AUTOMATIC QUESTION GENERATION FROM TEXT FOR SELF-DIRECTED LEARNING

by

David Lindberg

B.Sc., Simon Fraser University, 2010

A Thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science

in the
School of Computing Science
Faculty of Applied Sciences

SIMON FRASER UNIVERSITY
Summer 2013

APPROVAL

Name: David Lindberg

Degree: Master of Science

Title of Thesis: Automatic Question Generation from Text for Self-Directed Learning

Examining Committee: Dr. James Delgrande
Chair

Dr. Fred Popowich, Senior Supervisor
Professor, Computing Science
Simon Fraser University

Dr. Anoop Sarkar, Supervisor
Associate Professor, Computing Science
Simon Fraser University

Dr. Maite Taboada, Examiner
Associate Professor, Linguistics
Simon Fraser University

Date Approved: June 26, 2013

Partial Copyright Licence

Abstract

Question generation from text is a Natural Language Generation task of vital importance for self-directed learning. Learners have access to learning materials from a wide variety of sources, and these materials are not often accompanied by questions to help guide learning. Prior question generation techniques have focused primarily on generating factoid questions, which are often not the most pedagogically important questions for a learner. Furthermore, prior techniques have not fully leveraged the semantic content of learning materials and have not often been evaluated in a pedagogically-inspired framework. This thesis introduces a novel template-based approach to question generation that combines semantic roles with a method of generating both general and domain-specific questions. We evaluate our approach in a way that is mindful of the context in which the generated questions are to be used. This evaluation shows our approach to be effective in generating pedagogically-useful questions.

Keywords: natural language generation, question generation, semantic role labeling, templates, self-directed learning

Acknowledgments

There are many people I need to thank. First and foremost, thanks are due to my senior supervisor, Dr. Fred Popowich, for his guidance and patience and for enabling me to work on a variety of interesting research projects during the last two years. I would also like to thank my supervisor, Dr. Anoop Sarkar, and my thesis examiner, Dr. Maite Taboada, for their valuable comments and suggestions. Thanks also to Dr. Jim Delgrande for chairing my thesis defence.

I would be remiss in not thanking Dr. Phil Winne and Dr. John Nesbit from the Faculty of Education for providing the educational impetus for this research. Without them, this thesis would not exist. I am also indebted to Kiran Bisra for performing a very thorough evaluation.

Thanks must also go to the members of the natural language lab, past and present, for creating a positive and friendly environment.

Finally, thanks to family and friends for support and understanding. A special thank you goes to friends Ania, Beth, and Kevin.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Task and motivation

Natural Language Generation (NLG) is the task of translating machine-readable, non-linguistic information into an equivalent human language representation. In simple terms, the NLG task requires deciding *what to say* and *how to say it*. More concretely, the basic NLG tasks are content determination, discourse planning, sentence aggregation, lexicalization, referring expression generation, and linguistic realization [31]. Content determination involves deciding which information should be mentioned in the natural language text. Discourse planning involves determining the order in which content should be presented. Sentence aggregation involves organizing content into sentences. Lexicalization is the process of determining appropriate words and phrases to use in expressing the content. Referring expression generation is needed to make the language sound more natural (e.g., by replacing nouns with pronouns where appropriate). Finally, linguistic realization is the process of applying rules of grammar to produce valid sentences.

NLG falls under the larger umbrella of Natural Language Processing (NLP), which can be more broadly described as the study of computers interacting with human language. NLP also includes Natural Language Understanding (NLU), which can be thought of as the inverse of NLG. An NLU system seeks to convert human language into a machine-understandable representation.

Question Generation (QG) from text is an NLG task concerned with generating questions from unstructured text. As such, it uses tools from both NLU and NLG. In order to generate questions from text, we first have to understand it, even if that understanding is somewhat

shallow. Rus and Graesser describe just a few of the many possible applications of QG [32]:

1. Suggesting good questions that learners might ask while reading documents and other media
2. Questions that human and computer tutors might ask to promote and assess deeper learning
3. Suggested questions for patients and caretakers in medicine
4. Suggested questions that might be asked in legal contexts by litigants or in security contexts by interrogators
5. Questions automatically generated from information repositories as candidates for Frequently Asked Question (FAQ) facilities

The focus of this thesis is the first of these applications. QG from text is of vital importance for self-directed learning. It has been shown [6] that questioning is an effective method of helping learners learn better. Unfortunately, many studies have shown that students often are not aware of their own knowledge deficits, ask questions infrequently, and tend to ask shallow questions [16]. Learners have access to learning materials from a wide variety of sources, and these materials are not often accompanied by questions to help promote learning. An abundance of learning material and a poverty of questions makes QG from text a worthwhile task.

Given that the need for automatic QG is real, we need to consider not only the questions we *can* generate but also the questions we *should* generate. A framework that might help motivate this discussion is the Taxonomy of Educational Objectives, originally published in 1956 by Bloom et al.[4] and later revised [3]. For the purpose of our discussion, we will refer to the revised version and call it simply the "revised Bloom taxonomy" or simply "the taxonomy." The taxonomy is a hierarchy that defines a *knowledge* dimension and a *cognitive process* dimension. We will focus on the latter. The cognitive process dimension describes a hierarchy of six categories (Table 1.1) and their associated cognitive processes [22].

Generally speaking, the categories of the taxonomy increase in complexity at each level, but they do not form a rigid hierarchy, as some categories overlap. For example, *explaining*, a level 2 process, is more cognitively complex than *executing*, a level 3 process [22]. Although the taxonomy is somewhat loosely defined, it does suggest that we should consider generating

Level	Category	Cognitive Processes
1	Remember	recognizing, recalling
2	Understand	interpreting, exemplifying, classifying, summarizing, inferring, comparing, explaining
3	Apply	executing, implementing
4	Analyze	differentiating, organizing, attributing
5	Evaluate	checking, critiquing
6	Create	generating, planning, producing

Table 1.1: The cognitive process categories of the revised Bloom taxonomy

questions that are likely to fall somewhere above level 1. Questions that require that the learner merely recall facts as they read are precisely the shallow questions learners already ask themselves.

Much prior work has examined QG from single sentences, but these techniques have focused almost exclusively on generating *factoid* questions. Factoids are questions which require the learner to recall facts explicitly stated in the source text. These are distinctly level 1 questions. Many of these methods rely on transformations driven by syntax and/or semantics and question word replacement to transform declarative sentences into interrogatives. For example, consider the sentence in (1.1).

