting human-generated questions is a fair representation of questions that humans generate, then it appears that humans are failing to generate about half the shallow questions that students need. Thus, this suggests that machines may be better than humans in that they can generate a more complete set of shallow questions.

However, the results also sound a cautionary note, in that the biology students did not perceive much difference in the depth or in the pedagogical benefits of the different types of questions. It is not clear whether our classifications or their judgments are better predictors of the actual pedagogical value of the questions. Thus, the next step in this line of research is to compare learning gains from the three types of questions.

Another caution is concerns the generality of the question generation schemas, which must match the structure of the knowledge base. If a new knowledge base were used with a different structure, new questions schemas would be needed.

Moreover, the knowledge base can be considered to be a model of an ideal student's knowledge (Carbonell, 1970), so depending on the learning objectives defined by different instructors, the knowledge base may need to be adjusted correspondingly. For example, biology teachers in high school don't expect their students know as much about photosynthesis as college students who major in biology. While *removing* knowledge to change a college knowledge base into a high school one would not require changing the question schemas, *adding* more knowledge may bring in new predicates and thus necessitate new question schemas.

CHAPTER 3
ADAPTIVELY SELECTING BIOLOGY QUESTIONS GENERATED FROM A
SEMANTIC NETWORK

3.1 Introduction

Practicing is necessary for learning and especially for retention. Thus, a typical textbook chapter has exercises at its end that allow students to practice using the knowledge conveyed in the chapter. A typical math, physics or engineering textbook may have a hundred or more problems at the end of each chapter. If the textbook has an online homework website, then it typically can generate an infinite number of problems by plugging into randomly generated numbers into templates.

However, other subjects are not as fortunate as the mathematically intensive ones. For instance, a typical college biology textbook has only a handful of questions at the end of each chapter. Online resources provide a few more questions for students to answer, but the questions tend to duplicate those in the chapters and there are not enough of them. Thus, when students are trying to practice using knowledge for a non-mathematical subject, they rapidly run out of questions to answer and must resort to re-reading the chapter.

Our overarching goal is to develop technology for web-based practice systems in biology and other non-mathematical subjects. Such systems should have at least five properties: (1) They should be able to generate an essentially infinite number of practice questions so that students will have ample opportunity to practice using the knowledge they have acquired while listening to lectures or reading the textbook. (2) The systems should also be adaptive, in that they select or recommend questions that will maximize the student's learning and engagement. (3) The systems should provide helpful corrective feedback

when students answer incorrectly. (4) The systems should ask both open response questions, where students type in an answer, and multiple choice questions, where the answer is entered more rapidly, with a single click. (5) The multiple choice questions should offer plausible incorrect answers, so that students with inchoate misconceptions may be enticed to articulate them and get corrective feedback on them.

In earlier work (Zhang & VanLehn, submitted), we compared two techniques for generating biology questions automatically. One was based on mining the web. The other was based on reasoning from a biology knowledge base that others had generated for their own purposes (answering questions). Human judges compared questions from textbooks to both the web-mined questions and the inferred questions. The judges rated the questions for depth, pedagogical effectiveness, fluency and ambiguity. Analyses of the ratings found that inferred questions were just as good as the textbook questions and web-mined questions, although they tended to be shallower. An attempt to generate deep questions from the knowledge base foundered, and that issue remains a topic for future work. However, shallow questions are quite important in biology and similar domains, since it is likely that students will be unable to learn much from attempting to answer deep questions until they have first mastered the knowledge required by shallow questions.

This chapter reports our progress on the second required feature of a biology practice website: Adaptive selection of questions. Practice can be a boring waste of time when the questions exercise only knowledge that the student has mastered. On the other hand, practice can be a frustrating waste of time when the student is not yet prepared to learn from the question. Because different students have different prior knowledge and learning rates, a practice system should provide personalized selection of practice questions.

However, in order to select a question that matches the student's current state of knowledge, the system must formally represent both the student's current state of knowledge and the knowledge exercised by the question. That is, the system needs two models. It needs a *student model* that is updated by the system as the student works, and each task needs a *question model* that indicates what knowledge is required by that question.

Some mathematical practice systems do adaptive question selection (see review below). They typically require that the author of a question or question template to specify the knowledge required for answering it. Even though the topic is math, specifying the question models manually can be tedious and a source of errors. Given the less formal nature of biology and similar subjects, having human authors specify the question models is likely to be unacceptably inaccurate or a huge time sink.

Our contribution is showing that when the questions are generated from a knowledge base, the knowledge base can also be used to generate the question models and the student models. This is not technically challenging, but it is risky in the sense that the resulting models might not align well with student's actual knowledge structures, which could cause adaptive question selection to behave poorly. Thus, we tested the resulting adaptive question selection system to see if the adaptation had the desired impact on learning gains. We compared the learning of students who answered questions selected to *increase* their learning to students given questions selected to *decrease* learning. As expected, the manipulation did influence learning gains, and in the expected direction. This supports the viability of using a knowledge base to generate question models and student models.

The next section reviews relevant prior work. The subsequent section discusses the technology, focusing on the simple technique used to understand the students' typed responses to questions and on the Bayesian methods used for updating the student model and selecting questions. The last half of the chapter presents the experiment used to test the viability of the approach.

3.2 Prior works in adaptive task selection

Many practice systems on the market and in the literature are adaptive in that they make decisions based on the student's performance so far. This review focuses on systems that select tasks for the user (in biology, a question is a task), but some systems make other types of decisions, such as: Which hint to start with in a sequence of increasingly strong hints (Wood, 2001)? Whether to give an unsolicited hint or to respond with a hint when asked for one (J. Beck, Woolf, & Beal, 2000; Murray & VanLehn, 2006; Murray, VanLehn, & Mostow, 2004)? Which behavior an affective agent should perform (Arroyo, Mehranian, & Woolf, 2010; Muñoz, Mc Kevitt, Lunney, Noguez, & Neri, 2010)? Should the task be posed as an example, a problem with feedback or a problem without feedback (Rafferty, Brunskill, Griffiths, & Shafto, 2011; Whitehill, 2012)?

3.2.1 The student's state

The information that the system has available on a student consists of the whole history of interaction with the student up to this point. The history can be processed by collaborative filtering and related algorithms to select a next task (Brusilovsky, 2007). The basic idea is that if my history matches the history of several other students, and

some of those students turned out to be more successful than others, then I should be given the task that was given to the majority of the successful students at this point.

Alternatively, the history of interaction can be summarized as the student's "state." Thus, the decision-making algorithm only needs to examine the student's current state and not the whole student history. This is the approach followed here and in most tutoring systems.

The student's state can contain a variety of information. The information is typically divided into a student model and a dialogue model. The student model represents properties of the student such as:

- The student's knowledge level or skill level
- Misconceptions held by the student
- The student's affective state, which may include emotional state, engagement, interest in the topic and other variables.
- The student's attributes, which are psychological variables whose values are not expected to change. Common attributes include general intelligence, fluid vs. crystallized intelligence, verbal vs. special reasoning ability, working memory capacity, and many others.

The dialogue model represents features of the history of interaction between the system and the student, such as:

- The set of tasks that have been given to the student already. This is necessary to avoid accidentally assigning the same task twice.
- How many times a particular hint or feedback message has been given. This is necessary when systems try to vary the wording of such messages each time they are given, as this increases the chance that the student will read them.
- Whether the student received an unsolicited hint on the preceding task. This is necessary if the system's policy is to avoid giving two such hints in a row (Burton & Brown, 1982).
- Whether the student has been guessing too much, asking for too much help or not asking for enough help. This is necessary if the system is doing "meta-tutoring," that is, trying to teach the student how to use the system's help facilities properly

(Alevan, Stahl, Schworm, Fischer, & Wallace, 2003; Baker et al., 2006; Roll, Alevan, McLaren, & Koedinger, 2007, 2011; Zhang et al., 2014).

Although these possibilities open up a large design space, many investigations, including the one reported here, have concentrated on determining how a student model that just represents student competence can help in adaptive task selection.

Most tutoring systems often assign only one kind of task, typically a problem-solving task or a question-answering task. When a tutoring system must select among multiple types of tasks, then some systems divide competence into types that align with the types of tasks. For instance, ActiveMath (Melis et al., 2001) follows Bloom's taxonomy (Bloom & Krathwohl, 1956) and tracks three types of competence, corresponding to reading a text about a concept, studying examples of the concept and solving problems involving the concept. In our project, only one type of task (answering biology questions) is assigned, so only one type of competence is tracked.

3.2.2 Mastery learning and remediation

Unless the tutoring system covers only a small amount of content, the knowledge it covers is partitioned into modules (also called topics or units). Each module has its own set of tasks, and the sets are usually mutually exclusive. By the end of the course, students are typically required to have mastered every module in the course.

The modules are arranged in a partial order by prerequisite relationships. That is, if module A has module B as a predecessor in the partial order, then students should have mastered module B before beginning module A. Many courses arrange modules in a total order (linear sequence) as this often makes the teacher's job easier. The student's state lists the modules that the student has mastered. The adaptive task selector is usually

constrained to select tasks from modules that are not mastered and have their prerequisites mastered. The formal properties of this arrangement and algorithms for inferring the partial order from student data have been developed and generally refer to the partial ordering of modules as a *knowledge space* (Falmagne, Koppen, Villano, Doignon, & Johannesen, 1990).

Mastery learning (Bloom, 1984) refers to assigning tasks from a module until the students' competence in the module's knowledge has reached a certain threshold, called the mastery threshold. When mastery is reached, the student moves on to the next module in the sequence, which is sometimes called *move on when ready* or leveling up. If the system believes that a module has been mastered, and later evidence suggests that it has not been mastered, then mastered status can be removed, and tasks are once again assigned from the modules. This is called *remediation*.

In this paradigm, the traditional term "mastery" can be misleading. It refers to an operational concept not a cognitive one. If a module is mastered, then the task selector should no longer assign tasks from it. Each module has a "mastery criterion," which might be set so low that it amounts to acquiring mere familiarity with the contents of the module. Nonetheless, when the student's performance meets the criterion, the module is marked in the student's state as mastered (in the operational definition used here) even though its content is far from being mastered (in the cognitive sense that is not used here).

In order to apply the mastery criterion, some kind of assessment of the student's competence is needed. It can be embedded in the teaching tasks or done with testing tasks, which are designed to suppress changes in competence while they try to measure it. When the module includes tests, then the assessment can use conventional psychometric

techniques, such as item-response theory. When the teaching tasks are answering questions, then a typical embedded assessment is based on whether the student answers correctly on the first attempt. If they do not, then they get feedback and instruction that helps them learn. Thus, this kind of embedded assessments are trying to measure a constantly changing competence. Techniques for doing this will be described in the next section.

3.2.3 Embedded assessment

With test tasks, the student does not get feedback on their answer during the test or anything else that might serve as instruction and thus change the competence that is being measured. With instructional tasks, where students do get feedback and instruction on their answers, it is typical to measure competence based on the correctness of the students' first attempt. Although some systems also consider the number of attempts before success or the strength of the hints required for success, this does not appear to increase assessment accuracy much (Shute, 1995), so only the correctness of the first attempt is used here.

When assessing competence the knowledge taught in a module, there are different algorithms depending on whether the content is treated as a single factor or multiple factors. Let us consider single-factor algorithms first.

If the content is treated as a single factor, then all instructional tasks are considered to teach the same knowledge and all test items are considered to assess the same knowledge. Sometimes, it is reasonable to treat all tasks as having the same difficulty. If so, then a common embedded assessment technique is to use percentage correct on first attempt, but only count the most recent N tasks. For instance, the skill builder part of ASSISTments considers a student to have mastered a module when the student gets three tasks correct in

a row on the first attempt, that is, 100% correct on the most recent 3 tasks (Razzaq et al., 2007).

Alternatively, systems may assume that different tasks have different difficulties even though they all address the same knowledge. For test items with varying difficulties, the dominant method for computing competence is item-response theory (IRT). It is based on modeling the probability of a correct response with this formula (or ones similar to it):

$$P(S, T) = \frac{1}{1 + e^{-m(S,T)}}$$

$$m(S, T) = \text{competence}(S) - \text{difficulty}(T)$$

where S denotes the student, T denotes the task and P(S,T) denotes the probability that student S will get task T correct. Every task has a parameter, *difficulty(T)*. Although every student has a parameter, *competence(S)*, IRT assumes that its value is constant. Thus, the formulae above are used for assessing student competence and selecting tasks during testing, where competence is assumed not to be changing.

For updating the student's competence in an embedded assessment, where competence is assumed to be changing, one technique is a variant of IRT that that only counts evidence from the most recent N tasks. The general technique is called *Additive Factors Analysis* (Draney, Pirolli, & Wilson, 1995) because it adds other factors into the calculation of probability of correctness of a student S on a task T:

$$m(S, T, N) = \text{competence}(S) - \text{difficulty}(T) + \text{history}(N)$$

The *history(N)* factor represents evidence about the student's current competence gathered from the most recent N tasks. The parameter *competence(S)* is still an estimate of the student's initial, unchanging competence.

Different members of the Additive Factors Analysis family define $history(N)$ differently. Learning factors analysis (Cen, Koedinger, & Junker, 2006) defines it with:

$$history(N) = \gamma * percent_correct(N)$$

where $percent_correct(N)$ is the percentage of the most recent N tasks that were answered correctly on the first attempt. On the other hand, performance factors analysis (Pavlik, Cen, & Koedinger, 2009) defines

$$history(N) = \mu * successes(N) + \rho * failures(N)$$

where $successes(N)$ counts the number of tasks among the most recent N tasks that were answered successfully on the first attempt, and $failures(N)$ counts the number of tasks that were not answered correctly on the first attempt. Other members of the Additive Factors Analysis family have more elaborate definitions of $history(N)$ that distinguish tasks by their features, such as whether they are example-studying tasks or problem-solving tasks (Chi, Koedinger, Gordon, Jordan, & VanLehn, 2011).

Although only the formulae for predicting student performance have been given, there exist calibration methods that mine student data to provide values for the task difficulties, $difficulty(T)$, and the Additive Factors Analysis parameters such as γ , μ and ρ .

If data are not available for calibration, humans must estimate $difficulty(T)$ subjectively, which is difficult. Thus, a second approach, called Bayesian Knowledge Tracing (Corbett & Anderson, 1995), replaces the $difficulty(T)$ parameter with two parameters, $slip(T)$ and $guess(T)$. $Slip(T)$ is the probability that a student who has mastered the target knowledge will, despite their mastery, answer task T incorrectly. $Guess(T)$ is the probability that a student who lacks mastery will nonetheless answer task T correctly. Both $slip(T)$ and $guess(T)$ depend strongly on the form of the task's answer. For instance, if the task is a

true/false questions, then $guess(T)=0.5$ and $slip(T)$ is very small. On the other hand, if the task requires writing a complex mathematical expression, then $slip(T)$ is high and $guess(T)$ is very low. Thus, the task parameters are much easier to estimate subjectively with Bayesian Knowledge Tracing than with Additive Factor Analysis. Because the research reported here does not have calibration data, it uses Bayesian Knowledge Tracing (BKT). The technique will be described in detail later.

Both BKT and Additive Factors Analysis assume that all the tasks address the same knowledge, and that the tasks vary only in difficulty, or in the slip probability and guess probability. An alternative framework is to divide the knowledge taught by a module into several smaller pieces of knowledge, usually called *knowledge components* (Koedinger, Corbett, & Perfetti, 2012), and to assume that different tasks may address different sets of knowledge components even though all the tasks belong to the same module.

Although many practical systems, starting with BIP (Barr, Beard, & Atkinson, 1976), have made this assumption, it can make assessment more complex. If correctly solving a task requires applying two or more knowledge components (KCs), then failure to solve the task can be blamed on either or both of the knowledge components. This is called the assignment of blame problem (VanLehn, 1988). Although techniques exist to assign blame fairly, they dilute the evidence and slow down the assessment process (VanLehn & Niu, 2001). Thus, most practical tutoring systems, including the one presented here, try to analyze the student's response to a multi-KC task in such a way that they can determine which KCs were applied correctly and which were not. This works around the assignment of blame problem and allows the tutoring system to use BKT or Additive Factors Analysis in the simple form described earlier.

When a module is assumed to cover several KCs and different tasks are assumed to cover different KCs, then a mapping between tasks and the KCs they cover is necessary. Although this was earlier referred to as a question model, it is more commonly called a Q-matrix (Tatsuoka, 1996). It is difficult for human authors to develop a list of KCs and an accurate Q-matrix, in part because the authors are usually experts, so they have lost access to their naïve misconceptions and fragmented early knowledge, a phenomenon known as “expert blind spot.” (Nathan, Koedinger, & Alibali, 2001). Consequently, techniques have been developed to mine KCs and the Q-matrix from student performance data (Barnes, Stamper, & Madhyastha, 2006). Unfortunately, a great deal of data is required, and assumptions are still necessary (e.g., one must decide how many KC there are). Another technique is to start with crudely defined KCs from human authors, then use data to split them into more specific KCs (Cen et al., 2006). Once again, large amounts of student data are required. A novel contribution of this chapter is showing how both the KCs and Q-matrix can be generated automatically by generating the tasks (biology questions, in particular) from a knowledge base.

Once a Q-matrix has been generated, the partial ordering of modules and even the modules themselves are superfluous and be omitted. Instead of indicating whether a module has been mastered, the system can track whether a knowledge component has been mastered. The pre-requisite structure of the modules can be replaced by a pre-requisite structure on the knowledge components or more elaborate interrelationships (Kumar, 2006; Vassileva & Deters, 1998). Alternatively, both the module structure and the inter-KC structure can be retained; students stay in a module until mastered, and the pre-requisite structure among KCs partially determine the task selection within a module.

If a system employs a pre-requisite structure among KCs, then it can be used to solve the assignment of blame problem. Tasks are authored to have just one new KCs, and all the other KCs required for doing the task are included in the prerequisites. When the student fails to answer the task correctly on the first attempt, all the blame is cast on the new KC. Because the other KCs have been mastered, they probably are not the source of the error and thus receive no blame.

When students may work on tasks even when the prerequisites have not yet been mastered, and they succeed, then some systems update success on the prerequisite KCs as well as the target KCs (Melis et al., 2001; Weber & Brusilovsky, 2001).

Our system uses knowledge components, but does not assume a pre-requisite structure among them. Our system does not have multiple modules, but instead assumes all tasks belong to the same module.

3.2.4 Learner control

Any tutoring system that can select tasks can also display its recommendations to students and let the students select the tasks. This design variable is a special case of the general issue of learner control, that is, how much control to give to learners vs. the system (Kay, 2001). Some systems merely present their recommendations without trying to explain them (Brusilovsky, 2007). Others give the student relevant information for making a choice, such as the system's assessment of the student's current competence (Bull & Kay, 2013), a capability called *open learner modeling*. Some open learner modeling systems also provide for comparison the competence or task selections of other students, in which case the capability is called social learner modeling (Hosseini, Hsiao, Guerra, &

Brusilovsky, 2015; I-Han Hsiao, Bakalov, Brusilovsky, & Konig-Ries, 2011, 2013; I-Han Hsiao & Brusilovsky, 2012, submitted).

A challenge with open and social learner modeling is designing an easily understood display of the data that the student needs to know for making an informed decision. In this case, students are assumed to be working in a module, and they do not have choice about that, so what they need to see is the assessed competence and/or task selections of themselves and possibly other students. Our system has 162 knowledge components, so simply displaying a 162-bar chart of competencies would not be easily understood. Moreover, a display of the Q-matrix that pairs tasks with KCs would be even harder to understand.

Thus, this project does not allow learners any control over task selection because they would lack to information needed to participate fully. That is, it does task selection rather than task recommendation.

3.2.5 Task selection: Look ahead

Most task selection methods can be viewed as rating the expected utility of each task, and then choosing the task with the maximal expected utility. The expected utility of a task is sum of the utilities of the possible outcomes of the task multiplied by the probability that the outcome will occur. The utility of a student state can include many issues, such as the student's competence, engagement, understanding, etc. Options for calculating utility are discussed in the next section. This section discussed another design variable, which is how far to look ahead when collecting the outcomes of a selected task.