Expanding urbanization is competing with farmland for growth and putting pressure on available water stores. (1.1)

We would see these methods generating questions (1.2) through (1.4).

What is competing with farmland for growth and putting pressure on available water stores? (1.2)

Expanding urbanization is competing with what for growth and putting pressure on available water stores? (1.3)

What is expanding urbanization? (1.4)

While the last of these questions is perhaps more useful, because it requires the learner to go beyond the information given in the original sentence, the other two require only that the learner has read the sentence in question. Such questions are often not the most pedagogically important questions for a learner. A more useful question might be the more general *What are some of the consequences of expanding urbanization?*

Our motivation is to demonstrate that there is still value in single-sentence QG, and given the right approach, we *can* escape the realm of the factoid and pose questions that are not just syntactically and semantically sound but also more useful pedagogically. We want to demonstrate that by leveraging the semantic content of learning materials, we can generate questions that are more pedagogically-useful than factoids.

1.2 Contributions

This thesis makes several contributions. We demonstrate a novel approach to QG that combines sentence-level semantic analysis with semantically-motivated question templates. We enable a mixture of domain-dependent and general-purpose templates that are more compact than previous template-based approaches have produced. Finally, we evaluate our approach in a way that is driven by an awareness of the context in which the generated questions are to be used. This evaluation shows our approach to be effective in generating pedagogically-useful questions.

1.3 Outline of this thesis

In the next chapter, we review previous efforts in QG from text, focusing on generation from single sentences. In Chapter 3, we describe our approach to QG in detail. In Chapter 4, we describe our pedagogically-motivated evaluation and examine its results. Chapter 5 concludes this thesis with a review of our approach and its contributions and discussion of future work.

Chapter 2

Previous Work

In this chapter, we will consider the tasks common to all approaches to QG from text and describe the approaches others have taken to accomplish these tasks. Previous efforts in QG from text can be broadly divided into three categories: syntax-based, semantics-based, and template-based. These three categories are not entirely disjoint. Systems we might place in the syntactic category are often observed using elements of semantics and vice-versa. A system we would call template-based must to some extent use syntactic and/or semantic information. Regardless of the approach taken, systems must perform at least four tasks:

1. content selection: picking spans of source text (typically single sentences) from which questions can be generated
2. target identification: determining which specific words and/or phrases should be asked about
3. question formulation: determining the appropriate question(s) given the content identified
4. surface form generation: producing the final surface-form realization

There is disagreement in the literature concerning the number of tasks and their description. The above list is our own interpretation. These tasks are not always discrete, and may not necessarily occur in this order. Task 2 need not always precede task 3. Target identification can drive question formulation and vice-versa. A system constrained to generating specific kinds of questions will select only the targets appropriate for those kinds of questions.

Conversely, a system with broader generation capabilities might pick targets more freely and (ideally) generate only the questions that are appropriate for those targets. Examples of both of these cases are found in the literature and will be discussed in the following sections.

For the following discussion, we consider the methods used in performing tasks 2 and 4 to be the primary discriminators in determining the category into which a given method is best placed.

2.1 Syntax-based methods

Given a sentence, the syntax-based methods follow a common strategy: parse the sentence to determine syntactic structure, simplify the sentence if possible, identify key phrases, and apply syntactic transformation rules and question word replacement. Syntax-based methods comprise a large portion of the existing literature. Kalady et al.[20], Varga and Ha [35], Wolfe [37], and Ali et al.[2] provide a sample of these methods. Let us now examine the techniques used by these syntax-based methods in more detail. Afterward, we will see some examples to demonstrate the results.

2.1.1 Sentence simplification

Sentence simplification is often viewed as a necessary pre-processing step. With the exception of Varga and Ha, each of the works cited above uses one or more simplification steps, including splitting sentences containing independent clauses, appositive removal, prepositional phrase removal, discourse marker removal, and relative clause removal. Varga and Ha prefer to use a filtering mechanism to prevent generation from complex sentences rather than attempt simplification [35]. While simplification makes some aspects of question generation easier, it also introduces new problems that must be handled. For example, when sentences containing independent clauses are split, it might also be necessary to perform anaphora resolution to avoid generating vague questions. Consider sentence (2.1).

The boy went to school on Monday, and he came home on Thursday . (2.1)

Sentence simplification would cause this to be split into sentences (2.2) and (2.3).

The boy went to school on Monday. (2.2)

He came home on Thursday. (2.3)

The subject of (2.3) is simply *He*, which could result in a vague question such as (2.4) being generated.

When did he come home? (2.4)

Replacing *He* with *The boy* prevents this type of vagueness.

2.1.2 Key phrase identification

Key phrases are phrases (sometimes single words) that are the best targets (by some measure) for question generation. Various approaches have been adopted for identifying these key phrases. One approach borrows techniques from automatic summarization to identify all the key phrases from an entire document before delving into question generation at sentence level [20]. Another approach is to consider subject, object, and prepositional phrases containing a named-entity to be key phrases [2]. The latter approach has also been extended to include other phrases, such as adverbials [35].

2.1.3 Syntactic transformation and question word replacement

The final steps in syntax-based methods transform declarative sentences into questions by manipulating syntax trees and inserting question words. Heilman and Smith [18] use predefined interaction rules to define a priori the types of questions that will be generated from subject-verb-object-preposition patterns having specific named-entity types in each of those positions. Kalady et al.[20], who use the document-level key phrase identification mentioned previously, define separate heuristics for creating questions depending on whether key phrases are contained in subject noun phrases, object noun phrases, appositives, prepositional phrases, or adverbials.

2.1.4 Examples

A few examples will demonstrate the behaviour and capabilities of syntax-based methods. Consider (2.5), an example given by Kalady et al.[20].

Barack Obama is the president of America. (2.5)

This sentence does not need any simplification. Its subject noun phrase is extracted and identified as an entity of type *person*. The question in (2.6) is formed by replacing the subject with the appropriate question word.

Who is the president of America? (2.6)

Kalady et al. provide another example, seen in (2.7), in which sentence simplification would be performed.

Mexico City, the biggest city in the world, has many interesting archaeological sites. (2.7)

This sentence contains an appositive, *the biggest city in the world*. Extracting the appositive and transforming it into a question yields (2.8).

Which/Where is the biggest city in the world? (2.8)

A shorter sentence demonstrates the predictable behaviour of syntax-based QG. Consider (2.9), an example given by Ali et al.[2].

Tom ate an orange at 7 pm. (2.9)

Questions (2.10) through (2.13) are generated by Ali et al.

Who ate an orange? (2.10)

Who ate an orange at 7 pm? (2.11)

What did Tom eat? (2.12)

When did Tom eat an orange? (2.13)

All of these examples serve to illustrate the fundamental behaviour of syntax-based methods. QG from text is essentially reduced permuting syntactic elements of a sentence and replacing words or phrases with question words.

The most important strength of these syntax-based methods is their portability. These methods rely on general-purpose transformations that are applicable to any domain. It might, however, be necessary to re-train a named entity recognition model in order to better identify named entities when applying these methods to more esoteric domains. Unfortunately, the portability of these methods limits the kinds of questions they can produce. As the examples have shown, by merely rearranging syntactic elements of a sentence as these methods do, factoid questions abound.

2.2 Semantic Role Labeling

The methods we have labeled "semantics-based" differ from other work in that their method(s) of target identification are based on semantic rather than syntactic analysis. Here, we refer to shallow semantic analysis, such as Semantic Role Labeling (SRL). SRL is a key component of our approach as well, so we will provide an overview before we discuss semantics-based methods.

Given a sentence, a semantic role labeler attempts to identify the predicates (relations and actions) and the semantic entities associated with each of those predicates. The set of semantic roles used in PropBank [29] includes both predicate-specific roles whose precise meaning is determined by their predicate (`Rel`) and general-purpose adjunct-like modifier roles whose meaning is consistent across all predicates. The predicate specific roles are `Arg0`, `Arg1`, ..., `Arg5` and `ArgA`. Table 2.1 shows a complete list[1] of the modifier roles.

Role	**Meaning**
ArgM-LOC	location
ArgM-EXT	extent
ArgM-DIS	discourse connectives
ArgM-ADV	adverbial
ArgM-NEG	negation marker
ArgM-MOD	modal verb
ArgM-CAU	cause
ArgM-TMP	time
ArgM-PNC	purpose
ArgM-MNR	manner
ArgM-DIR	direction
ArgM-PRD	secondary predication

Table 2.1: PropBank ArgM roles

As noted, the precise meaning of roles `Arg0` through `Arg5` and `ArgA` depend on the predicate in question. Generally speaking, though, `Arg0` can be considered the *agent* of the predicate, and `Arg1` can be considered the *patient* or *theme*. The `ArgA` role is used to identify the agent of some induced action. The `ArgM` labels are general-purpose, and have

[1]The list of modifiers defined in PropBank was expanded in 2012. The SRL-based work referred to in this thesis is consistent with the previous 2005 revision.

a consistent meaning across all predicates. A few examples will make this clear. Consider the sentence in (2.9). The SRL parse would be as seen in (2.14).

$$[Tom]_{Arg0}\ [ate]_{Rel}\ [an\ orange]_{Arg1}\ [at\ 7\ pm]_{ArgM-TMP}. \tag{2.14}$$

In this case, *Tom* is the *eater*, *an orange* is the *thing eaten*, and *7 pm* is the *time* at which the action happened. A more complicated example is seen in (2.15).

Climate change can include changes in pressure, temperature, and precipitation. (2.15)

In this case, there is no agent causing the *include* action. The SRL parse of this sentence is as seen in (2.16).

$$\begin{aligned}&[Climate\ change]_{Arg2}\ [can]_{ArgM-MOD}\ [include]_{Rel}\\&[changes\ in\ pressure,\ temperature,\ and\ precipitation]_{Arg1}.\end{aligned} \tag{2.16}$$

We see that, indeed, there is nothing to fill the `Arg0` role. Instead, *Climate change* is a *group*, and *changes in pressure, temperature, and precipitation* are the things that climate change *can include*.

Additional prefixes are available to identify arguments that refer to semantic roles realized elsewhere in a sentence (`R-`) or roles tat are split into disjoint substrings (`C-`). Consider (2.17) and (2.18).

Greenhouse gases that occur naturally... (2.17)

The word *that* can be assigned the `R-Arg1` role, because it refers to the `Arg1` argument, which is *greenhouse gases*.

Climate change, they say, is a real threat. (2.18)

In (2.18), the thing *they say* is *climate change is a real threat*, which semantically fills the `Arg1` role. However, because the argument is split, we must assign *climate change* to the `Arg1` role and *is a real threat* to the `C-Arg1` role.

The inner-workings of a semantic role labeler are outside the scope of this thesis. Refer to Pradhan et al.[30] and Collobert et al.[11] for example implementations of an SRL system. We have used the system of Collobert et al.

Semantic Role	Question Type
ArgM-MNR	How
ArgM-CAU	Why
ArgM-PNC	Why
ArgM-TMP	When
ArgM-LOC	Where
ArgM-DIS	How

Table 2.2: Roles and their associated question types

2.3 Semantics-based methods

Like syntax-based methods, semantics-based methods have relied on transformations to turn declarative sentences into questions. The difference, though, is that semantics-based methods have used semantic rather than syntactic analysis to drive these transformations. Mannem et al.[25] show us a system that combines SRL with syntactic transformations. In the content selection stage, a single sentence is first parsed with a semantic role labeler to identify potential targets. Targets are selected using simple selection criteria. Any of the predicate-specific semantic arguments (`Arg0-Arg5`), if present, are considered valid targets. It should be noted that any predicates having fewer than two of these arguments are not considered viable and are ignored, the justification being that at least two arguments are needed to formulate questions. We will return to this consideration when we describe our approach in Chapter 3. Mannem et al. further identify `ArgM-MNR`, `ArgM-PNC`, `ArgM-CAU`, `ArgM-TMP`, `ArgM-LOC`, and `ArgM-DIS` as potential targets. These roles are used to generate additional questions that cannot be attained using only the `Arg0-Arg5` roles. For example, `ArgM-LOC` can be used to generate a *where* question, and an `ArgM-TMP` can be used to generate a *when* question. See Table 2.2 for complete list of question types.