From the point of view of the tutoring system, tutoring is a sequential decision making problem. The student is initially in some state, State1. The tutoring system doesn't know

exactly what that state is, but it has a probabilistic estimate of it. That is, it has a probability for each logically possible state. Now the tutor makes its first decision, about what task to present. Presenting the task causes the student to move to State2 and to respond to the task. Once again, the tutor cannot observe State2, but can use the response of the student to help infer a probability distribution across all possible states for State2. Now the tutor makes its second decision, about what kind of feedback to give the student. Receiving the feedback drives the student to State3, and the student may also emit a response. This process repeats for a long time, generating a sequence of student states.

Now back when the tutor was making its first decision, it could look ahead to the next state or even further. Looking ahead to State2 means predicting what State2 would be for each possible choice of task. For each predicted version of State2, the tutor runs the utility measure to calculate the expected utility of that version of State2. Then it chooses the task that leads to the version of State2 with maximal utility. This is one-state look-ahead. Similarly, it could look ahead to State3, State4 or even further.

This kind of computational problem is called a POMDP—Partially Observable Markov Decision Process. There are methods to solve such problems, but the amount of computation required depends strongly on the branching factor, which is number of possible tasks the tutor can choose (first kind of branch) and the number of possible types of feedback the tutor can give (second kind of branch). A few tutoring systems do use POMDP algorithms or related algorithms (Cakmak & Lopes, 2012; Clement, Oudeyer, Roy, & Lopes, 2014; Lopes, Clement, Roy, & Oudeyer, 2013; Muldner & Conati, 2007; Murray & VanLehn, 2000, 2006; Murray et al., 2004; Pek & Poh, 2000a, 2000b; Rafferty et al., 2011; Whitehill, 2012).

In the most extreme case, the tutor keeps looking ahead until it generates state that have the student reaching mastery with high probability. Clearly, this much computation is too much to do every time tutor has a decision to make, so it is typically done once and all the decisions along successful paths are saved. That is, the tutor plans ahead and saves the plan. Some tutoring systems take this approach to adaptive task selection (Peachy & McCalla, 1986; Vassileva & Deters, 1998), but it is computationally feasible only in highly constrained applications.

However, most systems do not look ahead at all. They apply the utility measures directly to the decision alternative themselves. That is, when the student is State1, utilities of each task selection are compute. This save computation.

Moreover, it is likely that looking ahead only begins to have added value over this method when utilities can be applied to State3 or State4. That is, the tutor is considering pairs of actions instead of just a single action. This would be advantageous, for instance, if the tutor was teaching a concept and could present minimally contrasting examples as consecutive tasks.

Our project had a large branching factor in that the tutor must choose among 48 questions to pose to the student. Thus, it does no looking ahead, and merely evaluates the expected utility of each task, then chooses the task with maximal expected utility.

3.2.6 Task selection: Utility issues

The whole essence of adaptive task selection is that the utility of a task can be high when a student is in one state and low when the student is in a different state. Thus, it is helpful to crudely categorize states into three phases, and then discuss the utility issues for each state separately.

Theorists of cognitive skill acquisition (Anderson, 1982; Fitts & Posner, 1967; Koedinger et al., 2012) often distinguish three phases:

- *Understanding*: The student reads about a concept or skill, sees an initial example, or discovers it.
- *Refinement*: The student changes the content of the concept or skill, refining it by removing superfluous or incorrect features and perhaps adding new features.
- *Strengthening*: Although the content of the concept or skill no longer changes, the students' use of it becomes more fluent with practice. That is, time, errors and cognitive resource usage all decrease.

When students are in the understanding phase, tasks that convey the target knowledge clearly and succinctly have high utility, but tasks that activate prior knowledge might need to come just before them (Bransford, Brown, & Cocking, 2000). There are many issues here, such as: whether a general description should come before or after an example of it; whether pictures are more effective than text; whether presenting parts should be presented before or after the whole, etc. Although heuristics exist for making these choices when designing instruction (Patton, Chao, & Reigeluth, 1986; Reigeluth, 1999), little work has been done on adapting them for use in dynamic task selection.

When students are in the refinement phase, a high utility is placed on tasks that encourage generalization and discrimination (Koedinger et al., 2012). Learners have a tendency to incorporate the wrong features into the target knowledge when first acquiring it, a phenomenon called encoding specificity. The tasks selected during refinement phase should help students remove incorrect features and add correct ones, so the emphasis is on the variety of tasks and use of closely contrasting pairs of tasks. Once again, heuristics for instructional design exist (Patton et al., 1986; Reigeluth, 1999). Although early work on

concept acquisition occasionally addressed this phase (Park & Tennyson, 1980), there has been little recent work on adaptive task selection for refinement.

When students have are practicing in order to strengthen their memory, then the utility of tasks shifts towards keeping students engaged by making the tasks neither too easy nor too hard—the so-called *zone of proximal development* (Chaiklin, 2003). When tasks must be repeated, the spacing them as far apart as possible without making them too difficult, can lead to longer retention (Cepeda, Pashler, Vul, Wixted, & Roher, 2006; Lindsey, Mozer, Huggins, & Pashler, 2013; Lindsey, Shroyer, Pashler, & Mozer, 2014; Pavlik & Anderson, 2005). In order to get more tasks completed per unit of time, short-duration tasks are often given higher utility (Pavlik & Anderson, 2005). Although example-studying tasks may be more effective than problem-solving tasks during the first two phases, the reverse is true in the strengthening phase, a phenomenon known as the expertise reversal effect (Kalyuga, Ayres, Chandler, & Sweller, 2003).

Unfortunately, most tutoring systems have used heuristic rules for selecting tasks that obscure the utilities that they are trying to maximize. Some examples are:

- One ActiveMath (Melis et al., 2001) rule says that if the user's problem solving competence for a particular KC is less than 0.3, then assign four problem solving tasks of gradually increasing difficulty: 0.3, 0.3, 0.5 and then 0.7.
- The SQL-Tutor (Mitrovic, Martin, & Mayo, 2002) uses Bayesian reasoning to predict the number of errors that that a student will make, then ranks problems by how close the number of predicted errors is to an desired number of errors. The desired number of errors gradually increases as the student's overall competence increases.
- A common technique for selecting tasks is based on ordering the tasks by difficulty, then jumping to higher difficulty tasks if the students succeeds on a task, and jumping to a lower difficulty task if the student fails on the task (G. Corbalan, Kester, & van Merriënboer, 2008; Salden, Paas, Broers, & Van

Merriënboer, 2004; Salden, Paas, Jeroen, & van Merriënboer, 2006; Weber & Brusilovsky, 2001). This assumes that the candidate tasks all address the same KC.

In short, all the issues that bear upon instructional design are also relevant to adaptive task selection. However, the issues must be formalized as rules or utilities so that they can be applied in real time to select tasks.

Our contribution to this effort is to define a particular utility and test its efficacy. Although no great claims can be made about its merits based on just one study, perhaps the methodology of carefully defining a utility may inspire others to formulate their task selection algorithms the same way. Progress on adaptive task selection research could perhaps be faster if there were greater clarity about the basis for making the decisions.

3.3 Description of the tutoring system

From the students' point of view, the tutoring system is extremely simple (see Figure 10). A question appears on the screen along with a box. When the student has typed an answer into the box, the student clicks an Enter button, and the system presents feedback. This continues until the student has worked for 30 minutes.

The process of how questions were generated from the semantic network knowledge base was detailed in our previous paper (Zhang & VanLehn, submitted), so this section describes the rest of the technology.

Question: What is/are the raw materials of calvin cycle in photosynthesis ?

Your answer to the question:

water, co2

Feedback:

You mentioned that

carbon dioxide is the reactant of calvin cycle in photosynthesis

But you missed the following points:

atp is the reactant of calvin cycle in photosynthesis

nadph is the reactant of calvin cycle in photosynthesis

ribulose bisphosphate is the reactant of calvin cycle in photosynthesis

Figure 10 The system feedback on student's answer

3.3.1 Populating the Q-matrix

All the questions were generated from a knowledge base comprised of semantic relations. For example, here are the relations that represent that photosynthesis happens in the chloroplast and consist of the light action and the Calvin cycle:

Relation1: has(photosynthesis1514, site, chloroplast18798).

Relation2: has(photosynthesis1514, subevent, light_reaction13426).

Relation3: has(photosynthesis1514, subevent, calvin_cycle13425).

Relation4: has(photosynthesis1514, instance_of, photosynthesis).

Relation5: has(chloroplast18798, instance_of, chloroplast).

Relation6: has(light_reaction13426, instance_of, light_reaction).

Relation7: has(calvin_cycle13425, instance_of, calvin_cycle).

The terms ending with numbers are instances of classes. For example, photosynthesis1514 is an instance of photosynthesis. The knowledge representation is described in more detail in another paper (Zhang & VanLehn, submitted).

The Q-matrix was populated by listing the knowledge components used to generate a question. However, only some relations correspond to knowledge components. For instance, relations 1, 2 and 3 are each knowledge components. On the other hand, instance_of relations, such as relation4 to relation7, are essentially for bookkeeping and are not considered to be knowledge components. For example, the machine generated question “What are the two stages of photosynthesis?” has two corresponding knowledge components: relations 2 and 3.

3.3.2 Understanding student answers

As mentioned earlier, adaptive task selection requires embedded assessment, which entails deciding if the student’s response exhibits application of all the knowledge components that it should have, according to the Q-matrix. Because the student’s response is in typed natural language, our tutoring system has to understand it, at least partially.

The previous methods for understanding natural language could be classified into two types. One is to treat student’s answer as a bag of words, and the other is to extract semantic relations from the text and make inference on the relations. The latter method is rarely used because it requires a great deal of computational linguistic technology, including a robust parser, semantic interpreter and domain-specific inference rules

(Makatchev, Hall, Jordan, Pappuswamy, & VanLehn, 2005). Along with most other work on automated text understanding in education, we used a bag of words technique.

In bag of words methods, the “word” need not be the original word in the sentence. It could be the stem of the word, a syntactic labeling, and other features extracted from a student’s answer. In order to force generalization and reduce computation, words are translated into vectors in a vector developed by principle-components analysis or other compression techniques. The vectors in the student’s answer are processed with classification methods such as Naïve Bayes (Lewis, 1998), decision trees (Apté, Damerau, & Weiss, 1994) , or K-nearest neighbor (Wiemer-Hastings, Wiemer-Hastings, & Graesser, 1999) or cosine distance (Wiemer-Hastings et al., 1999). Training the classifiers for these methods requires considerable data which are hand-coded by experts. Since we did not have such data, we could not use these methods.

We used a method based on keywords. The keywords of a knowledge component were extracted from the relation representing the knowledge component. For example, the keyword for relation2 was “light reaction” and the keyword for relation3 was “calvin cycle.” To find synonyms for the keywords, we use on-line dictionaries like WordNet and Theaurus. The dictionaries recognized common synonyms like “CO2” and “carbon dioxide,” but did recognize unusual terms like GA3P is the abbreviation of glyveraldehyde-3-phosphate. There were only a few of these, so they were added manually.

Figure 11 presents the response-processing algorithm.

1. Take question id and student answer as the inputs
2. Extract the knowledge components associated to the question and the corresponding keywords
3. Visited the associated knowledge components one by one, and for each knowledge component:
 - a. Visited the corresponding keywords one by one, and for each keyword:
 - i. Check existence of the keyword in the student's answer.
 - ii. If it is existed, go to (a.) to check next keyword,
 - iii. If it is not existed, get all the synonyms of the keyword.
 - iv. Visit all the synonyms one by one, and for each synonym:
 - Check existence of the synonym in the answer.
 - If it is existed, go to (a.) to check next keyword.
 - If it is not existed, go to (iv.) to check next synonym.
 - b. If all the keywords (or the synonym) for the knowledge component can be found in the answer, the knowledge component is tagged as "correct".
 - c. If at least one keyword for the knowledge component is not existed in the answer, the knowledge component is tagged as "incorrect".
 - d. Update the probability of mastery of the knowledge component for the current student.

Figure 11 Analysis of the student's response

3.3.3 Checking the accuracy of the natural language understanding

In order to test whether our simple text analysis mechanism was able to achieve satisfactory accuracy, we ran a pilot study comparing decisions made by the machine and the decisions made by a human. This section describes the study.

While the human judge was making decisions, he could see a table that contained 4 columns: questions, students' answers to the questions, the natural language descriptions of the associated knowledge components to the questions, and the column used to record human decisions. The natural language descriptions of the knowledge components were also used as part of feedback to students. Given the table, the human judge was asked to annotate 'Y' or 'N' for each knowledge component. The human judge was asked to put 'Y' when he thought the knowledge component was mentioned in the answer and 'N' when the knowledge component was not mentioned. In total, there were 153 questions.

Since one question usually covered more than one knowledge components, the total number of decision points was 391.

Kappa was calculated to measure the agreement between the human judges Y/N judgments and the algorithm's. The kappa was 0.802. The result suggested that machine and human expert agreed with each other at most of time. So the simple assessing mechanism did work in our case.

Machine/Human	Yes	No
Yes	124	5
No	31	230

The confusion matrix (see Table 12) showed that most of the errors came from false negatives. The system could only recognize synonyms at the word level but failed to detect synonyms at the phrase level. Here is an example:

Question: What are the 4 stages of cyclic photophosphorylation in light reaction?

Student's answer:

conversion of light energy into chemical energy

absorption of photon

splitting of water

Generating energy carrier

Although the question covered four knowledge components, only one raised a false negative. The knowledge component was "chlorophyll absorbs photons." The keywords

of the knowledge component were: chlorophyll, absorb, photon. The human judge decided that the knowledge component was mentioned because of the phrase “absorption of photon”. But the system decided that the knowledge component was not mentioned because the word “chlorophyll” was not found in the answer and the noun form of the keyword “absorb” was not recognized. Although it is probably feasible to improve the accuracy of the text analysis method, that was not a focus of the research, so it will be left for future work.

There is a potential issue that was not fully evaluated in the kappa test. Both the human judge and the algorithm ignored the terms that appeared in the answer when they should not have. For example, some students included carbon dioxide in their answer to the question “What does photosynthesis make?” Clearly the probabilities of masteries of all the associated knowledge components to the question needed to be somehow decreased as a punishment. Fortunately, we found that students in the pilot study seldom made these errors.

3.3.4 Embedded assessment

When an expected knowledge component is mentioned in the student’s answer to a question, the system’s estimate of the student’s proficiency in using the knowledge component should be increased. On the other hand, if an expected knowledge component is not included in the answer, its estimated proficiency should be decreased. We used Bayesian Knowledge Tracing (Corbett & Anderson, 1995) to represent and update the proficiencies.

In Bayesian Knowledge Tracing, for each student and each knowledge component, there is a single number between 0 and 1 representing the probability that the student has

mastered the knowledge component. This number changes as questions are answered and evidence about the student's proficiency changes. When there is no information about a student's initial competence, it is typical to set the initial value for probability of mastery to 0.5 for all knowledge components. If the student repeatedly fails to mention the knowledge component, the probability falls. If the student repeatedly does mention the knowledge component, then the probability rises.

Although the algorithm can be explained by reference to Bayesian networks or hidden Markov models (VanLehn, 2008), it will be explained here with simple algebraic equations and graphs, from (Van de Sande, 2013), in order to give a more intuitive sense for how the algorithm works.

When a question has been answered, the BKT algorithm runs once for each knowledge component that should have been mentioned by the student. For each knowledge component, the algorithm is given the following inputs:

- B – the probability of mastery *before* the questions was asked
- G – the probability of a lucky *guess*. That is, the probability that the student will mention the knowledge component even though the student has not mastered it.
- S – the probability of a *slip*. That is, the probability that the student will fail to mention the knowledge component even though the student has actually mastered it already.
- L – the probability of *learning*. That is, the probability that in the course of answering the question and receiving feedback, the student's competence on this knowledge component will change from unmastered to mastered.
- E – the *evidence*. This is “correct” if the student mentioned the knowledge component and “incorrect” if the student failed to mention it.

The algorithm outputs

- A – the probability of mastery *after* the student has answered the question and received feedback on their attempt.

Different questions have different values for the slip (S) and guess (G) parameters, and the values can be estimated from the format of the question. For instance, a true-false question has $G=0.5$. An open-response questions has a low G and a high S. When the answer to a question should mention multiple KCs, the same S and G parameters are used them all.

However, the probability of learning (L) is difficult to estimate, so it is typical to assume that it is the same for all students and all questions, then adjust it to get reasonable results overall.

The BKT algorithm has two stages. During the first stage, it updates the estimated probability of mastery *before* the question was answer based on the student's answer. During the second stage, it calculates the probability of mastery *after* the question and feedback were processed by the student. The second stage is easier to describe, so let's start with it.

Let BE be the output of the first stage. It represents the probability of mastery *before* the question was posed to the student, but it has been updated to reflect the evidence provided by the student's answer. That is, BE is a more accurate version of B, because B represents the probability of mastery *before* the question was posed and it does not include the evidence provided by the student's answer.

The formula for A is $BE+(1-BE)*L$. That is, there are two cases. (1) If the student had mastered the knowledge component before processing the question and feedback, then the student still has mastery. Thus, the first term is BE. (2) If the student had not mastered the knowledge component before processing (the chance of that is $1-BE$), then there is a chance (L) that the student will learn during the processing. Thus, the second

term is $(1-BE)*L$. This is clearly a rather simple model of learning. One could imagine having multiple probabilities of learning depending on what the student did during the processing of the question. However, the simple BKT model has proven useful in many intelligent tutoring systems, and attempts to improve it by adding more parameters have provided only marginally more accuracy (Shute, 1995).

In order to calculate BE, it helps to consider the joint probability distribution, which is a table showing all possible cases and their probabilities (see Table 13). There are four cases, and their probabilities (shown in the last column) sum to 1.0. When a question is answered correctly, then two of the four rows can be eliminated: rows 1 and 4. Of those two remaining cases (rows 2 and 3), only one has the student at mastery. Thus, given that the question was answered correctly, the probability of mastery is $P(\text{row 2}) / [P(\text{row 2}) + P(\text{row 3})]$ or $B*(1-S) / [B*(1-S) + (1-B)*G]$. On the other hand, if the question was answered incorrectly, that leaves rows 1 and 4, so the probability of mastery is $P(\text{row 1}) / [P(\text{row 1}) + P(\text{row 4})]$ or $B*S / [B*S + (1-B)*G]$.

Table 13 Joint probability distribution for answering the question			
Mastered?	Slip?	Guess?	Probability
Yes	Yes	Not applicable	$B*S$
Yes	No	Not applicable	$B*(1-S)$
No	Not applicable	Yes	$(1-B)*G$
No	Not applicable	No	$(1-B)*(1-G)$

In short, BKT has two slightly different formulas for BE depending on whether the answer was correct or incorrect. The left pane of Figure 7 shows how these two values vary depending on B, S and G. This means that A, the probability of mastery after the student processed the question, also has different values depending on whether the answer was correct or incorrect, like the right pane of Figure 12 shows.