After targets have been identified, these, along with the complete SRL parse of the sentence are passed to the question formulation stage. The first step in this stage is to identify the *verb complex* for each identified target in the sentence. This consists of the main verb and any auxiliaries or modals and is identified from the dependency parse of the sentence. For each target, questions are generated using a series of simple transformations. First, the target itself is deleted. If the target contained any prepositions, the target is replaced with its first preposition. Second, a question word is picked. The correct question

word is determined based on the semantic role of the target and the named entity (if any) it contains. Third, any adjuncts appearing to the left of the predicate are moved to the end of the sentence. In the last transformation step the sentence is transformed into an interrogative. Auxiliaries or modals, if present, are moved to the beginning of the sentence before the question word is inserted in the initial position. If no auxiliaries or modals are present, one of *do, does,* or *did* is added depending on the POS tag of the predicate.

Consider the SRL parse in (2.14). Using Mannem et al.'s method, we would expect to generate (2.19) through (2.21), each generated from one of the target semantic roles:

Who ate an orange at 7 pm? (2.19)

What did Tom eat at 7 pm? (2.20)

When did Tom eat an orange? (2.21)

Mannem et al. participated in the 2010 Question Generation Shared Task Evaluation Challenge (QGSTEC) [33], which called for the generation of six questions from a given paragraph. They ranked their questions using two heuristics and then selected the top six. In their ranking scheme, questions are ranked first by the depth of their predicate in the dependency parse of the original sentence. This is based on the assumption that questions arising from main clauses are more desirable than those generated from deeper predicates. In the second stage, questions with the same rank are re-ranked according to the number of pronouns they contain, with questions with fewer pronouns having higher rank.

As part of their participation in the shared task, they were also required to generate questions at three scopes: general, medium, and specific. General questions were intended to be paragraph scope, medium questions were intended to be about main ideas expressed in a paragraph, and specific questions were supposed to be answerable from a sentence or phrase. Mannem et al. did not perform any paragraph-level analysis to generate their general questions. Instead, they generated a question from the first sentence of each paragraph using a different set of transformation rules. If the main verb was a copula, they generated a question about its right argument. Otherwise, they generated a question by relativizing the right argument of the main clause. The two cases are demonstrated by (2.22) and (2.24) respectively.

Intelligent tutoring systems consist of four different subsystems or modules. (2.22)

The predicate in (2.22), *consist*, is a copula, so the question seen in (2.23) is formed using the right argument, which is an `Arg2` in this case.

What are the four different subsystems or modules that intelligent tutoring systems consist of? (2.23)

The predicate *have* in (2.24) is not a copula, so the second transformation rule is used to generate (2.25).

But one-handed backhands have some advantages over two-handed players. (2.24)

What are some advantages over two-handed players that one-handed backhands have? (2.25)

Yao and Zhang [39] demonstrate an approach to QG based on minimal recursion semantics (MRS), a framework for shallow semantic analysis developed by Copestake et al.[12] Their method uses an eight-stage pipeline to convert input text to a set of questions. The key parts of their approach are MRS decompositions and MRS transformations. Decompositions convert complex sentences into simple sentences from which MRS transformations can generate questions. Yao and Zhang note that these decompositions are not optional, as MRS transformations can only be applied to simple sentences. This is a distinction between their work and other approaches in which sentence simplification is an optional step.

If we look at the output of this method, we can see that it suffers from severe over-generation. An example provided by Yao and Zhang is seen in (2.26).

Jackson was born on August 29, 1958 in Gary, Indiana. (2.26)

Yao and Zhang generate 12 questions, each of which is merely a permutation of the semantically-meaningful substrings of the original sentence or a permutation with one semantic constituent replaced with a question word. Two of these questions are seen in (2.27) and (2.28).

When was Jackson born in Gary , Indiana? (2.27)

On August 29 , 1958 was Jackson born in Gary , Indiana? (2.28)

To deal with this over-generation, Yao and Zhang implement statistical ranking based on a linear weighting of a maximum entropy score for candidate MRS parses and a language

model score for candidate surface realizations. Their language model is estimated from a corpus of questions augmented with English Wikipedia[2] for increased lexical coverage.

Although their techniques differ from syntax-based methods, these semantics-based methods achieve a very similar result, so their strengths and weaknesses are very much the same as those of syntax-based methods. They are indeed portable, but they too produce factoids. Granted, Mannem et al.[25] do demonstrate that SRL can be used to identify appropriate question words in the absence of named entities, but question word replacement is not the greatest challenge to be solved in QG from text.

2.4 Template-based methods

Question templates offer the ability to ask questions that are not as tightly-coupled to the exact wording of the source text as syntax and semantics-based methods. A question template is any pre-defined text with placeholder variables to be replaced with content from the source text. Question templates allow question generation systems to leverage human expertise in language generation. Our approach is template-based, so we will spend more time discussing previous efforts in template-based methods than we spent discussing syntactic and semantics-based methods.

Cai et al.[5] present NLGML (Natural Language Generation Markup Language), a language that can be used to generate questions of any desired type. NLGML uses syntactic pattern matching and semantic features for content selection and question templates to guide question formulation and surface-form realization. In writing an NLGML script, a human author defines *category-pattern-template* blocks. *Categories* bind *patterns* to *templates*. Patterns, which combine syntactic phrase structure and semantic features, are used to identify appropriate sentences, and templates define rules used to transform the source sentence into the desired question. An example is shown in Figure 2.1

As this example demonstrates, a pattern need not specify a complete syntax tree. The `<star />` tag is a wildcard that matches any subtree. The variables `_person_` and `_place_` are used by the templates to extract the text of the corresponding noun phrases, though in general, variables may be assigned to any portion of the subtree. Variables provide a mechanism for extracting substrings regardless of the underlying subtree syntax.

[2]obtained from http://meta.wikimedia.org/wiki/Wikipedia_Machine_Translation_Project

```
<category>
  <pattern>
    <S>
      <NP person="true">_person_</NP>
      <VP> <VBD>went</VBD>
        <PP> <TO>to</TO> <NP location="true">_place_</NP> </PP>
      </VP>
      <star />
    </S>
  </pattern>
  <template> Where did _person_ go? </template>
  <template> Who went to _place_? </template>
  <template> Why did _person_ go to _place_? </template>
</category>
```

Figure 2.1: Example of an NLGML category-pattern-template block

Patterns can also impose semantic constraints. In this example, the `person="true"` attribute of the first noun phrase indicates that only noun phrases containing a named entity of type `person` are considered a valid match. Similarly, the second noun phrase requires a named entity of type `location`. Matching these semantic features enables an NLGML-based system to determine when it is appropriate to ask *who* and *where* questions. A `time` semantic feature can similarly be used to generate *when* questions [5].

The pattern in Figure 2.1 can be thought of as encoding the idea that "somebody went somewhere." When a source sentence matches this pattern, three questions are generated using the defined templates. Cai et al. give the example seen in (2.29), a sentence which matches the pattern and generates the questions seen in (2.30) through (2.32).

The boy went to school. (2.29)

Where did The boy go? (2.30)

Who went to school? (2.31)

Why did The boy go to school? (2.32)

As you can see, simple "copy and paste" templates are not a panacea for surface-form realization. Mechanisms for changing capitalization of words and changing verb

conjugation (when source sentence verbs are to appear in the output text) need to be provided. NLGML provides such functions, though the only examples given by Cai et al. are `_lowerFirst(arg)`, which changes the first character of `arg` to lowercase, and `_getLemma(verb)`, which returns the lemma of `verb`.

Ceist [38] is another example of a template-based approach. Like NLGML, Ceist uses pattern matching, variables, and templates to transform source sentences into questions. One key difference between Ceist and NLGML is how patterns are specified. In Ceist, patterns are specified using the syntax of Stanford Tregex [23]. A second difference is that Ceist does not rely on explicit named entity recognition. Instead, Ceist treats named entity recognition as an additional pattern matching exercise. A proposed pattern for identifying `person` entities appears in (2.33).

```
personNP : NP <- NNP !<, DT
```
(2.33)

Here, `personNP` is matched with any noun phrase whose first word is not a determiner and last word is a proper noun. Clearly, this would not match with *the boy* from (2.29), as the first word is the determiner *the.* But suppose we had the source sentence seen in (2.34).

John Smith went to school. (2.34)

In this case, *John Smith* would indeed be identified as a `personNP`. Ceist could then use the pattern and templates shown in Figure 2.2 to generate output similar to the NLGML template in Figure 2.1.

```
Pattern:
    S < (personNP1 . (VP < (VBD . (PP < (TO . NP=g2 )))))
Templates:
    Where did /1 go?
    Who went to /2?
    Why did /1 go to /2?
```

Figure 2.2: Example of a pattern and templates in Ceist-compatible format

We will explain only enough Tregex syntax to understand this example. Refer to [23] for more thorough coverage. The `<` operator identifies a dominance relationship, with the left operand dominating the right operand, and the `.` operator indicates a sibling relationship

between its operands. Parentheses are necessary, because these two operators have the same precedence. As with NLGML, variables can be assigned to subtrees. Each variable name ends with a reference number. In this example, `personNP1` is a variable referring to the first phrase immediately dominated by `S` and is of type `personNP`. The inner-most noun phrase is assigned to the variable `g2`. In the templates, `/X` sequences are replaced with the text of the variable whose reference number is `X`.

Applied to the sentence seen in (2.34), this template generates (2.35) through (2.37).

Where did John Smith go? (2.35)

Who went to school? (2.36)

Why did John Smith go to school? (2.37)

Though patterns in Ceist are definitely more compact than an equivalent specification in NLGML, they are arguably harder to write. Wyse et al. describe an iterative process in which human template authors iteratively refine patterns until only desired questions are produced [38]. In other words, extensive trial-and-error is required to produce good templates for Ceist.

Mostow and Chen show us a template-based system designed for a highly-constrained interaction scenario [27]. Their system is designed to help children improve their reading comprehension. The approach, which is based on an instructional model proposed by Duke and Pearson [13], includes four key steps:

1. describe a strategy to the learner
2. model the strategy
3. scaffold the strategy
4. prompt the strategy

A fifth step in Duke and Pearson's model, the learner practicing independently, is outside the scope of the approach. In step 1, the student is presented with a description of the strategy they should follow. The example given is as seen in (2.38).

Good readers often ask themselves questions before, during, and after they read, to help them deepen their understanding of a text. (2.38)

Note that the content of the prompt depends on the strategy the student is being encouraged to apply. It is independent of the particular source text the student is reading.

Steps 2 and 3 are where the system must incorporate the content of the source text into the interaction with the learner. At step 2, the system generates prompts such as (2.39).

Right now, the question I'm thinking about is, X? (2.39)

Here, X is a question specific to the text the student is reading and can be a *what, why, or how* question. Step 2 is executed the first time an opportunity to demonstrate the strategy is detected. The system is designed to help students ask questions about the mental states of characters, so it must first identify sentences that indicate changes in mental state. Identifying these sentences is done by locating modal verbs from one of ten categories. These categories are identified using a WordNet-based semantic similarity metric [10]. Transforming sentences into appropriate questions requires inferring the mental state being described. This is done using an inference engine, which is also described in [10]. Among the output of this inference engine is a situation model describing characters and their mental states.

Question templates and an un-specified morphology generator are used to generate the text-specific questions. Due to the constrained scope of the system, only the three question templates seen in (2.40) through (2.42) are defined. In these examples, `character` refers to a character in the story the child is reading. The characters, verbs, complements, and participles are taken from the situation model.

`What did <character> <verb>?` (2.40)

`Why/How did <character> <verb> <complement>?` (2.41)

`Why was/were <character> <past-participle>?` (2.42)

In step 3, the system guides the learner in crafting new questions or the same type. In step 4, the system offers additional text-independent prompts, such as *What's a good question to ask about what you just read?*, to encourage the learner to apply the strategy again. Refer to [27] for a worked example.

Stanescu et al.[34] provide another example of a template-based system designed for a narrow usage scenario. Their Test Creator tool is designed to partially automate the process of creating tests from course materials. Using Test Creator, an instructor creates a set of

tags and templates. A tag can be thought of as a label for a category of questions. For example, a `<DEFINE>` tag might be created to group definition questions. For each of the tags, the instructor creates one or more question templates. A template for a definition question might be `Define #`. The `#` denotes an insertion point for source text. To generate questions, the instructor must load course material, manually highlight a portion of text, select a tag, and select a template. A question is then generated by replacing the # in the template with the highlighted text.

Whether this approach should be included under the umbrella of automatic question generation is a matter of how one defines "automatic." Clearly, much more human labour is required than in any of the approaches we have discussed.

An obvious strength of these template-based methods is their ability to generate questions with arbitrary wording. Consequently they are not in principle constrained to generating only certain types of questions, such as *who*, *what*, *when*, *where*, and so on. This would seem to have the potential to elevate us above factoid questions or at the very least, allow for questions that more easily paraphrase the original text. A template-based approach was certainly the ideal choice for Mostow and Chin's reading comprehension system [27], as templates are well-suited to generating meta-cognitive questions.