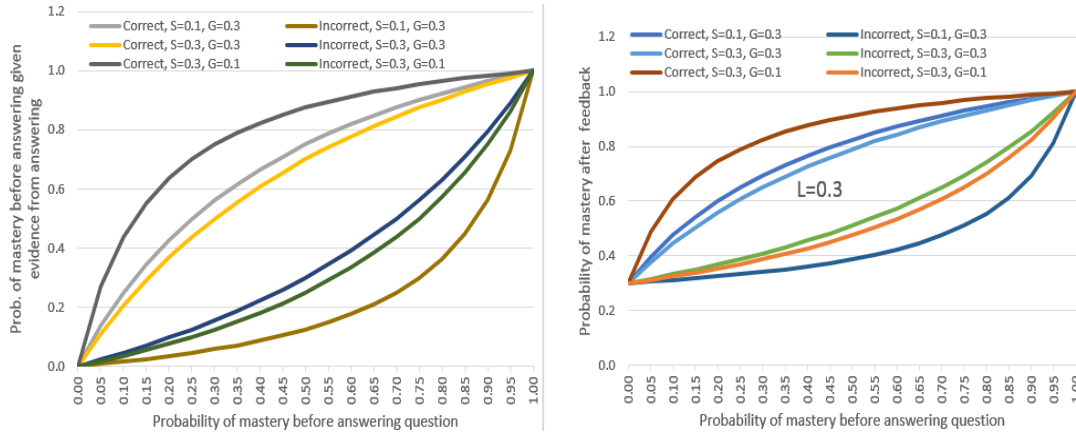


Figure 12 Probabilities of mastery calculated by BKT

3.3.5 Adaptively select the next question

Suppose that the student has answered a question, the system has updated the probabilities of mastery of the knowledge components that should have been included in the answer, and it is time to select a new question to pose to the student. The system goes through every question in the database that has not yet been asked, calculates its utility, then selects a question with maximal utility.

The utility is a composite of two factors that correspond to the two stages of the BKT algorithm. That is, we want to maximize A, the probability of mastery after the student has processed the question, by maximizing $L \cdot (1 - BE)$, the probability of learning, and by maximizing $BE - B$, the probability of raising the estimated probability of mastery by getting

evidence from this question. The latter is necessary because some questions, namely those with a high probability of guessing and slipping, provide much less evidence than others.

However, we don't know BE yet, because the student has not answered the question, so we must do a weighted sum of the two cases: a correct answer and an incorrect answer. We weight the two cases by their probability of occurrence, namely B and (1-B) respectively. Thus,

$$\text{Utility} = B*[L*(1-\text{BE}_{\text{correct}}) + (\text{BE}_{\text{correct}}-B)] + (1-B)*[L*(1-\text{BE}_{\text{incorrect}}) + (\text{BE}_{\text{incorrect}}-B)]$$

Where $\text{BE}_{\text{correct}}$ is the formula given above for the case where the answer was correct, and $\text{BE}_{\text{incorrect}}$ is the other formula for BE. Figure 13 shows how utility varies depending on B and the other parameters.

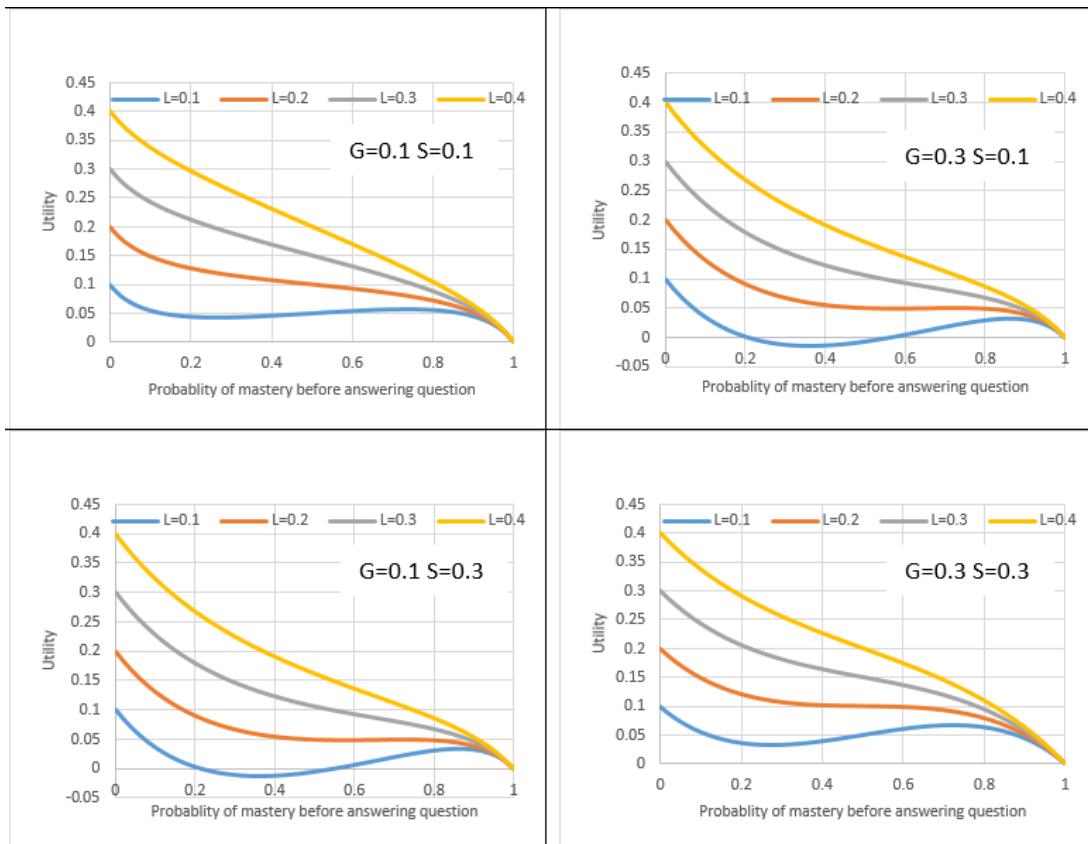


Figure 13 Utility of a question vs. probability of mastery

When the learning rate L is at least 0.3, low probabilities of mastery are preferred. This makes sense, as unlearned knowledge components have a greater chance of becoming learned. But when the learning rate is less than 0.3, then the dominant term in the utility formula is the increase due to evidence i.e., $(BE-B)$. Thus, there are bumps around 0.15 and 0.85 because that is where BE_{correct} and $BE_{\text{incorrect}}$ are changing rapidly (see Fig. 3). Thus, the curves make intuitive sense.

We chose $L=0.3$ because all the knowledge components in the teaching domain appeared to be simple to learn, e.g. photosynthesis produces oxygen, and a higher learning rate is preferred.

When a question is associated with multiple knowledge components, we took the average of the utilities of the knowledge component utilities as the question's utility. This treats all knowledge components as equally important, which seems reasonable in our context. It could be argued that taking the sum of the utilities is more logical. However, that means a question with more knowledge component has more utility than a question with few knowledge components. Moreover, since knowledge component with low probability of mastery are preferred, this would mean preferring questions that required lots of knowledge that was highly unfamiliar. Clearly, this would be a bad policy. Thus, we define question utility as the average rather than the sum of the relevant knowledge component utilities.

3.4 Evaluation

To evaluate our adaptive question selection method, we conducted a between-subject experiment, where the experiment group had biology questions that were adaptively

selected by the system and the control group was treated with inverse adaption, which meant that the question with lowest utility was always selected as the next question. The study was IRB approved. All the participants of the study were college students who were not biology majors. Students who participated in the study earned \$10 per hour. There were 42 students in total, 21 students in experiment group and the rest 21 subjects in control group.

Test

Which of the following does NOT happen in cyclic photophosphorylation?

- ATP is produced
- Electron transport occurs in the photosynthetic membranes
- Light energy is utilized
- NADPH is formed
- Don't know

Figure 14 Sample test question

3.4.1 Procedure

The experiment was divided into three parts: pre-test, training session where the difference took place, and the post-test. Pre-test and post-test used the same set of multiple choice questions with different order. There were 10 questions, such as the one in Figure 14, in the test set. Besides original choices of the questions, we added “Don’t know” bullet to each test question. Because performance in the test did not affect participant’s compensation, we believed that our participants would select the “Don’t know” button instead of guessing the correct answer when they had no idea about the answer. There was no time limit for pre-test and post-test. Students usually spent 5 to 10 minutes in each test. The test was closed book. Students had to work on their own. The training session lasted

30 minutes and participants were allowed to use a researcher provided biology textbook as the learning material to answer the questions. As mentioned earlier, students needed to type their answers to the training questions. There were 48 machine generated training questions stored in the database, but students answered only as many as they could in the 30 minutes allotted for training. The whole procedure of the experiment was summarized in Table 14. Procedure of the experiment.

Table 14. Procedure of the experiment			
	Time	Number of questions	Auxiliary material
Pre-test	5-10 minutes in average	10 multiple choice questions	Close book
Training Session	30 minutes	Answered 14.8 out of 48 questions in average	Open to a provided textbook
Post-test	5-10 minutes in average	10 multiple choice questions	Close book

3.4.2 Results:

Students were asked to answer as many questions as they could during the 30 minute training period. Students in both groups answered on average exactly 14.76 questions. This suggests that the set of questions were equally difficult. Thus, any difference in pre-to-post gain is probably due to the exact nature of the questions chosen.

The experiment aimed to answer two questions:

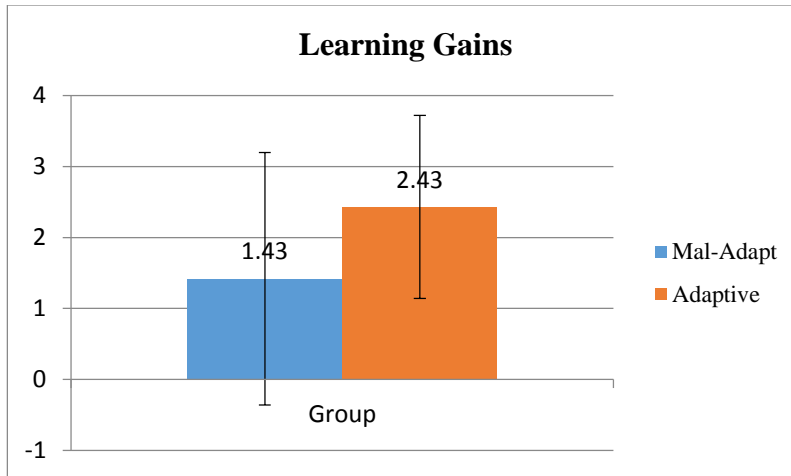
1. Could students learn with the machine generated questions?
2. Could students learn more with adaptive selection than the mal-adaptive selection?

To answer the first question, we compared the mean of the students' scores on the post-test to the mean of the students' scores on the pre-test for control group and experiment

group separately. According to T-test, students in both of the groups increased their test score significantly. Students in control group scored 3.32 (SD=1.57) in pre-test and increased to 4.74 (SD=1.24) in post-test. Two-tailed T-test reported the p value as 0.0023, $t=3.26$ and Cohen's $d=1.01$. Students in experiment group scored 2.86 (SD=1.49) in pre-test and increase to 5.29 (SD=1.52) in post-test. Two-tailed T-test reported the p value smaller than 0.001, $t=5.22$ and Cohen's $d=1.61$.

To compensate for students' differing prior knowledge, learning gains were used to test for a difference between the two groups. The learning gain was calculated as the difference of scores in pre-test and post-test. The average learning gain of control group was 1.42 (SD=1.78), and the average learning gain of experiment group was 2.43 (SD=1.29). A two-tail T-test reported p value was 0.043, $t=2.097$ and Cohen's $d=0.65$. So the adaptive question selection did significantly increase students' learning compared to mal-adaptive selection of questions. See Figure 15 and Table 15.

Table 15. Comparison between experimental group and control group			
	Pre-test	Post-test	Learning gain
Mal-Adaptive	3.33(SD=1.56)	4.76(SD=1.22)	1.43(1.78), p=0.0023, d=1.01
Adaptive	2.86(SD=1.49)	5.29(SD=1.52)	2.43(1.29), p<0.001, d=1.61



The difference is significant with $p=0.043$, $t=2.097$, $d=0.65$

Figure 15. The learning gain of adaptive treatment group vs. the learning gain of mal-adaptive treatment group

Instead of comparing learning gains between the two groups, an alternative way is to compare students' post-test scores with their pre-test scores as covariates in an ANCOVA. Applying an ANCOVA requires that the pretest score is linearly correlated with the posttest score (Eric, 1998), which it was in this case ($p=0.0089$). Although students with adaptive treatment scored higher in the post-test than those with counter-adaptive treatment, the difference of their adjusted post-test scores was only marginal significant ($p=0.075$, $F=3.348$, $d=0.58$). See Figure 16, left. However, we noticed that there were 8 students who got extremely low scores (<2 out of 10) in pretest. These students could be thought to have no prior knowledge in the domain at all, so adaptive question selection became meaningless to these students. The difference between the two groups turned out to be significant after the 8 students were removed ($p=0.034$, $F=4.933$, $d=0.77$). See Figure 16 right.

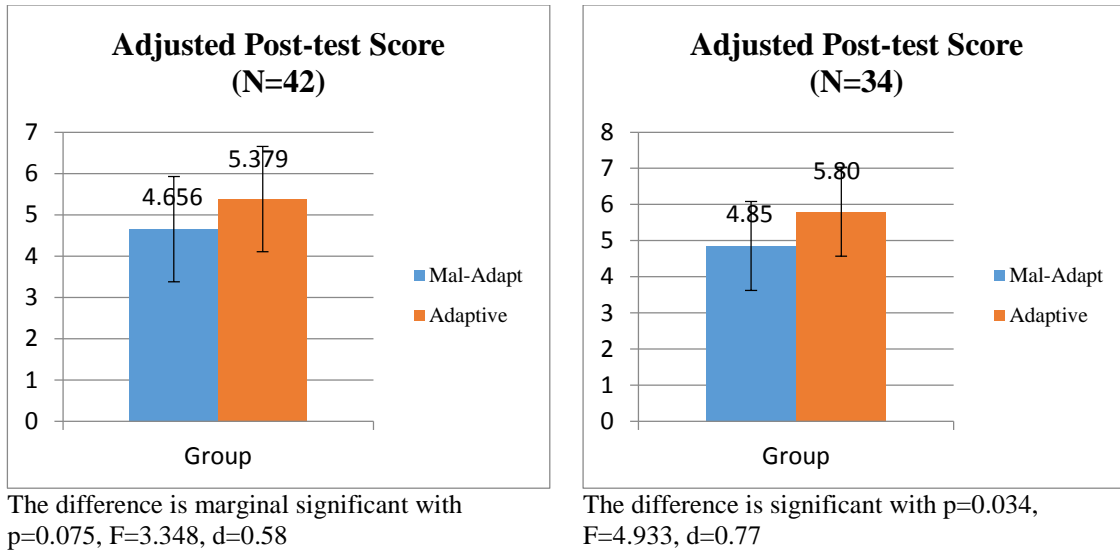


Figure 16 The adjusted post-test score of adaptive treatment group vs. that of counter-adaptive treatment group.

The difference between gain scores of the two groups is an unbiased estimation of the treatment's effect when gain score is not affected by pretest score. To test this assumption, we calculated Pearson correlation coefficient between the pretest score and the gain score in our experiment. In a result, the gain score was moderately correlated to the pretest score ($r=-0.61$). To obtain a gain score that was independent with the pretest score, we normalized the original gain score by dividing it with the difference between the pretest score and the total score. The normalized gain score was less correlated with the pretest score ($r=-0.40$), and the difference between the normalized gain scores of the two groups was still significant ($p=0.023$, $F=5.56$).

3.4.3 Discussion

Three commonly used measures of learning were used to test the effect of adaptive treatment. Gain scores and adjusted normalized gain scores showed that the treatment made a significant difference. ANCOVA suggested that the treatment was only marginally

reliable with all the participants included, but significantly reliable after removing the participants who had no prior knowledge. It did intuitively make sense because adaptive selection would be degenerated to be random selection for the students who had zero competence on all the knowledge components. Thus, these results suggest that adaptive question selection is associated with larger learning gains than mal-adaptive question selection.

3.5 Conclusion

Adaptively questions selection was not a new idea. However, we managed to provide an automatic way to build the adaption into a system. Except for providing a few synonyms for esoteric biology terms, no human knowledge was inserted into the question generation and adaptive question selection process. We also found that a simple method for analyzing students' answer in natural language was effective, in that it allowed the system to update its estimates of the student's probability of mastery of the knowledge components and its judgments agreed well with human judge. A between-subjects experiment was conducted to evaluate the effectiveness of the adaptive selection algorithm. Students with adaptive question selection had larger learning gains than those with mal-adaptive question selection, and both of the students scored higher on the post-test than the pre-test.

Although the system achieved satisfying results, it did have many aspects that could be improved. One issue was that the system was only able to detect the presence or absence of the correct elements in student's answer but failed to detect the presence of wrong elements. The problem could be solved in two different ways: one would be to figure out common error items for each question, so that the system can recognize them in student's

answer. Another would be to make the system be able to generate multiple choice questions to avoid the trouble of evaluating student's answers in natural language. The key technique for both of the two methods is to figure out wrong elements for the current generated questions. This would be part of our future work.

CHAPTER 4

MAPPING QUESTIONS TO THEIR RELATED PARAGRAPHS IN THE TEXTBOOK

4.1 Task definition

When open response questions are used in intelligent tutoring system, they can be organized as dialogs as AutoTutor (Graesser et al., 2004) did, so that students can learn deeply by going through the dialogs. Such a dialog has students generate an initial answer and then guides them to expand the answer in more detail. However, designing such dialogs is a very time consuming for domain experts. Reading related paragraphs in the textbook could provide an alternative and equally beneficial way of learning. Presenting the text and questions together may help students to learn how to generate questions, which is known to be an important skill in learning (Rosenshine, Meister, & Chapman, 1996). Therefore, we designed an algorithm to match questions to their related paragraphs in the textbook.

The mapping task is defined as the following: given a number of questions and a set of paragraphs in plain text that is also referred to “the text” in the rest of the chapter, find reasonable connections between the two groups, where a connection consists of one question and one paragraph. The same paragraph may be connected to several questions, and the same question may be connected to several paragraphs.

Suppose that there are n questions, $Q = \{q_1, q_2, \dots, q_i, \dots, q_n\}$, and m paragraphs, $D = \{d_1, d_2, \dots, d_i, \dots, d_m\}$, then the goal is to generate a related set of connections for each q_i , and one connection can be annotated as (q_i, d_j) . Because any paragraphs can be mapped to any set of questions, there should be 2^m possible connections for each question.

4.2 Dataset

The goal was to connect related paragraphs to each question, so it was important to define what a related paragraph is. A related paragraph to a question here was interpreted as a paragraph that could help in answering the question. In particular, the related paragraph should share similarities with the answer to the question.

We managed to generate good photosynthesis questions as well as their answers from a semantic network in our early work. While the questions were being generated, each of them was associated with a list of knowledge components, which were essentially the relations in the semantic network. The answers to the questions were just another representation of the relations. For example,

What does photosynthesis need? the machine generated question
Sunlight and water molecule are the reactants of the light reaction in photosynthesis. the answer
has(light_reaction13426,raw_material,sunlight13428). the corresponding knowledge components
has(light_reaction13426,raw_material,water_molecule2658).	

Because the corresponding knowledge components were good generalization of the answers, they could be used to locate the related paragraphs. To take the advantage of our previous work, the question set consisted of machine-generated questions, each of which was associated to its related knowledge components already.

The question set domain was photosynthesis, so we chose the chapter “photosynthesis” in the biology textbook published by OpenStax College as the text that was going to be matched against.

There is another benefit of connecting machine-generated questions to the paragraphs in the textbook. Based on the connections and the correctness of a student's answer to the machine generated question, it should be easy to infer the student's level of mastery on different knowledge components, and this information could be potentially used for generating adaptive reading materials for the student by highlighting the parts where the student should pay more attention.

Therefore, the machine-generated questions from our previous work, which came with their related knowledge components, formed the question set, and the photosynthesis chapter in the textbook formed the text.

4.3 Methodology

4.3.1 Keywords generation

Although knowledge components represented the answer, they still needed further processing in order to help in locating related paragraphs. Thus, we transformed each knowledge component into a set of keywords, so that the questions and the text could match against each other based on the keywords.

As shown in the previous example, the pattern of a knowledge component was [S, V, O] where V stood for a predicate, S stood for the subject of the predicate, and O stood for the object of the predicate. Both S and O were instances of some classes, and usually were written as the combination of the name of the class and a number. A class was usually a biology concept. For example, an instance of photosynthesis could be written as photosynthesis001, and an instance of oxygen could be written as oxygen001, so

(photosynthesis001, product, oxygen001) meant that photosynthesis could produce oxygen.