This flexibility comes at the price of needing extra human effort to make these systems work well. The systems that are designed to be domain-portable rely on syntax-based pattern matching. While this does allow matching constraints to be very specific, it can make templates difficult to write. More importantly, though, while this can prevent over-generation, it can also cause viable questions to be missed. As we saw in the example of *The boy went to school* vs. *John Smith went to school*, a slight difference in syntax will cause semantically-similar sentences to be treated differently, and a potentially viable question like *Why did John Smith go to school?* will be missed. Using existing template-based methods, we have to deal with this by having either multiple patterns or more complicated patterns. This problem becomes much more obvious when we consider cases in which a single semantic relationship can have two (or more) grossly different syntactic realizations. Such an example is presented in the next chapter to help motivate a key aspect of this thesis.

2.5 Evaluation

Our purpose in this section is to describe some of the issues and methodologies in the literature. Because corpora and methodologies have differed, we cannot present any kind of comparative evaluation. Refer to the individual papers cited for individual experimental details and results.

2.5.1 Methodologies

There remains no standard set of evaluation metrics for assessing the quality of QG output. Some present no evaluation at all ([38] and [34]). Among those who do perform an evaluation, there does appear to be a consensus that some form of human evaluation is needed. Despite this agreement in principle, approaches tend to diverge thereafter. There are differences in the evaluation criteria and the evaluation procedure.

Two common types of evaluation methodologies have been practiced. The first relies on direct human evaluation of system output. In some cases ([27] and [35]), people directly involved in the research evaluated their own system's output. More desirable, though, is to have impartial evaluators (as in [19] and [5]). While some have used a binary rating scheme in which a question is given a single acceptability rating ([5] and [27]), those with more interesting evaluation results have asked raters to evaluate more specific qualities of the questions, such as grammaticality, meaningfulness, answerability, vagueness, question word choice, and formatting ([19] and [35]).

The second common methodology relies on a comparative evaluation between system output (Q_g) and questions humans have generated (Q_h) from the same corpus. We might think of this as indirect human evaluation. Ali et al.[2] and Kalady et al.[20] are two approaches that have been evaluated using this methodology. For a given document or corpus, *precision* and *recall* can be calculated as shown in (2.43) and (2.44), where the human-generated questions are considered the gold-standard. Obviously, this type of evaluation is only possible if human-generated questions are already available or it is feasible to have humans generate questions.

$$precision = \frac{Q_g \cap Q_h}{Q_g} \tag{2.43}$$

$$recall = \frac{Q_g \cap Q_h}{Q_h} \tag{2.44}$$

Cai et al. conducted an evaluation that combined these two methodologies [5]. This evaluation involved human raters blindly comparing automatically-generated questions with questions written by human experts for the AutoTutor system [15]. Questions were presented to five students who were asked to identify each question as being either "good" or "bad."

2.5.2 Ranking

Relatively little attention has been paid to the ranking of automatically-generated questions. Heilman and Smith [19][17] and McConnell et al.[26] are among those who have paid attention. Heilman and Smith developed a logistic regression raking model that defined the probability of acceptability, $p(a|q,t)$, given a question (q) and the source text (t). This ranker was employed after a human evaluation in which questions were evaluated according to a set of possible deficiencies (see Table 2.3). Seven types of features are used: length, wh-word type, negation, language model, grammar, transformations, and vagueness.

Two ranking experiments were performed. In the first experiment, questions featuring any one of the deficiencies were considered unacceptable, and a single ranking model was trained. In the second experiment, separate models were trained over each of the possible deficiencies in Table 2.3, and these models were then combined using (2.45) to give a single estimated probability of acceptability.

$$p(a|q,t) = \prod_{i=1}^{K} p_i(a_i|q,t) \tag{2.45}$$

The variable i is an index over the K possible deficiencies.

To compare the two ranking methods, Heilman and Smith calculated precision-at-N scores for each method. These scores measure the percentage of questions judged acceptable by the human evaluators that also appeared in the top N ranked questions. Results showed that across multiple corpora, the second method significantly out-performed the first.

McConnell et al.[26] added additional ranking criteria to the system developed by Mannem et al.[25]. They add two ranking criteria: *information content* and *grammaticality*. These extra features were added to address the limitations of the ranking scheme of Mannem et al., which considered only predicate depth and pronoun counts. They used techniques borrowed from text summarization to identify topic words in the source text. Source sentences were scored according to the fraction of topic words they contained, which was

Deficiency	Description
Ungrammatical	The questions is not a valid English sentence
Does not make sense	The question is grammatical but meaningless.
Vague	The question is too vague to answer, even after reading the source text.
Obvious answer	The question can be answered without reading the source text.
Missing answer	The answer to the question is not contained in the source text
Wrong WH word	The question uses the wrong WH word (e.g., *what* instead of *who*).
Formatting	The question has capitalization and/or punctuation errors

Table 2.3: Possible question deficiencies identified by Heilman and Smith

considered a measure of their information content. They report, though only anecdotally, that questions with higher topic scores (i.e., higher information content) produced better questions.

To measure the grammaticality of generated questions, they used a bi-gram language model estimated from the Microsoft Encarta question database. They report, again anecdotally, that language model scores appeared to significantly improved ranking. They admit that a much larger study is needed to properly evaluate the benefit of adding language model scores.

2.5.3 Shared tasks

Several factors converge to make a comparative evaluation of QG systems difficult. First, as described above, methodologies have differed. A manual evaluation cannot be compared to an evaluation based on precision and recall. Second, even intra-methodology comparisons are tenuous at best. In the case of entirely manual evaluations, different systems have obviously used different evaluators, so comparison is not possible. In the case of precision/recall evaluations, the gold standard has been different, so again, we cannot compare results. Third, corpora used for evaluation have differed.

In recent years, proposals have been made for a QG shared task in order to facilitate comparative evaluation that might give us a better notion of what the current *state-of-the-art* is in QG from text. Nielsen has proposed decomposing QG into two smaller tasks, key concept identification and question construction, the first of which he believes is conducive to automatic evaluation methods [28]. This proposal does not appear to have gained significant support.