Ordinarily, the class name of the subject and the class name of the object formed the keyword set. However, in some cases, the class name itself provides little information about the event or object being described, so more information must be included in order to make the keyword set specific enough. Take the class “break” as the example. The class break was used to describe a process of a chemical component being split. For example,

(cellular_respiration6632,subevent,break72) (1)

(break72, object, glucose1224). (2)

(break72, result, carbon_dioxide28589). (3)

Keywords were required to be generated for relation 1, and relation 1 meant that the process described by *break72* was a sub-process of an instance of cellular respiration. Relations 2 and 3 further described *break72*, meant that an instance of glucose was broken into pieces including carbon dioxide. To transform these relations into keywords, instead of directly putting class name “break” into the keyword set, we put both glucose and carbon dioxide in the set of keywords for Relation 1.

4.3.2 Task reformation

At this point, each question was associated with several knowledge components, and each knowledge component was associated with a set of keywords. The next task could be further described in the following: There were n machine-generated questions, $Q = \{q_1, q_2, \dots, q_i, \dots, q_n\}$, each of which was associated to a number of knowledge components $KC = \{kc_1, kc_2, \dots, kc_i, \dots, kc_l\}$. Each of the knowledge components could be represented

as a set of keywords $W = \{w_1, w_2, \dots, w_i, \dots, w_k\}$. There were m paragraphs that roughly described the same content as the answers to the questions, $D = \{d_1, d_2, \dots, d_i, \dots, d_m\}$. The goal was to build connections between the questions and the paragraphs. Each connection could be denoted as (q_i, d_j) , which meant question i connected to paragraph j . The output of the algorithm was a set of (q_i, d_j) . The hierarchical relationships among question, knowledge component and keyword were described as figure 17.

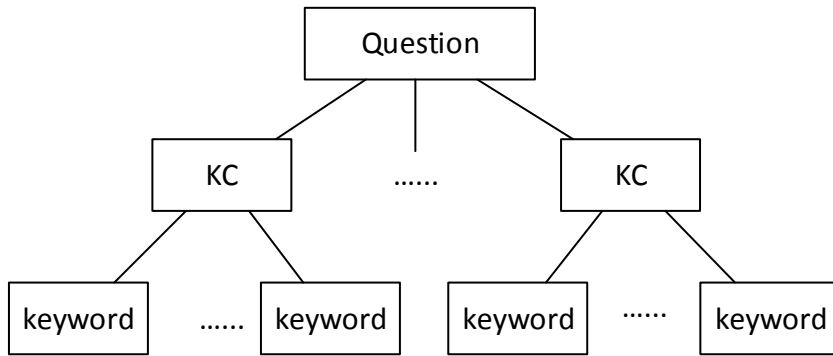


Figure 17 Relationships among questions, knowledge components and keywords.

Keywords were the features of questions we used to build the connections. Exact matching the keywords against the paragraphs would probably lead to omitting important connections, so some kind of more general matching was needed. In information retrieval, this is usually done by matching words with similar meaning (Cao, Cong, Cui, Jensen, & Zhang, 2009; Gao & Nie, 2012), so we used the same strategy. Our keyword generation process ignored the predicates in the relations and only put the subject class names and the object class names into the keyword set, which guaranteed that all the keywords were nouns. So we simply implemented fuzzy matching on noun phrases by referring to the synonyms in WordNet (Fellbaum, 1998). However, WordNet mainly encoded everyday knowledge instead of the terms in biology. So 9 sets of synonyms were manually added

(see Appendix C). Matching any synonym of a keyword was treated as matching the keyword itself.

The entire matching process was divided into two steps: (1) All the paragraphs in the text were preprocessed to remove punctuations and stop words and to replace all the synonyms of the keywords with the keywords themselves. (2) For each question, Okapi BM25 was used to calculate the relatedness of the paragraphs to the question, and the related paragraphs were ranked in terms of the relatedness scores. The next two sections described the two steps separately.

4.3.3 Preprocessing

All the punctuation in the text was removed first. Because our text processing was not case sensitive, all the terms were transformed to lower case as well. The original text was mixed with some figures, so there were some phrases like “as figure 1” embedded in the text. These phrases were also removed during preprocessing.

As mentioned before, when a synonym of a knowledge component’s keyword was recognized, this synonyms would be replaced with the original keyword. To implement this, a synonyms set was built for each keyword appeared in the question set. For example,

Light reaction -> {light reaction, light dependent reaction}

As mentioned earlier, both WordNet and manual input were used to form the entire synonyms set. Whenever a phrase in the list of synonyms sets was detected, the phrase was replaced with the keyword.

4.3.4 Question mapping

Because keywords were associated to knowledge components instead of the questions, for each question, we first found the related paragraphs to each of its knowledge components, then combined these paragraphs together as the final related paragraphs to the question.

We used BM25 to calculate the relatedness of a paragraph to a knowledge component. BM25 was originally used to quantify the relatedness of a document to a given query in information retrieval. Given a query, the score of the document to the given query is:

$$score(d, query) = \sum_{i=1}^n IDF(w_i) \cdot \frac{f(w_i, d) \cdot (k_1 + 1)}{f(w_i, d) + k_1 \cdot (1 - b + b \cdot \frac{|d|}{avgdl})} \quad \dots\dots (3)$$

where d is the document, and $query$ represents the given query. w_i is the i th keyword in the query. $f(w_i, d)$ represents the frequency of the i th keyword in the document. $f(w_i, d)$ is the number of occurrence of w_i in d , $|d|$ is the length of paragraph d and $avgdl$ is the average length of all the documents. k_1 and b are free parameters for tuning performance. We set $k_1=0.6$ and $b=0.75$ as earlier studies suggested. $IDF(w_i)$ is inverse document frequency of w_i , which is calculated as:

$$IDF(w_i) = \log \frac{N - n(w_i) + 0.5}{n(w_i) + 0.5} \quad \dots\dots (4)$$

where N is the total number of documents and $n(w_i)$ is the number of documents that contain the keyword w_i . So the more popular the word is, the lower $IDF(w_i)$ is.

Each paragraph in the text was treated as a document in the original BM25 calculation. The knowledge components associated with keywords became the queries. So the

relatedness score, which is $score(d, kc)$, of a paragraph to a knowledge component is calculated according to equation (5)

$$score(d, kc) = \sum_{i=1}^n IDF(w_i) \cdot \frac{f(w_i, d) \cdot (k_1 + 1)}{f(w_i, d) + k_1 \cdot (1 - b + b \cdot \frac{|d|}{avgdl})} \quad \dots\dots (5)$$

where d represents a paragraph instead of a document, kc is the knowledge component and w_i is the i th keyword associated with the knowledge component. Other notations are exactly the same those in original BM25.

To calculate the relatedness score for each pair of paragraph and knowledge component, we first iterate through all the knowledge components' keywords, calculating $IDF(w_i)$ and $f(w_i, d)$ for each paragraph. The time complexity of this process was $O(Nw * |d|)$, where Nw is the total number of keywords appeared in all the knowledge components, and $|d|$ is the total number of paragraphs. All the calculated values were stored in a hash table so that they could be easily retrieved afterwards. Secondly, we iterated through all the knowledge components and calculated the paragraph's score for the knowledge component with the corresponding $IDF(w_i)$ and $f(w_i, D)$.

Now that the relatedness of paragraphs to knowledge components were calculated, the relatedness of a paragraph to a question could be calculated as the sum of the paragraph's relatedness to all the knowledge components of the question. However, this was not the final calculation we adopted, because we wanted to tackle a special case.

Suppose that given a question, one paragraph was highly related to one of its knowledge components but not related to any other of its knowledge components, and another paragraph was related to all its knowledge components albeit not strongly. Which

paragraph should be ranked higher? We preferred to rank the latter paragraph higher because the latter paragraph probably covered all the key points of the answer to the question, while the first paragraph was probably a short one concentrating on just part of the answer. Reading the first paragraph would not be enough for a student to fully understand the answer. In other words, we think it was always more important for a paragraph to cover many knowledge components rather than concentrating on just one knowledge component. In our case, the relatedness scores of the knowledge component (KC) -paragraph pairs were between 0 and 5, where 0 means that the paragraph is unrelated to the KC. To ensure that the paragraphs covering more knowledge components always rank higher, every time the relatedness score of a KC-paragraph pair is added into the relatedness score of a question-paragraph pair, if the relatedness score of KC-paragraph is strictly greater than 0, an extra 10 is added to the relatedness score of the question-paragraph pair. Thus, the score is essentially counting the number of question-related KCs that the paragraph covers, and the relatedness scores of KC-paragraph pairs make a difference only when two paragraphs cover the same number of question-related KCs. The relatedness score of a question-paragraph pair was calculated as equation (6)

$$score(d, q) = \sum_{i=1}^n 10 \cdot g_i + score(d, kc_i) \quad \dots\dots (6)$$

$kc_i \in \{All\ the\ KCs\ associated\ to\ the\ current\ question\ q\}$

$$g_i = \begin{cases} 0 & \text{if } score(d, kc_i) = 0 \\ 1 & \text{if } score(d, kc_i) > 0 \end{cases}$$

where d stands for a paragraph, q stands for a question, and $score(d, kc_i)$ is relatedness score of one of the question's associated knowledge components to the paragraph.

Recall that the main purpose of mapping questions to their related paragraphs was to provide reading materials as a part of the feedback in addition to the correct answer. Students could be overwhelmed if there were too many paragraphs attached to one single question. So three related paragraphs at most were permitted per question. If more than three paragraphs were related to the question (i.e., $\text{score}(d,q) > 0$), then the three paragraphs with the top scores were selected. If there were less than three related paragraphs whose relatedness score to the question were bigger than 0, then they were selected.

4.4 Evaluation

The output of the machine-generated question-paragraph mapping was a set of pairs, represented as (d, q) , where d is a paragraph and q is a question. One way to evaluate such a mapping is to have domain experts go through all the pairs one by one and annotate each pair as good or bad. It is simple and straightforward, but the problem is that domain experts' opinions could be biased by the machine-generated pairs. Moreover, a domain expert could not indicate that he/she think there should be a connection pair that is absent from the machine-generated pairs. Therefore, the experiment was designed so that question-paragraph connections could be built by domain experts completely independent of machine-generated ones.

Two college students were recruited to generate the question-paragraph connections individually, and then they came together to solve any conflicts they had between their individual judgments. The two students were considered as domain experts because both of them majored in biology and had good grades in their course work. Both of them were simply told to connect the related paragraphs to each question.

54 questions and 53 paragraphs were given to the two students. There was no restriction on the number of paragraphs that could be connected to each question. Students could connect as many as they wanted as long as they thought the paragraphs were related. They took the assignment home and reported that it took each of them more than 10 hours in total to finish mapping all the 54 questions. After both of them finished their individual work, they were asked to come to our lab and resolved all the conflicts, which took about 3 hours. Each of them got \$200 as compensation upon completing the entire experiment.

4.4.1 Measures

The experiment generated 4 groups of question-paragraph connections in total: the individual work of student 1 (S1), the individual work of student 2 (S2), their mutually-agreed mapping after discussion, and the machine-generated mapping. Kappa was usually used to test inter-rater agreement, but it was not an appropriate measure here because there were 2^{53} , 53 was the number of paragraphs, combinations of connections for each question, which lead to 2^{53} categories. Instead of kappa, precision and recall were used to measure how the 4 mappings agreed with each other. Recall that an item in a group was a question-paragraph connection, represented as (d, q) . Given two sets of questions-paragraph connections, *group1* and *group2*, with *group2* treated as the gold standard, the precision of the mapping for one question q_i was calculated as equation (7), the recall was calculated as equation (8), and F-score was calculated as equation (9).

$$\begin{aligned}
 & \textit{precision}(q_i) \\
 &= \frac{|\{(d, q_i) \in \{\textit{group1 connections}\}\} \cap \{(d, q_i) \in \{\textit{group2 connections}\}\}|}{|\{(d, q_i) \in \{\textit{group1 connections}\}\}|} \dots (7)
 \end{aligned}$$

$recall(q_i)$

$$= \frac{|\{(d, q_i) \in \{group1\ connections\}\} \cap |\{(d, q_i) \in \{group2\ connections\}\}| \dots (8)}{|\{(d, q_i) \in \{group2\ connections\}\}|}$$

$$F(q_i) = \frac{2 \cdot precision(q_i) \cdot recall(q_i)}{precision(q_i) + recall(q_i)} \dots (9)$$

Now that the precision and recall for each question were calculated, the precision, recall and F-score for *group1* were the average of those values over the questions.

4.4.2 Results

Among the 4 groups of connections, the mapping that the two judges agreed upon after discussion (let's call it the *joint* mapping) was treated as the gold-standard mapping. To explore how reliable the gold-standard was, we calculated the average precision, recall and F-score of S1 and S2 as Table 16 showed.

Table 16. Result of question mapping (N=54)

	All questions (N=54)		
	Precision	Recall	F measure
S1	0.68	0.56	0.62
S2	0.38	0.66	0.48

The precision and recall were much lower than we expected, which meant that the judges disagreed with the joint mapping. However, considering that the chance of getting both

precision and recall being 1.0 for one question was $\frac{1}{2^{54}}$, the result was fairly acceptable. On average, there were 1.35 questions mapped to each question in the joint mapping. S1 had a high precision with a low recall because S1 always tended to map a small number of paragraphs (1.09 on average) to each question. On other side, S2 always tended to map much more paragraphs (3.06 on average) to each question, which ended up with low precision and high recall. In a result, the two individuals made many disagreements. It seemed easier for them to make agreement on finding the related paragraphs to the questions whose answers were emphasized only once in the text, such as “Could you describe the three sub processes involved in the calvin cycle in photosynthesis?”. However, it was more difficult for them to make agreement for the questions whose answers were mentioned several times in the text, even though the questions themselves were fairly simple, such as “What is the role of chloroplast in photosynthesis?”.

The mapping algorithm did not successfully generate mappings for all the questions. There were 33 questions where it got at least 1 paragraph mapped, out of the 54 questions. It was probably because the answers to those unmatched questions were buried in the paragraph, and the keyword-based method failed to find the information.

Nonetheless, if the algorithm can find related paragraphs for some of the questions and leave the rest for domain experts to do, it still can save domain expert’s time. Alternatively, we could only use the questions with mapped paragraph to assist students learning. Therefore the rest of this section was focus on reporting the data about the 33 questions.

Table 17 showed the precision, recall, and F-score of S1, S2 and the algorithm based on the 33 questions.

Table 17 Result of mapping on subset (N=33) of questions

	Precision	Recall	F measure
S1	0.60	0.55	0.57
S2	0.29	0.64	0.40
Algorithm	0.28	0.52	0.36

It looks like the algorithm did a fairly good job because its F-score is close to that of S2. However, we were still bothered with the relatively low precision and recall of the individual judges, so we wanted to understand more about how they made the agreement. Then we calculated S1's precision and recall on S2's mapping, and S2's precision and recall on S1's mapping, showed as Table 18. The result suggested that the two students disagree substantially before the discussion. The joint mapping was a result of their compromising with each other.

Table 18 Result of question mapping with different gold standards

	S1's mapping as gold standard (N=33)			S2's mapping as gold standard (N=33)		
	Precision	Recall	F measure	Precision	Recall	F measure
P1	1	1	1	0.45	0.14	0.21
P2	0.14	0.45	0.21	1	1	1
Machine	0.17	0.33	0.22	0.27	0.20	0.23

The two students also reported why they had many conflicts. The most important reason they stated was that it was always difficult to select the exact paragraph from several adjacent paragraphs, and it was too much to include them all. To explore how much this factor affected the precision and recall, paragraphs were merged in terms of the index in

the textbook. In a result, 54 paragraphs were aggregated into 15 segments. All the precisions and recalls were calculated again based on the 15 segments like table 19 showed

Table 19 Result of question mapping with aggregation by the index in textbook on (N=33) questions			
	Precision	Recall	F measure
S1	0.71	0.62	0.66
S2	0.57	0.80	0.67
Algorithm	0.63	0.73	0.68

F-scores of all three groups, especially S2 and algorithm, increased significantly. However, the improvement could be due to two factors: (1) although students disagreed with the gold-standard, they chose the paragraphs that were close to the one in gold standard; (2) aggregation of paragraphs made the possible number of combinations of connections for each question reduced from 2^{54} to 2^{15} .

We hypothesized that the improvement was mainly because of the first factor. To test this hypothesis, instead of aggregating paragraphs by the index in the textbook, we randomly created 15 segments, each of which at least contained one paragraph. There was no maximum number of paragraphs that could be included in one segment. Every paragraph had an equal probability to be assigned to any of the 15 segments. Ideally, we should go through all the possible assignments (about 15^{54}) of paragraphs to 15 segments, and calculate precision/recall for each assignment. But there were too many legal assignments. So we estimated precision and recall by sampling. We first repeated 1000 times of generating 15 segments by random assignment, and precision/recall was calculated at each iteration. Then the first step was repeated 100 times to calculate mean

average precision and mean average recall, which were treated as the estimated precision/recall of random assignment. The result was showed in Table 20. Compared to Table 18, the F-score was increased a little bit after random aggregation, but the aggregation by the textbook index made F-score increase much more significantly. Therefore, we believed that most of the disagreement was due to the difficulty of deciding the exact paragraph from several adjacent ones.

Table 20 Result of question mapping with random aggregation for auto-matched questions (N=33)			
	Precision	Recall	F measure
S1	0.63	0.58	0.60
S2	0.33	0.69	0.45
Algorithm	0.33	0.58	0.42

4.5 Discussion

The big disagreement between the two individuals was unexpected before the experiment. They finally came to the agreement after a serious discussion. The level of disagreement was reasonable given the number of possible combinations of question-paragraph connections for each question. A further exploration by aggregating adjacent paragraphs according to the textbook index showed that, most of the time, the two students disagreed with each other because they had a hard time in selecting a mutually agreeable paragraph from several adjacent ones. Although they made the final agreement during the discussion, this suggests that adjacent paragraphs were coherent with each other, so it might be better to present them all to a learner instead of always reducing the presented

paragraphs to the three best. On the other hand, providing too much reading after each question would overwhelm a learner.

Another way to use the related questions could be providing the related reading after a learner answered all the questions or a considerable number of questions. The learner should not be overwhelmed by multiple-paragraph reading materials in this case.

Regarding the reliability of the joint mapping, the main source of making the agreed mapping differ from the individual mappings was the agents picking other close paragraphs, and the two participants discussed seriously to resolve all their conflicts. It should be trustful.

While comparing the machine-generated mapping to the gold standard, it turned out that the performance of the machine-generated one was very close to that of S2. A more promising finding was that the machine-generated mapping agreed more with the joint mapping than either student's individual mapping. This suggests that their discussion moves the mapping towards the machine-generated one without knowing what it looked like. Therefore, the machine-generated mapping could be a valuable aid to human judges' discussion. Indeed, one single individual might be able to use the machine-generated mapping to refine his own mapping and made the mapping closer to the gold standard without discussing with another person.

4.6 Related works

4.6.1 Document retrieval

Document retrieval has always been an active research area in information retrieval. Given a query. Its goal is to find all documents related to a given query. More specifically,

researchers seem to be more interested in retrieving similar questions to a given question because of the popularity of question answering communities. The classical model to solve the retrieval problem is BM25 (Robertson, Walker, Jones, Hancock-Beaulieu, & Gatford, 1995) where words are weighted with tf-idf. But this model only works for exact match, and fails to detect the words with similar meanings (e.g. *good & well*). Therefore, Xue, Jeon, & Croft (2008) further used IBM translation model 1 to identify the words with similar meanings. Other works (Cao et al., 2009; Gao & Nie, 2012) in questions retrieval are also dedicated in solving the same problem. In intelligent tutoring system field, Autotutor (Graesser et al., 2004) represented every word as a vector learned by Latent Semantic Analysis (LSA), so that words with similar meaning had a smaller cosine distance.

4.6.2 Question answering

The most well-known question answering (QA) system is probably IBM's Watson (Ferrucci et al., 2010), which won the American TV quiz show Jeopardy against humans in 2010. Watson's question answering process could be divided into three big steps: question analysis, hypothesis generation, and hypothesis evaluation. Other research on question answering (Baral et al., 2012; Fader, Zettlemoyer, & Etzioni, 2013; Puente, Sobrino, & Olivas, 2009) usually focuses on one or two steps, assuming perfect inputs and outputs of other steps. Despite any tricky reasoning process a QA system might have, the goal of QA systems was to generate precise and specific answers. Question-paragraph mapping was similar to hypothesis generation and evaluation in QA, but the goal was not to generate the exact answer.

4.7 Conclusion

We developed an algorithm to automatically map the paragraphs in the textbook to the machine-generated biology questions. It was able to do so for 33 of the 54 target questions. To test the quality of the auto-generated mapping, two college biology majors manually created the mapping for the same set of questions and paragraphs. It turned out that the mapping task was a challenge even for the human participants. Nonetheless, the algorithm managed to produce a mapping that compared favorably to one of the participants. Precision and recall were increased significantly after the paragraphs were aggregated in terms of the index in the textbook, which suggests that when there were disagreements, it was caused by agents picking paragraphs that were adjacent or at least close to each other. Thus, we believe that our algorithm is good enough to be useful.

The mapping technique could be used to generate further reading as a part of feedback for each question. Alternatively, the suggested reading could also be provided only after a student finished a considerable number of questions.

CHAPTER 5

EVALUATION OF AUTO-GENERATED DISTRACTORS IN MULTIPLE CHOICE QUESTIONS

5.1 Introduction

5.1.1 Background of multiple choice questions

A multiple choice question is consisted of three components: stem, answer, and distractors. For example:

The light reactions of photosynthesis occur in the _____

- a. Cytochromes*
- b. Thylakoid membranes*
- c. Reaction centers*
- d. Stroma*
- e. Antenna complexes*

“The light reaction of photosynthesis occur in the _____” is the stem of the question, and the five choices below are the alternatives (sometimes called “foils”), where alternative b is the correct answer and all the rest are distractors.

This type of question is widely used because it is easy to score. However, generating good multiple choice questions is not easy even for human being. Instructors need to put considerable amount of time in making plausible distractors. A distractor that helps student learn is even harder to invent. So we want to develop an algorithm to at least save instructor’s time in finding distractors for shallow multiple choice questions.

Many guidelines have been developed for generating good multiple choice questions. They describe features of good stems and good alternatives (Al-Rukban, 2006; McMillan, Hellsten, & Klinger, 2007; McMillan & Lawson, 2001). In this chapter, we focus on

generating good distractors with the assumption that stems have been generated already, so we summarized 5 features of good distractors that appeared frequently in these guidelines:

1. All alternatives should be plausible
2. Alternatives should be homogeneous in content
3. Alternatives should be mutually exclusive
4. Alternatives should be free from clues about which response is correct
5. Alternatives should be placed in some logical order

We interpreted the rules as: a distractor in a multiple choice question should be similar to the correct answer but definitely a wrong response to the stem. This interpretation was translated into constraints to find qualified distractors in a semantic network, which was described in the next section.

5.1.2 Semantic network

A semantic network is a network that describes the relations of a set of concepts. It records domain knowledge. We obtained the semantic network about photosynthesis from Baral & Liang (2012), and used it to generate open response questions in our previous work. For example, representing “photosynthesis produces sugar and oxygen” in the semantic network needs the following two rules:

(photosynthesis, result, sugar)

(photosynthesis, result, oxygen)

where, result is a predefined predicate in the semantic network, photosynthesis is the subject of the predicate, and sugar/oxygen is the object of the predicate.

Besides relationships between concepts, the semantic network also defines the ontology of these concepts. For example, photosynthesis is a subclass of process and sugar is a subclass of entity. This ontology information is also described with triples.

5.2 Related works

We are not the first one to generate multiple choice questions from a domain knowledge base. Alsubait, Parsia, & Sattler (2014) generated multiple choice questions from OWL ontologies for evaluating everyday knowledge. Their domain knowledge base also described some simple relationship beyond ontology (e.g. `marriedTo(Mark, Sara)`). However, their generation schema will lead to over-generation in our domain because relations in our knowledge base are more complicated. There are several other works that also generated multiple choice questions from ontologies, especially for English learning (Brown, Frishkoff, & Eskenazi, 2005; Lee & Seneff, 2007; Y.-C. Lin, Sung, & Chen, 2007). One thing that makes our work different from previous work was that we generated multiple choice questions for a domain that had more complicated relations.

In terms of evaluating distractor or question quality, one common way widely used in previous works was to have human judges score the quality of each question or distractor (Karamanis, Ha, & Mitkov, 2006; Lee & Seneff, 2007; Papasalouros, Kanaris, & Kotis, 2008). A generation algorithm was thought to be good if a majority of the generated multiple choice questions had good scores from the judges. The issue with this type of evaluation is the expertise of the judges. After all, not every instructor is an expert in generating good distractors, for otherwise there would not be so many guidelines for teaching people how to make good multiple choice questions. Mitkov, Ha, Varga, & Rello (2009) used Item Response Theory (IRT) to evaluate question qualities, which we believed

is a more objective and convincing approach. Another issue with previous work was that machine-generated distractors were seldom compared to human-generated distractors. Huang & Mostow (2015) was one exception. But their measures were special as well. Instead of directly evaluating distractor quality, they tested whether their algorithm could generate desired types of distractors. In this study, we used IRT parameters to compare human-generated and machine-generated distractors.

5.3 Multiple choice question generation

Open response questions were generated in our previous work. This is another reason we focused on generating and evaluating distractors in this chapter. There have been many different methods developed for finding distractors of a question based upon the answer to the question. The most popular way was to calculate semantic similarity between a candidate distractor and the answer according to WordNet (Fellbaum, 1998) or some other well-known ontologies. If the similarity was bigger than some threshold, the candidate distractor would become one of the final distractors. Pho, Ligozat, & Grau (2015) recently used this method to generate distractors for multiple choice questions in English language learning. Another widely used method was to write rules to extract distractors from a domain specific ontology (Al-Yahya, 2011; Papasalouros et al., 2008). We adopted the second methodology because we had a domain specific knowledge base available, and stems as well as answers had been generated from the knowledge base already.

In our previous work, four question generation schemas were developed. The four schemas were for *input/output* questions, for *where* questions, for *what* questions, and for *connection* questions. Multiple choice question generation guidelines suggest that a good distractor should be a plausible answer to the question for an inexperienced person in the

domain but also be a clearly wrong answer for a domain expert. This suggests using the schemas that generated the questions as schemas for generating the distractors as well. The next section described the four distractor-generation schemas respectively.

5.3.1 Distractor generation schemas

Input/output questions

Input/output questions ask for the raw materials or the products of a process. There were two intuitions for generating distractors for this type of question:

1. The raw materials of a process can be used as the distractors for a question asking for the products of the process, and vice versa.
2. The intermediate raw materials or products of a process can be used as distractors for a question asking for either the initial raw materials or the final products of the process.

Figure 18 depicted the scenario of the first intuition, where entity1 could be treated as the distractors of entity2, and vice versa. In this case, distractors and the correct answer were connected via one node and two specified predicates. Figure 18 depicted the second intuition: Given a process, entity2 is one of the products of the sub process of the given process, so entity2 could be treated as the distractors of entities1, which were the final products of the given process. In this case, the distractors and the correct answer were connected with two intermediate nodes and three specified predicates. Obviously, removing some of the constraints would lead to an increased number of distractors.

We were also interested in finding out the quality of new distractors when constraints were relaxed. Therefore, we also generated some distractors by finding out the entities that were connected to the correct answer with one intermediate node with the predicates not being specified. According to our informal evaluation, although loose constraints did lead

to more distractors, the distractor qualities were also clearly decreased. So the distractors generated after constraints relaxation were not evaluated in the experiment.

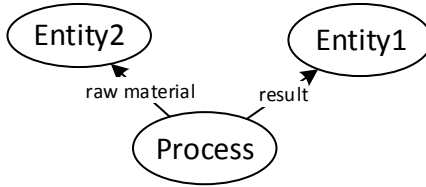


Figure 18. Entity1 can be a distractor, if Entity2 is the correct answer.

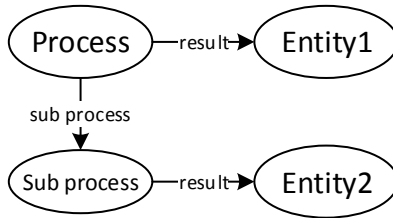


Figure 19. Entity2 can be a distractor, if Entity1 is the correct answer.

“Where” questions

This type of questions asks for the location of a process. Because the location had to be an entity, a good distractor should also be an entity and somehow related to the correct answer. We defined a good distractor of this type of questions as the entity that was another part of the same structure as the correct answer. The relation between the correct answer and a distractor was depicted in Figure 20, where entity1 represented the correct answer, entity2 represented a distractor, and entity3 was the structure which both of them were part of. We also tried to relax the constraints in this case in order to generate more distractors. When the constraints were relaxed, the predicate could be any one without being limited to “part_of”

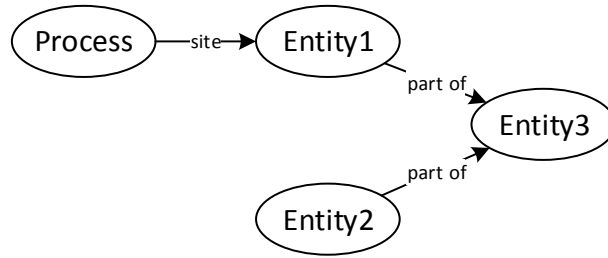


Figure 20 A distractor for where questions.

“What” questions

“What” questions ask what sub-processes a parent process should contain. A good distractor would be a sub-process contained by a related process to the parent process. Now the issue was how to find the related processes. Two processes were considered to be related to each other if they satisfied one of the three conditions below:

1. The two processes shared the same parent process.
2. The two processes shared at least one product
3. The two processes shared at least one raw material

The relations of two related processes are depicted in Figure 21, where process1 and process2 represent the two related processes. Figure 21a stands for condition 1, Figure 21b stands for condition 2, and Figure 21c stands for condition 3.

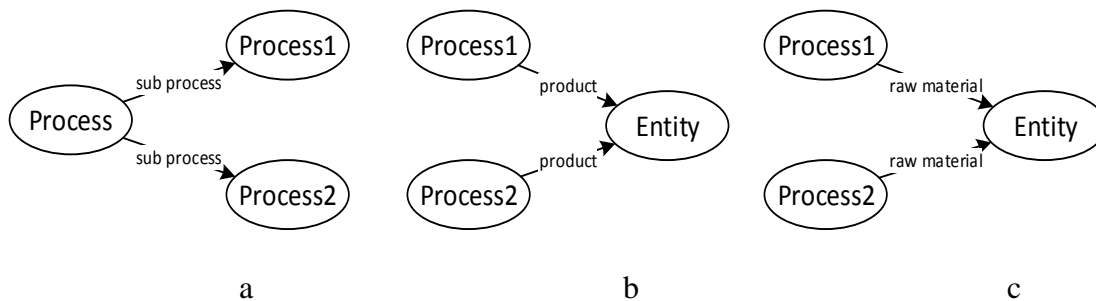


Figure 21 process1 and process2 are related processes

“Connection” questions

This type of question asks what products of one process were raw materials for another process. There were two types of entities that can be potentially used as distractors. One was the set of products of the first process that were not used as the inputs to the second process, the other was the set of raw materials of the latter process that were not able to be produced by the first process. The relations were depicted as Figure 22, where entity1 represented the first type of distractors and entity2 represented the second type of distractors.

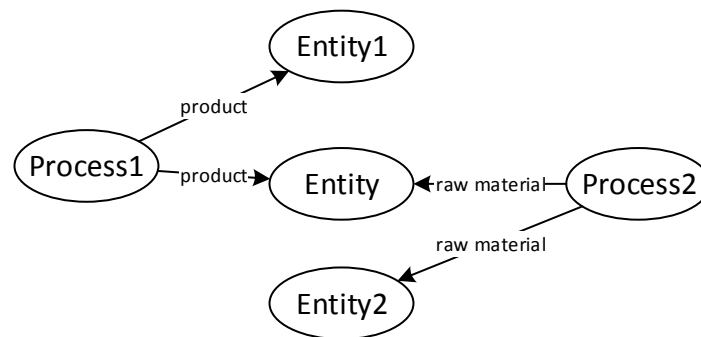


Figure 22 distractors for connection questions

5.3.2 Post-possessing

Not all the distractors generated from the 4 schemas above could directly become the final distractors in multiple choice questions. To distinguish the two types of distractors, the former distractors were called as raw distractors, and the latter distractors were called as final distractors. This section mainly described how raw distractors were further processed to be final distractors.

In terms of stems, the original machine-generated open response questions could potentially be processed to be the stems in the corresponding multiple choice questions.

However, we wanted to evaluate how machine-generated distractors compared to human-generated ones in this chapter. Moreover, we wanted the human-generated distractors to be “naturally occurring” rather than generated by experts in our lab. Thus, we looked for human-generated multiple choice questions whose stems were essentially asking the same thing as the machine-generated open response questions, and replaced the original distractors in human-generated questions with machine-generated raw distractors. Thus, in this study, the stems were human-generated and the distractors were both human-generated and machine-generated.

What two energy carrying molecules are produced by the light of reaction of photosynthesis?

- a. NADP+ and ADP
- b. NADP+ and ATP
- c. NADPH and ATP correct answer
- d. ADP and ATP
- e. NADP+ and NADPH

Figure 23. Human-generated multiple choice question with complex alternatives

A distractor in a human-generated multiple choice question often does not contain just one concept. For instance, Figure 23 is an example of such a question. Each alternative of this question contains two concepts. Each of the two concepts corresponds to one machine-generated raw distractor. To avoid this essentially format-related factor in this study, we adopted the combination patterns of the original human-generated multiple choice questions. Thus, the alternatives in machine-generated questions were generated by replacing the phrases in human-generated alternatives. If a phrase in a human-generated distractor was a correct response to the question, that phrase would be kept without

replacement. When a phrase in a distractor was to be replaced, a raw distractor was randomly selected out of all the machine-generated raw distractors for that question. The question in Figure 24 was the corresponding machine-generated multiple choice question to the human-generated question in Figure 23. The final machine-generated distractors would be made by replacing NADP⁺ and ADP with sunlight, water, sugar or RuBP, as Figure 25 showed.

Original machine generated open response question:

What are the products of the light reaction in photosynthesis?

Stem: What two energy carrying molecules are produced by the light of reaction of photosynthesis?

Raw distractors: sunlight, water, sugar, RuBP

Figure 24. Pre-processed machine-generated multiple choice question

What two energy carrying molecules are produced by the light of reaction of photosynthesis?

- a. Sunlight and RuBP
- b. Water and ATP
- c. NADPH and ATP
- d. RuBP and ATP
- e. Sugar and NADPH

Figure 25 Final machine-generated multiple choice question

Sometimes, raw distractors could directly replace human-generated distractors without combination, as in the question of Figure 26. The raw distractors of the corresponding machine-generated question were: thylakoid, chloroplast membrane, ribosome, thylakoid space and DNA. To evaluate as many distractors as possible, each multiple choice question

was required to contain 5 alternatives (i.e. one correct answer and four distractors). Thus, the final machine-generated distractors were supposed to be directly built by randomly selecting 4 out of the 5 machine-generated raw distractors. However, the machine-generated raw distractors and the human-generated distractors shared one common distractor in this case. To make machine-generated distractors be as different from human-generated ones as possible, the common distractors were excluded before random selection except when machine-generated raw distractors were not enough.

The light-independent reactions of photosynthesis take place in the

- a. Stroma correct answer
- b. Thylakoids
- c. Nucleus
- d. Mitochondria
- e. Grana

Figure 26 Human-generated multiple choice question with simple alternatives

In a summary, each multiple choice question contained 5 alternatives, one of which was the correct answer except for two questions that had two correct answers. Although machine-generated distractors did overlap with human-generated distractors as in the example above, the distractors in machine-generated questions were selected in a way, so that the same distractors would not appear in both of the two types of questions. Sometimes, distinct distractors were not enough to form 5 alternatives, and the overlapping distractors were used to keep the number of alternatives at 5. When the machine-generated raw distractors needed to be combined to form final distractors, the combination patterns in the corresponding human-generated distractors were adopted.

5.4 Evaluation

Our algorithm generated 37 pre-processed multiple choice questions. Unfortunately, only 9 of them were found to correspond to human-generated multiple choice questions. So the evaluation was done based on the 9 questions.

Before comparing the qualities of the two types of distractors, we first wanted to explore how much machine-generated distractors and human-generated distractors agreed with each other. The agreement should be checked in the level of raw distractors. Each phase in human-generated distractors was treated as a human-generated raw distractor. Among the 9 pairs of questions, there were 44 machine-generated raw distractors and 42 human-generated raw distractors. The number of overlapped raw distractors between the two sets was 11. Therefore, 25% of the human-generated raw distractors were covered by machine, and 26.2% machine-generated raw distractors were covered by humans.

5.4.1 Measures

Good multiple choice questions should make the students who are knowledgeable in the teaching domain be able to answer them correctly and those who lack knowledge in the teaching domain should fail to correctly answer the questions. So discrimination of a question is probably the best measure for quantifying the quality of a question.

Rather than quality of a multiple choice *question*, we are more interested in quality of each *distractor*. Therefore, we also calculated the discrimination of distractors, which was called as *usefulness* by Mitkov et al. (2009). Question difficulty can somehow reflect question quality as well, and it was a measure that potentially helped us tell the difference between machine-generated distractors and human-generated distractors. So question difficulty was also calculated.

Either difficulty, discrimination or distractor usefulness can only be calculated once from one dataset. To explore the reliability of these measures, we used bootstrap, so that each sample of the original data could produce a group of measures. For each measure, the mean of values produced by all the samples was used as the final output.

Question difficulty

A question's difficulty reflected the percentage of students who were able to answer the question correct. Difficulty should be in [0,1]. The higher the value, the more difficult the question. In general, neither extremely hard questions nor extremely easy questions are good. But extreme questions might be useful in distinguishing some particular set of students. A question's difficulty was calculated as equation (1)

$$Dif(Q) = \frac{1 - N_c}{N} \dots\dots (1)$$

where N_c was the number of students who answered the question correct.

N was the total number of students

Question discrimination

Discrimination of a question was defined as the difference between the number of upper level students who answer the question correct and the number of lower level students who answer the question correct divided by the total number of students who correctly answer the question. The value would be positive if a majority of correct responses are from upper level students, otherwise it would be negative. In our study, we used 10 anchor questions, which kept the same across the two conditions, to define whether a student was upper level or lower level. Students whose anchor scores were between the 25th and 50th percentile classified as lower level students. Students whose anchor scores were greater than the 75th

percentile were classified as upper level students. The rest of the students were excluded from the analysis. The discrimination of question was calculated as equation (2).

$$Dis(Q) = \frac{N_U - N_L}{N_U + N_L} \dots\dots (2)$$

where N_U was the number of upper level students who got the question correct, N_L was the number of lower level students who got the question correct.

So the bigger the discrimination of a question was, the better the quality of the question was. Because both machine-generated questions and human-generated questions had the same stems and the same pattern for distractor combination, the difference on raw distractors would be the only source for different discriminations. Thus, this metric could be used to evaluate the overall quality of the two different types of distractors.

Usefulness of distractors

Usefulness of a distractor was defined slightly different from that made by Mitkov et al.(2009). A distractor was considered as useful if it attracted more lower-level students than upper-level students. In this case, the usefulness of the distractor was annotated as 1. When a distractor failed to attract any student or attracted equal number of students from different levels, the usefulness of the distractor would be equal to 0. When a distractor attracted more students from upper-level than students from lower-level, the usefulness of the distractor would be equal to -1. The calculation was like equation (3).

$$Use(A) = \begin{cases} -1, & \text{if } Na_L - Na_U < 0 \\ 0, & \text{if } Na_L - Na_U = 0 \\ 1, & \text{if } Na_L - Na_U > 0 \end{cases} \dots\dots (3)$$

where Na_U was the number of upper level students who selected the distractor, Na_L was the number of lower level students who selected the distractor.