Another recent proposal is the Question Generation Shared Task Evaluation Challenge

(QGSTEC) proposed by Rus and Graesser [32]. The first QGSTEC was run in 2010 [33]. QGSTEC 2010 had a generation from paragraphs task (Task A) and a generation from single sentences task (Task B). Task A was the task in which Mannem et al.[25] participated, and theirs was the only submission to this task. In Task B, participants were given a set of inputs consisting of a single sentence and a target question type, and their task was to generate two questions of the desired question type. Human evaluators were used for both of these tasks. For Task A, the evaluation criteria were scope (rater-selected vs. participant-selected), grammaticality, semantic validity, question type correctness, and diversity. For Task B, the grammaticality and question type criteria were retained, and relevance, ambiguity, and variety criteria were added. We can find no evidence that this shared task continued after its first iteration, and it does not appear as though any new shared tasks have surfaced. However, there has been a recent call for NLG shared task proposals as part of the 14th European Workshop on Natural Language Generation (ENLG'13).[3]

2.6 General impressions

Previous work has focused on generating questions that are the result of transformations from the declarative to the interrogative. This makes these approaches very domain-portable. However, the not-so-covert supposition underlying these methods is that a question can only be asked after its answer has been found. This limits both the types of questions they can generate (*who*, *what*, *when*, *where*, etc.) and the scope of those questions. The template-based methods, which have the potential to generate much more interesting questions, are described (by their authors) using examples that produce results very similar to those of the syntax and semantics-based methods. Mostow and Chen's reading comprehension system [27] is an exception, but it was purpose-built for a very constrained interaction scenario and is not a general-purpose QG system.

It is easy to understand why these approaches have been so cautious in their scope. Attempting to generate questions by leveraging multiple sentences requires some understanding of the semantic relationships among those sentences, which is considerably more difficult than dealing with single sentences. Even single sentences can be challenging. Sentence simplification has been preferred, particularly by the syntax-based methods, because

[3] http://www.um.edu.mt/events/enlg2013/

it reduces the syntactic variation QG systems must account for when identifying targets and formulating questions.

In terms of evaluation, direct human judgment of automatically-generated questions has been preferred. This is as it should be, especially as we attempt to move beyond the shallow factoid questions most systems have been designed to produce.

Chapter 3

Our Approach

We develop a template-based framework for QG. The primary motivation for this decision is the ability of a template-based approach to generate questions that are not merely declarative to interrogative transformations. We aim to address some of the limitations of the existing approaches outlined in the previous chapter while leveraging some of their strengths in novel ways. We combine the benefits of a semantics-based approach, the most important of which is not being tightly-constrained by syntax, with the surface-form flexibility of a template-based approach. The novel idea explored in this thesis is semantics-based templates that use SRL in conjunction with generic and domain-specific scope.

3.1 Semantics-based templates

As seen in Chapter 2, previous template-based methods have used syntactic pattern matching, which does provide a great deal of flexibility in specifying sentences appropriate for generating certain types of questions. However, this flexibility comes at the expense of generality. As seen in Wyse and Piwek [38], the specificity of syntactic patterns can make it difficult to specify a syntactic pattern of the desired scope. Furthermore, semantically similar entities can span different syntactic structures, and matching these requires either multiple patterns (in the case of [5]) or a more complicated pattern (in the case of [38]).

3.1.1 Semantic patterns

If we want to develop templates that are semantically motivated, more flexible in terms of the content they successfully match, and more approachable for non-technical users, we need to move away from syntactic pattern matching. Instead, we match *semantic patterns*, which we define in terms of the SRL parse of a sentence and the named entities (if any) contained within the span of each semantic role. From this point forward, we adopt the shorter CoNLL SRL shared task naming conventions for role labels [7] (e.g., `V` (for predicates), `A0` and `AM-LOC`). We use Stanford NER [14] for named entity recognition. A sentence will have one semantic pattern for each of its predicates. Figure 3.1 shows a sentence and its corresponding semantic patterns. On the left-hand side of the figure, we see named entity annotations for the sentence. On the right hand side, we see two predicates and their semantic arguments. Each of these predicate-argument structures is a distinct *predicate frame* having a specific semantic pattern. The first predicate frame, belonging to the predicate *was*, has a semantic pattern described by an `AM-TMP` containing a `DATE`, an `A1` containing a `LOCATION`, and an `AM-MNR` containing an entity of type `MISC`. The second predicate frame, associated with the predicate *referred*, has a semantic pattern described by an `A1` containing no named entities and an `A2` containing an entity of type `MISC`.

By combining SRL with NER, we are able to create semantic patterns that enable us to ask more specific questions than SRL alone would allow. For example, this allows us to distinguish between an `AM-TMP` containing a `DATE` and an `AM-TMP` containing a `DURATION`. The latter case, it is more appropriate to ask a *how long* question rather than a *when* question.

Even the shallow semantic analysis of SRL is sufficient to identify the semantically-interesting portions of a sentence, and these semantically-meaningful substrings can span a range of syntactic patterns. Figure 3.2 shows a clear example of this phenomenon. In this example, we see two sentences expressing the same idea, namely, the fact that trapped heat causes the Earth's temperature to increase. In one case, this causation is expressed in an adjective phrase, while the other uses a sentence-initial prepositional phrase. The syntactic parse trees are generated using the Stanford Parser [21], and the SRL parses are generated using SENNA [11]. The `AM-CAU` semantic role in the SRL parse captures the cause in both sentences. It is impossible to accomplish the same feat with a single NLGML

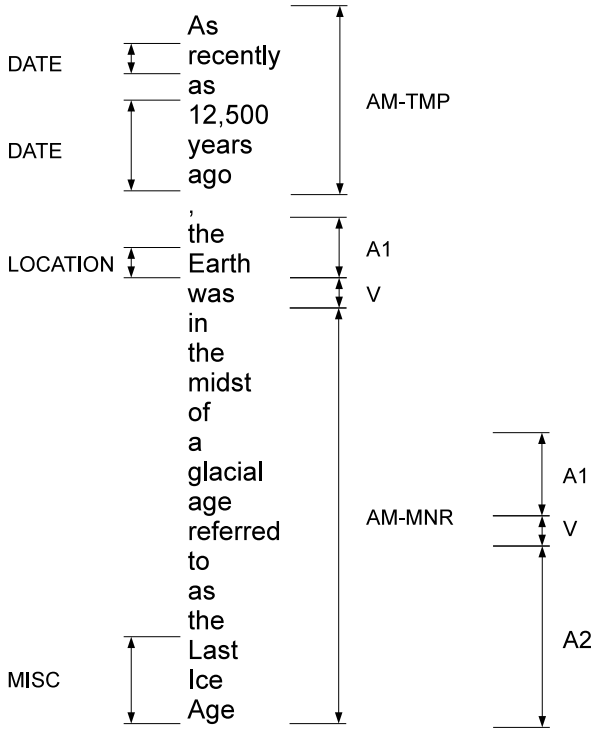

Figure 3.1: A sentence and its semantic patterns.