Because bootstrap was used for measurement calculation, the usefulness of a distractor could be calculated based on each sample. If a distractor's usefulness was equal to 1 in at least half of the samples, the distractor was treated as a good distractor. If a distractor's sum of usefulness was smaller than 0, the distractor was treated as a bad distractor. So the quality of a distractor was determined in terms of equation (4).

$$Quality(A) = \begin{cases} good, & \text{if } |\{Use(A_n) = 1\}_{n=1, \dots, N}| > \frac{1}{2}N \\ bad, & \text{if } \sum_{n=1}^{n=N} Use(A_n) < 0 \\ undefined, & \text{otherwise} \end{cases} \dots \dots (4)$$

where N was the total number of iterations in bootstrap.

5.4.2 Experiment design

It is necessary to collect a lot of data for calculating all the three measures. So we used Amazon Mechanical Turk (MTurk) to recruit participants. The advantage of using this platform was that we can rapidly collect data from a variety of participants in a short time at low cost. The disadvantage was the difficulty of controlling quality of the collected data. There has been more and more researchers starting to use crowd sourcing platforms like MTurk to collect data, so analysis about how to better use MTurk also has become a hot topic. The thing we most cared about was how MTurk compared to traditional recruiting methods. Both Buhrmester, Kwang, & Gosling (2011) and Paolacci, Chandler, & Ipeirotis (2010) concluded that the respondents in MTurk might be slightly different from the respondents in a traditional subject pool, but the data obtained from MTurk were at least as reliable as those obtained by traditional methods. In addition, several techniques are available that can help in improving the quality of the data collected from online surveys,

such as instructional manipulation checks (Oppenheimer, Meyvis, & Davidenko, 2009) and Kapcha (Kapelner & Chandler, 2010). Actually, Hauser & Schwarz (2015) even believe that Amazon Turkers performed better than participants in traditional subject pool. Therefore, we decided to use MTurk to recruit participants for this study.

By referring to previous works (Kapelner & Chandler, 2010; Oppenheimer et al., 2009), we designed three mechanisms to ensure our collected data quality. The rest of the section described the mechanisms in detail. According to Mason & Watts (2010), data quality seems to be not affected by payments. We took an average payments of similar surveys available in MTurk. Each of participants got \$0.50 upon finishing the experiment.

First of all, qualified participants should at least have basic knowledge of photosynthesis. To rule out participants who had poor domain knowledge, two simple fill-in-the-blank questions were asked in the very beginning of the survey:

1. What are the raw materials of photosynthesis?
2. What does photosynthesis produce?

Participants were not expected to answer these two questions completely correctly. But they should at least point out one correct product and one correct raw material. If participants who failed to do so, they would be redirected to a page telling them their non-qualifications. Qualified participants would be redirected to the page showing the consent form and start the main survey after filling out the consent form.

Secondly, we set a delay for each question, so that participants were not able to go through all the questions rapidly. A study conducted by Kapelner & Chandler (2010) showed that adding a delay for printing each word in a question was more effective than simply adding a delay for the submit button of the question. The technique was called

“Kapcha”. They conducted four groups of experiments to study the effect of different setting to a survey of sentiment: the first group was the control group without any treatment. The second group had a reminder for each question to ask participants to respond seriously. The third group disabled the submit button for a while, so that participants were forced to slow down. The fourth group had the text fade in word by word. Trick questions were used to check the quality of the responses. Those questions could not be answered correctly without careful reading the instructions. Our study setting was very similar to sentiment surveys, so we believed that the Kapcha technique should make participants answer more seriously. Figure 27 showed how a question faded in to force participants to read carefully.

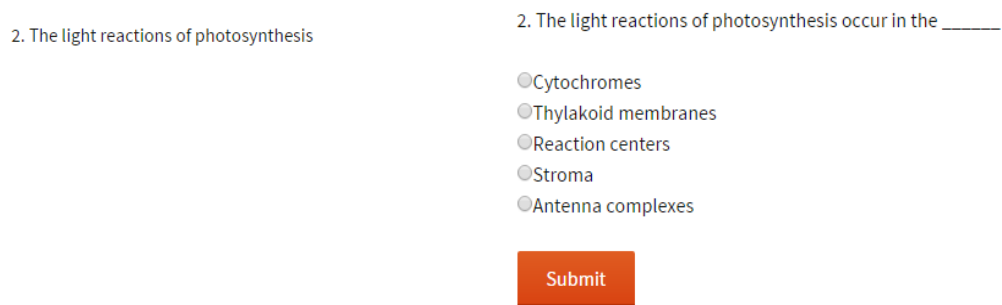


Figure 27 Force participants to read questions carefully

Our pilot study disclosed that some participants gave the same answer (i.e. the same order of alternatives) to all the multiple choice questions. They were apparently unserious participants. We assumed that a serious and consistent participant should always give the same response (i.e. the content of an alternative) to duplicated questions. So two more filters utilizing duplicated questions were to rule out these unserious responses. The difference of the two filters were: one filter set duplicated questions close to each other, called as close duplication filter, and the other filter set duplicated questions far from each other, called as far duplication filter.

In close duplication filter, a duplicated question of the n th question would appear right after the n th question or the $(n+1)$ th question with different order of alternatives. Participants were explicitly to be asked to select the same alternative in the new order. If a participant failed to do so, the participant would be asked to stop participating in the study. There were three pairs of close duplicated questions. All of them were located in the early phase of the survey. More specifically, question 4, question 6 and question 8 are duplicates of question 2, question 5 and question 7 respectively.

For the far duplication filter, there were at least 10 questions between two duplicated questions. But the duplicated questions appeared with the same order of alternatives. Participants were not told when a duplicated question appeared. The response to this type of duplicated questions would not affect whether a participant could continue the survey.

There were 2 conditions: control (human-generated) and experimental (machine-generated). Each condition contained 24 questions in total. The first 13 questions, 3 of them were anchor questions, which means that the whole questions—stem, alternatives and ordering—was the same in both conditions. We used these questions to decide whether a participant was upper-level or lower-level. The subsequent 9 questions were where the difference between conditions happened. The control group had 9 questions with human-generated distractors and the experimental group had 9 questions with machine-generated distractors. The last two questions were duplicated questions for the far duplication filter.

5.4.3 Results

200 participants were recruited through Amazon Mechanical Turk to take part in the experiment. It took about two weeks in total. 1 participant submitted on MTurk without actually doing the experiment, so he was excluded in the analysis. 100 out of the 199

participants were in control group, the remaining 99 were in the experimental group. Participants took 21.4 minutes on average to finish all the 24 questions.

Students from the two groups were first combined together, and classified as “upper”, “lower” or “undefined” based on their anchor scores. More specifically, students whose anchor scores were in [4,6] were lower achievement students. Students whose anchor scores were in [7,10] were upper achievement students. The reason of excluding the students whose scores were below 4 was that they had too little knowledge in the domain even though they passed the prescreen. In a result, there were 51 lower level and 41 upper level students in control group. There were 43 lower level and 40 upper level students in experimental group.

The three sections below reported the difference between the machine-generated distractors and the human-generated distractors in terms of three metrics: difficulty, discrimination and alternative usefulness. Recalled that bootstrap was used to estimate the mean value of a metric to each question. For example, the difficulty of a question was calculated as the mean of the difficulties among 10,000 iterations, and the standard deviation of the difficulty was the standard deviation of the 10,000 difficulties.

Question Difficulty

The average difficulty of the 9 questions in control group was 0.381 (SD=0.15), and the average difficulty of the 9 questions in experimental group was 0.376 (SD=0.097). Because question difficulties varied a lot within groups, to avoid the effect, we also compared the medians of the two groups. The median difficulty in control group was 0.364 (min=0.181, max=0.704), the median difficulty in experimental group was 0.366 (min=0.245, max=0.515). Two-tailed t-test was also run based on the 18 questions. It

turned out the difference was not significant ($p=0.94$, $d=-0.040$, $\text{power}=0.94$). The observed standard deviations and number of questions gave the experiment sufficient power to detect an effect size of 0.040 or more. Thus, if there is a difference between these two groups, it is smaller than 6%. All the comparisons suggested that the two groups of questions had the same difficulty in general.

Question Discrimination

The average discrimination of the questions in control group was 0.270 ($SD=0.20$), and the average discrimination in experimental group was 0.297 ($SD=0.23$). The medians of the groups were calculated as well. The median discrimination in control group was 0.325 ($\text{min}=-0.101$, $\text{max}=0.535$), and the median discrimination in experimental group was 0.340 ($\text{min}=-0.151$, $\text{max}=0.689$). It seemed that experimental group was a little bit better than control group. According to two-tailed t-test, there was no significant difference between the two groups ($p=0.79$, $d=0.13$, $\text{power}=0.80$). It suggested that the two groups of questions had the same discrimination in general.

Usefulness of distractors

Usefulness of the alternatives basically reflected the same information as discrimination of the questions, but in more detail. Out of the 34 distractors, there were 25 good distractor and 8 bad distractors in control group. In contrast, there were 23 good distractors and 7 bad distractors in experimental group. According to Chi-square test, $\chi^2=0.0072$, $p=0.93$, $\text{power}=0.93$, the two groups were very close to each other.

Recall that bootstrap was used to calculate the measure, and 10,000 samples were generated from the original data. Whether a distractor was good or bad depends on the

summary of its performance on the 10,000 samples. Table 21 shows the summary of performance of three typical distractors. Distractors like the distractor 1 in Table 21 had very unstable usefulness, which changed upon samples. Although distractor 1 was classified as a bad distractor according to our criteria, it actually worked well in about one third of the 10,000 samples. The distractor 2 in Table 21 appeared to be useful in almost all the samples, so as to be trusted as a good distractor. The distractor 3 in the table did not appear to be useful in any of the samples, so could be trusted as a bad distractor.

Table 21 summary of usefulness of two example distractors

	Negative	Neutral	Positive
Distractor 1	5134	1770	3096
Distractor 2	3	4	9993
Distractor 3	6376	3624	0

The summary of the results was shown in Table 21.

Table 22 Comparison of machine-generated distractors to human-generated distractors

		Human-generated distractors	Machine-generated distractors
Question Difficulty	Mean	0.381	0.376
	Median	0.364	0.366
Question Discrimination	Mean	0.270	0.297
	Median	0.325	0.340
Distractors Usefulness	Good/Bad	25/8	23/7

5.5 Discussion

Our entire user study was conducted through Amazon Mechanical Turk, which had been widely used in many areas for data collection, but rarely been used in evaluating educational systems. The biggest advantage of using Amazon Mechanical Turk was its

speed of data collection. We ran 200 participants within two weeks. The downside was the need of dealing with unqualified participants. It was necessary to have a prescreening test and gaming detectors because not every Turker would read instructions carefully and treat studies seriously, even only the Turkers with good tracks (historical acceptance rate >95%) were allowed to participate. Due to the length of our study and its required number of participants, Amazon Mechanical Turk probably was the best choice. It was probably impossible to finish the experiment within 2 months if participants were recruited from campus because there would be too much time required in advertisement and paper work preparation.

In terms of the results, all of the three measures suggested that the questions with machine-generated distractors were equivalent to the questions with human-generated distractors. It is good news for the researchers in the field of educational question generation. However, both of stems and distractors in the multiple choice questions were relatively shallow. Therefore, machine-generated multiple choice questions could not completely make instructors free of generating questions, but could save instructors time in fabricating shallow questions.

Besides distractors, stems in multiple choice question and combination patterns of raw distractors may affect question quality as well. These factors were excluded in our study. Further study should be aware of this assumption.

The results also suggested that the usefulness of some distractors varied a lot on different samples generated by bootstrap. It could be simply because the number of participants was not big enough. It also could be because the same distractors worked differently upon different students. It was well known that questions could be selected based upon a

student's competence (Conejo et al., 2004; Gemma Corbalan, Kester, & Van Merriënboer, 2006). Perhaps distractors can also be selected based on a student's competence. This might be done in a principled fashion when distractors are generated from an algorithm and annotated with the knowledge components. This would be an advantage of machine-generated distractors over human-generated distractors.

5.6 Conclusion

Amazon Mechanical Turk is an appropriate platform for recruiting participants for a short study such as ours that requires a large number of participants. With prescreening questions and gaming detectors properly set, the data appears to be sufficiently high quality in that the participants were taking the task seriously and had sufficient biology background to represent biology students in general. All the measures suggested that there was no difference between machine-generated distractors and human-generated distractors. The unstable performance on some distractors implied that distractors could also be adaptively selected based upon a student's competence.

CHAPTER 6

CONCLUSION

Studies in the dissertation suggest that despite the fact that the machine-generated questions mostly covered the shallow human-generated questions, differences in the two types of questions were small, especially in syntactic features such as fluency and ambiguity. In terms of the pedagogical values, the first study showed that the machine-generated questions had a close performance to the professional authored questions, and the second study showed that students made a significant improvement in their test scores by learning with the machine-generated questions no matter whether the questions were adaptively selected or not.

Therefore, it can be expected that instructors are able to save their time by using machine-generated shallow questions in a near future. These shallow questions can be used in many different situations, e.g., as the clicker questions in class, or as pre-assessment questions before class, or as the check questions in the middle of a video presentation.

The biggest advantage of the machine-generated question from a semantic network is that the questions are automatically annotated with their corresponding relations in the semantic network, and the relations can be easily transformed into knowledge components in the domain. This extra information about the questions has been shown to be very useful in the implementation of adaptive learning, which has become more and more popular due to the advent of big data technology. State of the art technologies in data mining let us collect and analyze students' data much more easily than before. However, analyzing data always needs to get domain experts involved. Sometimes, data experts and domain experts have hard time in figuring out how to cooperate with each other.

Specifically in educational data mining, domain experts need to be involved in identifying knowledge components, associating knowledge components to test items, evaluating student's response, and etc. The dissertation has shown that question generation from a semantic network is able to relieve domain experts from annotating questions and making question-paragraph mapping. Thus, this question generation technique may provide a better basis for adaptive learning.

The study on generating distractors for multiple choice questions suggests that the generation technique has potential for extending adaptive learning to a more fine-grained level, specifically, to the level of distractors. There is no previous work trying to adaptively select distractors probably because it requires too much involvement from domain experts. Building a Q-matrix that associates questions to knowledge components is already a burden for domain experts. Annotating all the distractors would significantly increase the workload. When distractors are auto-generated from a semantic network, there is no involvement is required of domain experts. Adaptive distractor selection is an issue ripe for future work.

The biggest practical problem with our generation technique is that building a semantic network itself needs domain experts. More importantly, for a given teaching domain, even domain experts may have different opinions in what should be included and how to represent it. Fortunately, people have been studying this issue for years and have started to establish widely accepted patterns for building knowledge bases. The most well-known platform is Wikipedia, where domain experts all over the world are building a single knowledge base together for many different subjects. Indeed, Wikipedia is hardly considered as a good example of semantic network because most of its content is written

in natural language. But it started a trend of knowledge sharing and has demonstrated how domain experts can cooperate with each other in making content under the same topic. This trend has formed a non-profit and free-content family called Wikimedia. Wikidata is another member of Wikimedia family. It is essentially a very large semantic network. Now there has been over 14 million items in Wikidata. In another aspect, making a semantic network is just like writing a textbook in a different format. The experience in making good textbooks can be borrowed to making good semantic networks. It also suggests that translation from natural language is another way to build a semantic network (Baral & Liang, 2012).

Studies in the dissertation also demonstrated how technologies in artificial intelligence, information retrieval, data mining, and crowd sourcing can be applied to improve tutoring systems. The semantic network used for question generation is a product in artificial intelligence, and it is the original initiator of the entire dissertation. Information retrieval techniques are used in building question-paragraph mapping. Data mining techniques empower data analysis on experiment results. Crowd sourcing methodology enabled collecting data during a very short time. Fortunately, intelligent tutoring system is an interdisciplinary field that has always accepted methods and results from many related research fields.

REFERENCES

- Al-Rukban, M. O. (2006). Guidelines for the construction of multiple choice questions tests. *Journal of family & community medicine*, 13(3), 125.
- Al-Yahya, M. (2011). *OntoQue: a question generation engine for educational assesment based on domain ontologies*. Paper presented at the Advanced Learning Technologies (ICALT), 2011 11th IEEE International Conference on.
- Aleven, V., Stahl, E., Schworm, S., Fischer, F., & Wallace, R. M. (2003). Help seeking and help design in interactive learning environments. *Review of Educational Research*, 73(2), 277-320.
- Ali, H., Chali, Y., & Hasan, S. A. (2010). *Automation of question generation from sentences*. Paper presented at the Proceedings of QG2010: The Third Workshop on Question Generation.
- Alsubait, T., Parsia, B., & Sattler, U. (2014). *Generating multiple choice questions from ontologies: lessons learnt*. Paper presented at the The 11th OWL: Experiences and Directions Workshop (OWLED2014).
- Anderson, J. R. (1982). The acquisition of cognitive skill. *Psychological Review*, 89, 369-406.
- Apté, C., Damerau, F., & Weiss, S. M. (1994). Automated learning of decision rules for text categorization. *ACM Transactions on Information Systems (TOIS)*, 12(3), 233-251.
- Arroyo, I., Mehranian, H., & Woolf, B. P. (2010). *Effort-based tutoring: An empirical approach to intelligent tutoring*. Paper presented at the The 3rd International Conference on Educational Data Mining.
- Baker, R. S., Corbett, A., Koedinger, K. R., Evenson, S., Roll, I., Wagner, A. Z., . . . Beck, J. (2006). Adapting to when students game an intelligent tutoring system *Intelligent Tutoring Systems* (pp. 392-401). Berlin: Springer.
- Banerjee, S., & Pedersen, T. (2002). An adapted Lesk algorithm for word sense disambiguation using WordNet *Computational linguistics and intelligent text processing* (pp. 136-145): Springer.

- Baral, C., & Liang, S. (2012). *From Knowledge Represented in Frame-Based Languages to Declarative Representation and Reasoning via ASP*. Paper presented at the KR.
- Baral, C., Vo, N. H., & Liang, S. (2012). *Answering Why and How questions with respect to a frame-based knowledge base: a preliminary report*. Paper presented at the ICLP (Technical Communications).
- Barnes, T., Stamper, J., & Madhyastha, T. (2006). *Comparative analyses of concept derivation using the q-matrix method and facets*. Paper presented at the Educational Data Mining Workshop at the AAAI 21st National Conference on Artificial Intelligence (AAAI2006), Boston, MA.
- Barr, A., Beard, M., & Atkinson, R. C. (1976). The computer as a tutorial laboratory: the Stanford BIP project. *International Journal of Man-Machine Studies*, 8, 567-596.
- Beck, J., Woolf, B. P., & Beal, C. (2000). ADVISOR: A machine learning architecture for intelligent tutor construction *Proceedings of the Seventeenth National Conference on Artificial Intelligence* (pp. 552-557). Menlo Park, CA: AAAI Press.
- Beck, J. E., Mostow, J., & Bey, J. (2004). *Can automated questions scaffold children's reading comprehension?* Paper presented at the Intelligent Tutoring Systems.
- Bloom, B. S. (1984). The 2 sigma problem: The search for methods of group instruction as effective as one-to-one tutoring. *Educational Researcher*, 13, 4-16.
- Bloom, B. S., & Krathwohl, D. (1956). *Taxonomy of educational objectives: The classification of educational goals, by a committee of college and university examiners. Handbook I: Cognitive Domain*. Green, NY: Longman.
- Bransford, J. D., Brown, A. L., & Cocking, R. R. (2000). *How people learn: Brain, mind, experience and school*. Washington, DC: National Academy Press.
- Brown, J. C., Frishkoff, G. A., & Eskenazi, M. (2005). *Automatic question generation for vocabulary assessment*. Paper presented at the Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing.

- Brusilovsky, P. (2007). Adaptive navigation support. In P. Brusilovsky, A. Kobsa, & W. Neidl (Eds.), *The Adaptive Web* (pp. 263-290). Berlin: Springer.
- Budanitsky, A., & Hirst, G. (2001). *Semantic distance in WordNet: An experimental, application-oriented evaluation of five measures*. Paper presented at the Workshop on WordNet and Other Lexical Resources.
- Buhrmester, M., Kwang, T., & Gosling, S. D. (2011). Amazon's Mechanical Turk a new source of inexpensive, yet high-quality, data? *Perspectives on psychological science*, 6(1), 3-5.
- Bull, S., & Kay, J. (2013). Open learner models as drivers for metacognitive processes *International Handbook of Metacognition and Learning Technologies* (pp. 349-365). New York: Springer.
- Burton, R. R., & Brown, J. S. (1982). An investigation of computer coaching for informal learning activities. In D. Sleeman & J. S. Brown (Eds.), *Intelligent Tutoring Systems*. New York: Academic Press.
- Cakmak, M., & Lopes, M. (2012). *Algorithmic and human teaching of sequential decision tasks*. Paper presented at the AAAI Conference on Artificial Intelligence, Toronto, Canada.
- Cao, X., Cong, G., Cui, B., Jensen, C. S., & Zhang, C. (2009). *The use of categorization information in language models for question retrieval*. Paper presented at the Proceedings of the 18th ACM conference on Information and knowledge management.
- Carbonell, J. R. (1970). AI in CAI: An artificial-intelligence approach to computer-assisted instruction. *Man-Machine Systems, IEEE Transactions on*, 11(4), 190-202.
- Cen, H., Koedinger, K. R., & Junker, B. (2006). Learning Factors Analysis -- A general method for cognitive model evaluation and improvement. In M. Ikeda, K. Ashley, & T.-W. Chan (Eds.), *Intelligent Tutoring Systems: 8th International Conference, ITS 2006* (pp. 164-175). Berlin: Springer.
- Cepeda, N. J., Pashler, H., Vul, E., Wixted, J. T., & Roher, D. (2006). Distributed practice in verbal recall tasks: A review and quantitative synthesis. *Psychological Bulletin*, 132(3), 354-380.

- Chaiklin, S. (2003). The zone of proximal development in Vygotsky's analysis of learning and instruction. *Vygotsky's educational theory in cultural context, 1*, 39-64.
- Chen, W. (2009). *Understanding mental states in natural language*. Paper presented at the Proceedings of the Eighth International Conference on Computational Semantics.
- Chi, M., Koedinger, K. R., Gordon, G., Jordan, P., & VanLehn, K. (2011). Instructional factor analysis: A cognitive model for multiple instructional interventions. In C. Conati & M. Ventura (Eds.), *Proceedings of the 4th International Conference on Educational Data Mining (EDM 2011)*.
- Clark, P., Porter, B., & Works, B. P. (2004). KM—The knowledge machine 2.0: Users manual. *Department of Computer Science, University of Texas at Austin*.
- Clement, B., Oudeyer, P.-Y., Roy, D., & Lopes, M. (2014). *Online optimization of teaching sequences with multi-armed bandits*. Paper presented at the Educational Data Mining.
- ConceptNet.
- Conejo, R., Guzmán, E., Millán, E., Trella, M., Pérez-De-La-Cruz, J. L., & Ríos, A. (2004). SIETTE: A web-based tool for adaptive testing. *International Journal of Artificial Intelligence in Education, 14*(1), 29-61.
- Corbalan, G., Kester, L., & Van Merriënboer, J. J. (2006). Towards a personalized task selection model with shared instructional control. *Instructional Science, 34*(5), 399-422.
- Corbalan, G., Kester, L., & van Merriënboer, J. J. G. (2008). Selecting learning tasks: Effects of adaptation and shared control on learning efficiency and task involvement. *Contemporary Educational Psychology, 33*, 733-756.
- Corbett, A., & Anderson, J. R. (1995). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction, 4*, 253-278.
- Curto, S., Mendes, A. C., & Coheur, L. (2012). Question Generation based on Lexico-Syntactic Patterns Learned from the Web. *Dialogue & Discourse, 3*(2), 147-175.

- Dillenbourg, P., & Self, J. (1992). A framework for learner modelling. *Interactive Learning Environments*, 2(2), 111-137.
- Draney, A. T., Pirolli, P., & Wilson, M. (1995). *A Measurement Model for a Complex Skill*. Hillsdale, NJ: Erlbaum.
- Eric, M. (1998). Covariance adjustment versus gain scores--revisited. *Psychological Methods*, 3, 309--327. doi:10.1037//1082-989X.3.3.309
- Fader, A., Zettlemoyer, L. S., & Etzioni, O. (2013). *Paraphrase-Driven Learning for Open Question Answering*. Paper presented at the ACL (1).
- Fahlman, S. E. (2006). Marker-passing inference in the scone knowledge-base system *Knowledge Science, Engineering and Management* (pp. 114-126): Springer.
- Falmagne, J., Koppen, M., Villano, M., Doignon, J., & Johannesen, L. (1990). Introduction to knowledge spaces: How to build, test and search them. *Psychological Review*, 97, 201-224.
- Fellbaum, C. (1998). *WordNet*: Wiley Online Library.
- Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A. A., . . . Prager, J. (2010). Building Watson: An overview of the DeepQA project. *AI magazine*, 31(3), 59-79.
- Fitts, P. M., & Posner, M. I. (1967). *Human performance*. Belmont, CA: Brooks/Cole.
- Foxvog, D. (2010). *Cyc Theory and Applications of Ontology: Computer Applications* (pp. 259-278): Springer.
- Gao, J., & Nie, J.-Y. (2012). *Towards concept-based translation models using search logs for query expansion*. Paper presented at the Proceedings of the 21st ACM international conference on Information and knowledge management.
- Goel, A. K., Rugaber, S., & Vattam, S. (2009). Structure, behavior, and function of complex systems: The structure, behavior, and function modeling language.

- Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 23(01), 23-35.
- Graesser, A. C., Lu, S., Jackson, G. T., Mitchell, H. H., Ventura, M., Olney, A., & Louwerse, M. M. (2004). AutoTutor: A tutor with dialogue in natural language. *Behavior Research Methods, Instruments, & Computers*, 36(2), 180-192.
- Hauser, D. J., & Schwarz, N. (2015). Attentive Turkers: MTurk participants perform better on online attention checks than do subject pool participants. *Behavior research methods*, 1-8.
- Heilman, M., & Smith, N. A. (2009). *Ranking automatically generated questions as a shared task*. Paper presented at the The 2nd Workshop on Question Generation.
- Hosseini, R., Hsiao, I.-H., Guerra, J., & Brusilovsky, P. (2015). *Off the beaten path: The impact of adaptive content sequencing on student navigation in an open student modeling interface*. Paper presented at the Artificial Intelligence in Education.
- Hsiao, I.-H., Bakalov, F., Brusilovsky, P., & Konig-Ries, B. (2011). Open social student modeling: Visualizing student models with parallel introspective views *User Modeling, Adaptation and Personalization* (pp. 171-182). Berlin: Springer.
- Hsiao, I.-H., Bakalov, F., Brusilovsky, P., & Konig-Ries, B. (2013). Progressor: social navigation support through open social student modeling. *New Review of Hypermedia and Multimedia*, 19(2), 112-131.
- Hsiao, I.-H., & Brusilovsky, P. (2012). Motivational social visualizations for personalized E-learning *21st Century Learning for 21st Century Skills* (pp. 153-165). Berlin: Springer.
- Hsiao, I.-H., & Brusilovsky, P. (submitted). *Social progress visualization: On the crossroads of learning analytics and open student modeling*.
- Hsiao, I.-H., Sosnovsky, S., & Brusilovsky, P. (2010). Guiding students to the right questions: adaptive navigation support in an E-Learning system for Java programming. *Journal of Computer Assisted Learning*, 26(4), 270-283.

- Huang, Y.-T., & Mostow, J. (2015). *Evaluating Human and Automated Generation of Distractors for Diagnostic Multiple-Choice Cloze Questions to Assess Children's Reading Comprehension*. Paper presented at the Artificial Intelligence in Education.
- Jiang, J. J., & Conrath, D. W. (1997). Semantic similarity based on corpus statistics and lexical taxonomy. *arXiv preprint cmp-lg/9709008*.
- Jouault, C., & Seta, K. (2014). *Content-Dependent Question Generation for History Learning in Semantic Open Learning Space*. Paper presented at the Intelligent Tutoring Systems.
- Kalady, S., Elikkottil, A., & Das, R. (2010). *Natural language question generation using syntax and keywords*. Paper presented at the Proceedings of QG2010: The Third Workshop on Question Generation.
- Kalyuga, S., Ayres, P., Chandler, P., & Sweller, J. (2003). The expertise reversal effect. *Educational Psychologist, 38*, 23-31.
- Kapelner, A., & Chandler, D. (2010). *Preventing Satisficing in online surveys*. Paper presented at the Proceedings of.
- Karamanis, N., Ha, L. A., & Mitkov, R. (2006). *Generating multiple-choice test items from medical text: A pilot study*. Paper presented at the Proceedings of the fourth international natural language generation conference.
- Kay, J. (2001). Learner control. *User Modeling and User-Adapted Interaction, 11*(1), 111-127.
- Kluger, A. N., & DeNisi, A. (1996). The effects of feedback interventions on performance: a historical review, a meta-analysis, and a preliminary feedback intervention theory. *Psychological bulletin, 119*(2), 254.
- Koedinger, K. R., Corbett, A., & Perfetti, C. (2012). The Knowledge-Learning-Instruction (KLI) framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive Science, 36*(5), 757-798.
- Koenker, R. (2005). *Quantile regression*: Cambridge university press.

- Krathwohl, D. R. (2002). A revision of Bloom's taxonomy: An overview. *Theory into practice*, 41(4), 212-218.
- Kumar, A. (2006). A scalable solution for adaptive problem sequencing and its evaluation. In V. Wade, H. Ashman, & B. Smyth (Eds.), *Adaptive Hypermedia and Adaptive Web-based Systems* (pp. 161-171). Berlin: Springer.
- Lazarinis, F., Green, S., & Pearson, E. (2010). Creating personalized assessments based on learner knowledge and objectives in a hypermedia Web testing application. *Computers & Education*, 55(4), 1732-1743.
- Le, N.-T., Kojiri, T., & Pinkwart, N. (2014). Automatic question generation for educational applications -- The state of the art. In T. van Do, H. A. L. Thi, & N. T. Nguyen (Eds.), *Advanced Computational Methods for Knowledge Engineering: Proceedings of the 2nd International conference on Computer Science, applied Mathematics and Applications* (pp. 325-339). New York: Springer.
- Leacock, C., & Chodorow, M. (1998). Combining local context and WordNet similarity for word sense identification. *WordNet: An electronic lexical database*, 49(2), 265-283.
- Lee, J., & Seneff, S. (2007). *Automatic generation of cloze items for prepositions*. Paper presented at the INTERSPEECH.
- Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval *Machine learning: ECML-98* (pp. 4-15): Springer.
- Lin, D. (1998). *An information-theoretic definition of similarity*. Paper presented at the ICML.
- Lin, Y.-C., Sung, L.-C., & Chen, M. C. (2007). *An automatic multiple-choice question generation scheme for english adjective understanding*. Paper presented at the Workshop on Modeling, Management and Generation of Problems/Questions in eLearning, the 15th International Conference on Computers in Education (ICCE 2007).
- Lindsey, R. V., Mozer, M. C., Huggins, W. J., & Pashler, H. (2013). *Optimizing instructional policies*. Paper presented at the Advances in Neural Information Processing Systems 26 (NIPS 2013).

- Lindsey, R. V., Shroyer, J. d., Pashler, H., & Mozer. (2014). Improving students long-term retention through personalized review. *Psychological Science*, 25(3), 639-647.
- Liu, M., & Calvo, R. A. (2012). *Using information extraction to generate trigger questions for academic writing support*. Paper presented at the Intelligent Tutoring Systems.
- Lopes, M., Clement, B., Roy, D., & Oudeyer, P.-Y. (2013). Multi-armed bandits for intelligent tutoring systems. *arXiv preprint*. Retrieved from
- Makatchev, M., Hall, B. S., Jordan, P. W., Pappuswamy, U., & VanLehn, K. (2005). *Mixed language processing in the Why2-Atlas tutoring system*. Paper presented at the Proceedings of the Workshop on Mixed Language Explanations in Learning Environments, AIED2005.
- Mason, W., & Watts, D. J. (2010). Financial incentives and the performance of crowds. *ACM SigKDD Explorations Newsletter*, 11(2), 100-108.
- McMillan, J. H., Hellsten, L., & Klinger, D. (2007). *Classroom assessment: Principles and practice for effective standards-based instruction*: Pearson/Allyn & Bacon Boston, MA.
- McMillan, J. H., & Lawson, S. R. (2001). Secondary Science Teachers' Classroom Assessment and Grading Practices.
- Melis, E., Andres, E., Budenbender, J., Frischauf, A., Gogvadze, G., Libbrecht, P., . . . Ullrich, C. (2001). ActiveMath: A generic and adaptive web-based learning environment. *International Journal of Artificial Intelligence in Education*, 12, 385-407.
- Mitkov, R., Ha, L. A., Varga, A., & Rello, L. (2009). *Semantic similarity of distractors in multiple-choice tests: extrinsic evaluation*. Paper presented at the Proceedings of the Workshop on Geometrical Models of Natural Language Semantics.
- Mitrovic, A., Martin, B., & Mayo, M. (2002). Using evaluation to shape ITS design: Results and experiences using SQL-Tutor. *User Modeling and User-Adapted Interaction*, 12(2-3), 243-279.

- Muldner, K., & Conati, C. (2007). Evaluating a decision-theoretic approach to tailored example selection *Proceedings of IJCAI 2007, the 20th International Joint Conference in Artificial Intelligence* (pp. 483-489). Menlo Park, CA: AAAI Press.
- Murray, R. C., & VanLehn, K. (2000). DT Tutor: A decision-theoretic, dynamic approach for optimal selection of tutorial actions. In G. Gauthier, C. Frasson, & K. VanLehn (Eds.), *Intelligent Tutoring Systems: 5th International Conference, ITS2000* (pp. 153-162). Berlin: Springer.
- Murray, R. C., & VanLehn, K. (2006). A comparison of decision-theoretic, fixed-policy and random tutorial action selection. In M. Ikeda, K. Ashley, & T.-W. Chan (Eds.), *Intelligent Tutoring Systems: 8th International Conference ITS 2006* (pp. 114-123). Berlin: Springer.
- Murray, R. C., VanLehn, K., & Mostow, J. (2004). Looking ahead to select tutorial actions: A decision-theoretic approach. *International Journal of Artificial Intelligence and Education*, 14(3-4), 235-278.
- Muñoz, K., Mc Kevitt, P., Lunney, T., Noguez, J., & Neri, L. (2010). PlayPhysics: an emotional games learning environment for teaching physics *Knowledge Science, Engineering and Management* (pp. 400-411): Springer.
- Nathan, M. J., Koedinger, K. R., & Alibali, M. W. (2001). *Expet blind spot: When content knowledge eclipses pedagogical content knowledge*. Paper presented at the Proceedings of the Third International Conference on Cognitive Science.
- Olney, A., Graesser, A. C., & Person, N. (2012). Question generation from concept maps. *Dialogue and Discourse*, 3(2), 75-99.
- Olney, A. M., Graesser, A. C., & Person, N. K. (2012). Question generation from concept maps. *Dialogue & Discourse*, 3(2), 75-99.
- Oppenheimer, D. M., Meyvis, T., & Davidenko, N. (2009). Instructional manipulation checks: Detecting satisficing to increase statistical power. *Journal of Experimental Social Psychology*, 45(4), 867-872.
- Paolacci, G., Chandler, J., & Ipeirotis, P. G. (2010). Running experiments on amazon mechanical turk. *Judgment and Decision making*, 5(5), 411-419.

- Papasalouros, A., Kanaris, K., & Kotis, K. (2008). *Automatic Generation Of Multiple Choice Questions From Domain Ontologies*. Paper presented at the e-Learning.
- Park, O.-C., & Tennyson, R. D. (1980). Adaptive design strategies for selecting number and presentation order of examples in coordinate concept acquisition. *Journal of Educational Psychology*, 72(3), 362-370.
- Patton, J. V., Chao, C.-I., & Reigeluth, C. M. (1986). A review of strategies for sequencing and synthesizing instruction. *Review of Educational Research*, 56(4), 437-471.
- Pavlik, P. I., & Anderson, J. R. (2005). Practice and forgetting effects on vocabulary memory: An activation-based model of the spacing effect. *Cognitive Science*, 29(4), 559-586.
- Pavlik, P. I., Cen, H., & Koedinger, K. R. (2009). Performance Factors Analysis: A new alternative to knowledge tracing *Proceedings of the 14th International Conference on Artificial Intelligence in Education*. Amsterdam: IOS Press.
- Peachy, D. R., & McCalla, G. I. (1986). Using planning techniques in intelligent tutoring systems. *International Journal of Man-Machine Studies*, 24, 77-98.
- Pek, P.-K., & Poh, K.-L. (2000a). Framework of a Decision-Theoretic Tutoring System for learning of Mechanics. *Journal of Science Education and Technology*, 9(4), 343-356.
- Pek, P.-K., & Poh, K.-L. (2000b). Using Decision Networks for Adaptive Tutoring. In S. S. Young, J. Greer, H. Maurer, & Y. S. Chee (Eds.), *Proceedings fo the International Conference on Computers in Education / International Convergence on Computer-Assisted Instruction* (pp. 1076-1084).
- Pho, V.-M., Ligozat, A.-L., & Grau, B. (2015). *Distractor Quality Evaluation in Multiple Choice Questions*. Paper presented at the Artificial Intelligence in Education.
- Puente, C., Sobrino, A., & Olivas, J. Á. (2009). Extraction of conditional and causal sentences from queries to provide a flexible answer *Flexible Query Answering Systems* (pp. 477-487): Springer.

- Rafferty, A. N., Brunskill, E., Griffiths, T. L., & Shafto, P. (2011). Faster teaching by POMDP planning. In G. Biswas (Ed.), *Artificial Intelligence in Education* (pp. 280-287). Berlin: Springer-Verlag.
- Razzaq, L., Feng, M., Heffernan, N. T., Koedinger, K. R., Junker, B., Nuzzo-Jones, G., . . . Walonoski, J. A. (2007). A Web-based authoring tool for intelligent tutors: Assessment and instructional assistance. In N. Nedjah (Ed.), *Intelligent Educational Machines*. Berlin: Springer.
- Reigeluth, C. M. (1999). What is instructional-design theory and how is it changing? In C. M. Reigeluth (Ed.), *Instructional-design theories and models: A new paradigm for instructional theory: Volume II* (pp. 5-30). Mahwah, NJ: Erlbaum.
- Resnik, P. (1995). Using information content to evaluate semantic similarity in a taxonomy. *arXiv preprint cmp-lg/9511007*.
- Robertson, S. E., Walker, S., Jones, S., Hancock-Beaulieu, M. M., & Gatford, M. (1995). Okapi at TREC-3. *NIST SPECIAL PUBLICATION SP*, 109-109.
- Roll, I., Aleven, V., McLaren, B., & Koedinger, K. R. (2007). Can help seeking be tutored? Searching for the secret sauce of metacognitive tutoring *Proceedings of the International Conference on Artificial Intelligence in Education* (pp. 203-210). Amsterdam: IOS Press.
- Roll, I., Aleven, V., McLaren, B., & Koedinger, K. R. (2011). Improving students' help-seeking skills using metacognitive feedback in an intelligent tutoring system. *Learning and Instruction*, 21, 267-280.
- Rosenshine, B., Meister, C., & Chapman, S. (1996). Teaching students to generate questions: A review of the intervention studies. *Review of educational research*, 66(2), 181-221.
- Rus, V., Cai, Z., & Graesser, A. C. (2007). Experiments on generating questions about facts *Computational Linguistics and Intelligent Text Processing* (pp. 444-455): Springer.
- Rus, V., Wyse, B., Piwek, P., Lintean, M., Stoyanchev, S., & Moldovan, C. (2010). *Overview of the first question generation shared task evaluation challenge*. Paper presented at the Proceedings of the Third Workshop on Question Generation.

- Salden, R. J. C. M., Paas, F. G. W. C., Broers, N. J., & Van Merriënboer, J. J. G. (2004). Mental effort and performance as determinants for the dynamic selection of learning tasks in air traffic control training. *Instructional Science*, 32, 153-172.
- Salden, R. J. C. M., Paas, F. G. W. C., Jeroen, J. G., & van Merriënboer, J. J. G. (2006). Personalize adaptive task selection in air traffic control: Effects on training efficiency and transfer. *Learning and Instruction*, 16, 350-362.
- Schwartz, D. L., & Bransford, J. D. (1998). A time for telling. *Cognition and instruction*, 16(4), 475-5223.
- Shute, V. J. (1995). SMART: Student Modeling approach for responsive tutoring. *User modeling and user-adapted instruction*, 5(1), 1-44.
- Tatsuoka, K. (1996). Architecture of knowledge structures and cognitive diagnosis: A statistical pattern recognition and classification approach. In P. Nichols, P. Chipman, & R. Brennan (Eds.), *Cognitively Diagnostic Assessment*. Mahwah, NJ: Erlbaum.
- Van de Sande, B. (2013). Properties of the Bayesian Knowledge Tracing model. *Journal of Educational Data Mining*, 5(2).
- VanLehn, K. (1988). Student modeling. In M. Polson & J. Richardson (Eds.), *Foundations of Intelligent Tutoring Systems* (pp. 55-78). Hillsdale, NJ: Lawrence Erlbaum Associates.
- VanLehn, K. (2008). Intelligent tutoring systems for continuous, embedded assessment. In C. A. Dwyer (Ed.), *The future of assessment: Shaping teaching and learning* (pp. 113-138). New York, NY: Lawrence Erlbaum Assoc.
- VanLehn, K., & Niu, Z. (2001). Bayesian student modeling, user interfaces and feedback: A sensitivity analysis. *International Journal of Artificial Intelligence in Education*, 12(2), 154-184.
- Varga, A. *Le An Ha 2010 WLW: A Question Generation System for the QGSTEC 2010 Task B*. Paper presented at the Proceedings of QG2010: The Third Workshop on Question Generation.

- Vassileva, J., & Deters, R. (1998). Dynamic courseware generation on the WWW. *British Journal of Educational Technology*, 29(1), 5-14.
- Weber, G., & Brusilovsky, P. (2001). ELM-ART: An adaptive versatile system for web-based instruction. *International Journal of Artificial Intelligence in Education*, 12, 351-384.
- Whitehill, J. (2012). *A stochastic optimal control perspective on affect-sensitive teaching*. (Ph. D.), University of California, San Diego.
- Wiemer-Hastings, P., Wiemer-Hastings, K., & Graesser, A. (1999). *Improving an intelligent tutor's comprehension of students with Latent Semantic Analysis*. Paper presented at the Artificial intelligence in education.
- Wikidata.
- Wolfe, J. H. (1976). *Automatic question generation from text-an aid to independent study*. Paper presented at the ACM SIGCUE Outlook.
- Wood, D. J. (2001). Scaffolding, contingent tutoring and computer-supported learning. *International Journal of Artificial Intelligence and Education*, 12, 280-292.
- Wyse, B., & Piwek, P. (2009). Generating questions from openlearn study units.
- Xue, X., Jeon, J., & Croft, W. B. (2008). *Retrieval models for question and answer archives*. Paper presented at the Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval.
- Zhang, L., & VanLehn, K. (submitted). *How do machine-generated questions compare to human-generated questions?*
- Zhang, L., Vanlehn, K., Girard, S., Burleson, W., Chavez-Echeagaray, M.-E., Gonzalez-Sanchez, J., & Hidalgo Pontet, Y. (2014). Evaluation of a meta-tutor for constructing models of dynamic systems. *Computers & Education*, 75, 196-217.

APPENDIX A

HUMAN-MACHINE QUESTIONS MAPPING

Human questions	Answer	Machine questions	Type
What are chloroplasts? -structure	Organelle in cell		Structure: boundary
What are chloroplasts? -function	Site of photosynthesis		function
What is light reaction process? -structure	It consist of cyclic photophosphorylation and noncyclic photophosphorylation	What are the 2 stages of the light reaction in photosynthesis?	Structure
What is light reaction process? -behavior-1	Light reaction takes water and sunlight as reactants and produces Oxygen, ATP and NADPH	What is/are the product(s) of the light reaction in photosynthesis? What does the light reaction in photosynthesis require?	Behavior-1
What is light reaction process? -behavior-2	It produces ATP and NADPH, both of which are reactants of Calvin cycle.		Behavior-2
What are dark and light reactions in photosynthesis?	Light reaction and Calvin cycle are the two stages of photosynthesis.	What are the 2 stages of photosynthesis?	Structure
What is light dependent and light independent reactions in photosynthesis?			
What is cyclic photophosphorylation? -structure	Chlorophyll first absorbs photons to make photosystem I release electrons. The electrons go through ferredoxin, cytochrome b, plastocyanin, and go back to photosystem I.	Could you describe the three sub-process involved in the cyclic photophosphorylation of light reaction?	Structure
What is cyclic photophosphorylation? -behavior-2	It produces ATP, which could be used in Calvin cycle.		Behavior-2

Where does photosynthesis occur in a plant?	chloroplasts	Where does photosynthesis by plant happen?	Structure
In which specific part of a leaf cell does photosynthesis takes place?			
What are the site of photosynthesis?			
Where is the chloroplast located?	Plant cell		Structure: boundary
In which part of the chloroplast does photosynthesis take place in?	Light reaction takes place in the thylakoid membranes, and Calvin cycle takes place in the stroma		Structure: boundary
What is the fluid part of the chloroplasts where the calvin cycle takes place?	stroma	Where does the calvin cycle of photosynthesis occur?	Structure
Where do the enzymatic reactions of the calvin cycle take place?			
Where do light reactions occur?	thylakoid membranes		Structure: boundary
In what part of photosynthesis sugar is produced?	Calvin cycle produces sugar	What is/are the product(s) of the calvin cycle in photosynthesis?	Behavior-1
In which part of photosynthesis is oxygen produced?	Light reaction produces oxygen	What is/are the product(s) of the light reaction in photosynthesis?	Behavior-1
What molecules are the reactants and products of photosynthesis?	The reactants of photosynthesis are water, CO ₂ , sunlight. The products of photosynthesis are oxygen, sugar	What does the photosynthesis need? What does the photosynthesis produce?	Behavior-1

What are the materials required by multicellular organisms for the processes of respiration and photosynthesis?	Respiration requires sugar and oxygen. Photosynthesis requires sunlight, CO ₂ and water	What does the cellular respiration need? What does the photosynthesis need?	Behavior-1
What are the raw materials of photosynthesis?	Photosynthesis requires sunlight, CO ₂ and water	What does the photosynthesis need?	Behavior-1
What are the four basic ingredients needed for photosynthesis?			
What are the raw materials of photosynthesis and its role?			
What are the three main limiting factors of photosynthesis?			
What is the energy molecule produced in the mitochondria and chloroplast?	ATP		Behavior-2
How do the raw materials of photosynthesis reach the chloroplasts of the leaves?	Water is absorbed through the root hair then into the xylem of the roots and into the xylem of the stem, it then goes through the xylem of the leaves into the mesophyll cells and finally into the chloroplasts. Carbon dioxide diffuses from the atmosphere through the stomata and then into the intercellular airspaces in the leaves and finally into the		Behavior-2

	<p>chloroplasts of the mesophyll cells.</p> <p>The chlorophyll and other pigments in the thylakoid membrane absorb the solar energy to drive photosynthesis.</p>		
What does the light reaction in photosynthesis produce?	It produces oxygen, ATP and NADPH	What is/are the product(s) of the light reaction in photosynthesis?	Behavior-1
What molecules produced during the light reaction of photosynthesis are needed to carry out the dark reaction?	Light reaction produces ATP and NADPH, both of which are used in Calvin cycle.	How does the light reaction support the calvin cycle?	Function
Why a toxin that inhibits an enzyme of the calvin cycle will also inhibit the light reactions?			
Which are the subproducts of the photochemical stage that are essential for the chemical stage of photosynthesis?			
What does the light reactions of photosynthesis supply the Calvin cycle?			
What is light reaction process? -function			

What are two products in photosynthesis of the light reaction that are used for the dark reaction?			
Where does the energy used to produce atp in the light reactions of photosynthesis come from?	Sunlight		Behavior-2
What does the calvin cycle use to produce high- energy sugars?	ATP and NADPH		Behavior-2
What else is needed besides water for the calvin cycle to take place?	CO2	What is the raw material of calvin cycle in photosynthesis?	Behavior-1
What does cyclic photophosphorylation produce?	ATP	What does the cyclic photophosphorylation in light reaction produce?	Behavior-1
What is cyclic photophosphorylation? -behavior-1			
What does non-cyclic photophosphorylation produce?	ATP, Oxygen and NADPH	What does the noncyclic photophosphorylation in light reaction produce?	Behavior-1
What is the function of the chloroplast? What is the chloroplast function?	They are the main site of photosynthesis in plant cells and help convert energy from the sun into sugars for the plant.		Function

<p>What is the function of the chloroplast in plant cells?</p> <p>What are chloroplasts and what is their role?</p> <p>What does chloroplasts enable plant cells to do?</p>			
<p>What is the function of chloroplast membranes?</p>	<p>The outer membrane is permeable to small organic molecules, whereas the inner membrane is less permeable and studded with transport proteins. The innermost matrix of chloroplasts, called the stroma, contains metabolic enzymes and multiple copies of the chloroplast genome.</p>		Function
<p>What do you think will happen if you insert chloroplast into animal cells in humans?</p>	<p>Animal may also perform photosynthesis.</p>		Function
<p>What is the role of light in the light reactions?</p>	<p>The light reactions use light energy to produce ATP and NADPH.</p>	<p>What is the role of the sunlight in the light reaction of photosynthesis?</p>	function
<p>What is the primary function of the calvin cycle in green plant?</p>	<p>Construct simple sugars from carbon dioxide (CO₂).</p>		function
<p>What is the main function of cyclic photophosphorylation?</p>	<p>The function of cyclic photophosphorylation is to produce ATP.</p>		Function
<p>What is cyclic photophosphorylation?</p> <p>-function</p>			

Why do plants need chloroplast and the animal cells don't need it?	Animals can acquire glucose by eating plants, but plants have to produce glucose by themselves.		Function
What would happen to photosynthesis if all three carbon sugar compounds produced in calvin cycle were used to make organic compounds?	The cycle would stop. You need to reinvest these compounds into the cycle to keep the biochemical process moving.		Function
How is light from the sun transformed into chemical energy to be used by the living beings on earth?	Light from the sun is transformed into chemical energy contained in organic material by the photosynthesis process. In photosynthesis light, water and carbon dioxide react and highly energetic glucose molecules and molecular oxygen are made.		Behavior-2
What are the stages into which photosynthesis is divided?	Light reaction and Calvin cycle	What are the 2 stages of photosynthesis?	Structure
What are the processes of the photochemical stage of the photosynthesis process?	cyclic photophosphorylation and noncyclic photophosphorylation	What are the 2 stages of the light reaction in photosynthesis?	Structure
What is NADP and NADPH? -structure	NADP is the abbreviation of the nicotinamide adenine dinucleotide phosphate cation. NADPH is made when NADP binds to one hydrogen atom.		Structure: boundary
What is NADP and NADPH? -function	NADP is a hydrogen acceptor. NADPH is the		Function

	form that actually transports hydrogen.		
Which chemical element is central in the chlorophyll molecule?	The chemical element that is central in the chlorophyll molecule is magnesium. One atom of magnesium is present in the center of an amalgam of eight nitrogen-containing carbon rings.		Structure
In which chloroplast structure are chlorophyll molecules found?	Chlorophyll molecules sit on the surface of each thylakoid		Structure: boundary
In photosynthesis, what is the molecule that donates hydrogen for photosynthesis?	H ₂ O		Function
Photosynthesis is the most important producer of molecular oxygen (O ₂) on our planet. From which molecule do oxygen atoms liberated by photosynthesis come? From which other molecule could one suspect they have come? What are the destinations of those oxygen atoms?	<p>The oxygen atoms liberated as molecular oxygen by the photosynthesis process come from water.</p> <p>One indeed could suspect that those oxygen atoms would have come from carbon dioxide. Oxygen atoms from carbon dioxide however are incorporated into glucose molecules and into water molecules liberated in the chemical stage of photosynthesis.</p>		Behavior-2
Where do the photochemical and the chemical stages of photosynthesis occur?	Stroma of chloroplast	Where does the calvin cycle of photosynthesis occur?	Structure
What is the destination of each of those substances produced	The electrons will replace those electrons lost by chlorophyll molecules in		Behavior-2

by water photosynthesis?	photophosphorylation. The hydrogen ions will be incorporated into hydrogen acceptor molecules (NADP) and later will be used in the synthesis of glucose during the chemical stage. Molecular oxygen is liberated to the atmosphere.		
How is the photic energy absorbed by chlorophyll transferred to ATP molecules in photophosphorylation?	Light excites chlorophyll and energizes electrons that jump off the molecule. The energy liberated when these electrons escape is used in the phosphorylation of ADP, forming ATP. The enzyme that catalyzes the reaction is the ATP synthase.		Behavior-2
How will be the resulting ATP of photophosphorylation used?	The resulting ATP is then consumed in the next chemical stage of photosynthesis to energetically enrich carbon dioxide for the formation of glucose.		Behavior-2
What are the chemical substances produced by water photolysis?	Free electrons, hydrogen ions and molecular oxygen are liberated, after the water photolysis.		Behavior-1
How are the large number of ATP and NADPH molecules used during the Calvin cycle consistent with the high value of	Glucose can be used by plants as food or structure and so without it a plant would die. The investment of so much ATP and NADPH is worth living for.		Behavior-2

glucose as an energy source?			
In the light actions, what is the initial electron donor? Where do the electrons finally end up?	Water (H ₂ O) is the initial electron donor; NADP ⁺ accepts electrons at the end of the electron transport chain, becoming reduced to NADPH.	Please explain the process of noncyclic photophosphorylation in the light reaction.	Structure
Which are the living beings that carry out photosynthesis?	Plants, algae and cyanobacteria are photosynthetic beings.		Function
Which is the cell organelle responsible for the absorption of light for the photosynthesis process in plants and algae?	Light is absorbed by chlorophyll, a molecule present in cytoplasmic organelles called chloroplasts.		Function
What are the roles of ATP and ADP for the cellular energetic metabolism?	The conversion between ATP, and ADP and phosphate, plays a central role in the energy metabolism of the cell.		Function
Why is it said that during photosynthesis carbon dioxide is enriched to form glucose?	During photosynthesis carbon dioxide is energetically enriched with hydrogen from water. Water broken by photolysis is the hydrogen donor of the reaction. Glucose is made of carbon and oxygen atoms obtained from carbon dioxide and of hydrogen atoms obtained from water.		Function
What are the roles of NADPH and ATP in the chemical stage of photosynthesis?	NADPH acts as reductant of carbon dioxide, it delivers highly energetic	In the calvin cycle of photosynthesis, what is the role of atp?	Function

	hydrogens to precursor molecules during the glucose formation process. ATP is an energy source for the reactions of chemical stage.	What does the nadph do in the reduction of 3 phosphoglycerate of calvin cycle?	
Why is the carbon dioxide concentration a limiting factor of the photosynthesis process?	The availability of carbon dioxide is a limiting factor for the photosynthesis process because this gas is a reagent of the reaction.	How is the carbon dioxide used in the carbon fixation of calvin cycle?	Behavior-2
When the carbon dioxide concentration is increased indefinitely, is photosynthesis also increased indefinitely?	Since enzymes catalyze the building of organic molecules with carbon atoms from carbon dioxide photosynthesis stops as soon as these enzymes become saturated, i.e., when all their activation centers are bound to their substrates. In that situation an increase of the carbon dioxide concentration will not increase the photosynthesis rate.		Behavior-2
Explain why a poison that inhibits an enzyme of the Calvin cycle will also inhibit the light reaction	The light reactions require ADP and NADP+, which would not be formed in sufficient quantities from ATP and NADPH if the Calvin cycle stopped.	How does the light reaction support the calvin cycle?	Function

APPENDIX B

THE SOURCE IN THE SEMANTIC NETWORK FOR SOME MACHINE

GENERATED QUESTIONS

Machine-generated questions	Knowledge base recorded relations	Intermediate relations
What are the 2 stages of photosynthesis?	subevent(light_reaction1, cyclic_photophosphorylation1) subevent(light_reaction1, noncyclic_photophosphorylation1) subevent(photosynthesis1, light_reaction1) instance_of(photothesis1,photosynthesis) instance_of(light_reaction1, light_reaction) instance_of(cyclic_photophosphorylation1, cyclic_photophosphorylation) instance_of(noncyclic_photophosphorylation1, noncyclic_photophosphorylation) sub_class_of(light_reaction, ...)	ancestor_of(event, light_reaction) event_num(light_reaction1, 2) var(noncyclic_photophosphorylation1, desc(noncyclic_photophosphorylation)) var(cyclic_photophosphorylation1, desc(cyclic_photophosphorylation)) var(ligt_reaction1, desc_detail(light_reaction, in , photosynthesis))
What is/are the product(s) of the light reaction in photosynthesis?	raw_material(light_reaction1, sunlight1) raw_material(light_reaction1, water_molecule1) subevent(photosynthesis1, light_reaction1) instance_of(photothesis1,photosynthesis) instance_of(light_reaction1, light_reaction) instance_of(sunlight1, sunlight) instance_of(water_molecule1, water_molecule) sub_class_of(light_reaction, ...)	ancestor_of(event, light_reaction) var(ligt_reaction1, desc_detail(light_reaction, in , photosynthesis)) var(sunlight1, desc(sunlight)) var(water_molecule, desc(water_molecule))
How does the light reaction support the	result(light_reaction1, atp1) result(light_reaction1, nadph1) raw_material(calvin_cycle1,atp1) raw_material(calvin_cycle1, nadph1)	ancestor_of(calvin_cycle, event) ancestor_of(light_reaction, event) ancestor_of(atp, entity)

calvin cycle?	instance_of(light_reaction1, light_reaction) instance_of(calvin_cycle1, calvin_cycle) instance_of(atp1, atp) instance_of(nadph1, nadph)	ancestor_of(nadph, entity) var(calvin_cycle1, desc(calvin_cycle)) var(light_reaction1, desc(light_reaction))
Where does the calvin cycle in photosynthesis occur?	site(calvin_cycle1, site, stroma1) instance_of(stroma1, stroma) subevent(phtotosynthesis1, calvin_cycle1) instance_of(calvin_cycle1, calvin_cycle) instance_of(photosynthesis1, photosynthesis)	var(calvin_cycle1, desc_detail(calvin_cycle, in , photosynthesis))

APPENDIX C

MANUALLY CREATED SYNONYMS

Light reaction -> {light reaction, light dependent reaction}

Calvin cycle -> {calvin cycle, light independent reaction, dark reaction}

Oxygen -> {Oxygen, O₂}

Ribulose biphosphate -> {Ribulose biphosphate, RuBP}

NADP plus -> {NADP plus , NADP⁺}

Three phosphoglycerate -> {three phosphoglycerate, 3-phosphoglycerate, 3PG}

Glyceraldehyde 3 phosphate -> { glyceraldehyde 3 phosphate, G3P}

Adenosine diphosphate -> {adenosine diphosphate, ADP }

Photosystem I -> {photosystem i, psi}