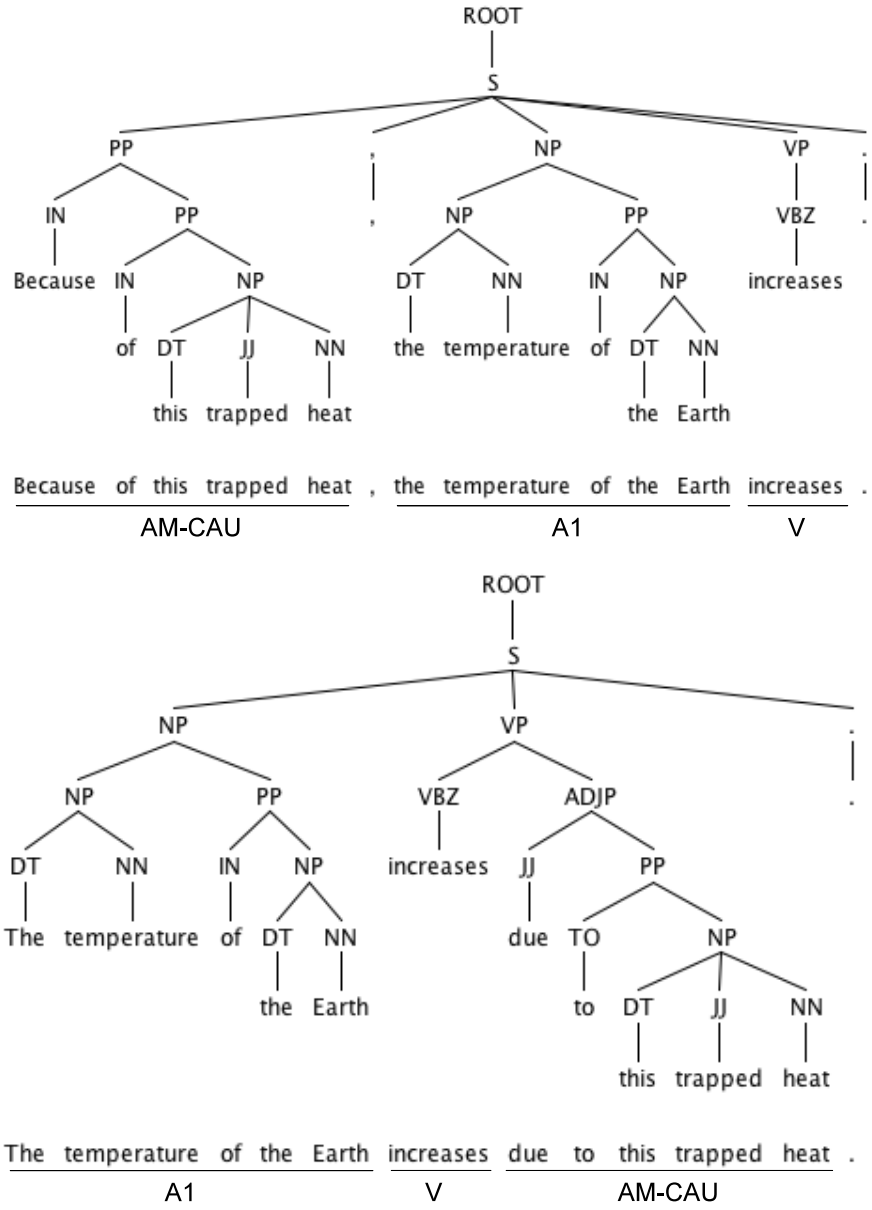

Figure 3.2: Two different syntax subtrees subsumed by a single semantic role

pattern. However, it is possible to capture both with the single Tregex pattern seen in (3.1).

```
S [[<<, PP=cause1] | [< (VP < ADJP=cause2)]]
```
(3.1)

This pattern matches any sentence that has either an initial prepositional phrase (assigned label `cause1`) or an adjective phrase (assigned label `cause2`) dominated by the main verb phrase. However, this pattern casts a wide syntactic net in the hopes of capturing a specific semantic relationship. It is possible to add additional restrictions to prevent the capture of non-causal relationships, but even in its current form, it is an eyesore.

The principle advantage of semantic pattern matching is that a single semantic pattern casts a narrow semantic net while casting a large syntactic net. A pattern defined in terms of semantic roles is not constrained by syntax. This means fewer patterns need to be defined by the template author, and the patterns are more compact.

3.1.2 Template specification

Our templates have three components: a question string, slots, and slot options. A question string forms the skeleton into which source text is inserted to create a question. Slots facilitate sentence and template matching. They accept specific semantic arguments, and can appear inside or outside the question string. These provide the semantic pattern against which a source sentence predicate frame is matched. A slot inside the question string acts as a variable to be replaced by the corresponding semantic role text from a matching predicate frame, while any slots appearing outside the question string serve only to provide additional pattern matching criteria. The template seen in (3.2) illustrates this idea. In this example, the question string is *What is one consequence of [A0]?* This template has an `A0` slot and an `A1` slot. The `A0` slot appears inside the question string, while the `A1` slot appears outside. The `##` characters denote the end of the question string. The `A0` and `A1` slots define this template's semantic pattern, which will match any predicate frame containing an `A0` and an `A1`. A question is created when the text of the predicate frame's `A0` is inserted into the question string. The text of the `A1` is not used, because that slot appears outside the template's question string. Often, slots appearing outside the question string will refer to a role that holds the answer to the question to be generated, but this is not universally true, because multiple slots can appear outside the question string, slots can serve to simply negate the presence of a particular role, or there may be no slots outside the question string

at all.

```
What is one consequence of [A0]? ## [A1]
```
(3.2)

The template author does not need to specify the complete semantic pattern in each template. Instead, only the portions relevant to the desired question need to be specified. This is an important point of contrast between a template-based approach such as ours and syntax and semantics-based approaches, because we can choose to generate questions that do not include any predicates from the source sentence but instead ask more abstract or general questions about other semantic constituents. We believe these kinds of questions are better able to escape the realm of the factoid. Whether or not this proves to be true is investigated in the next chapter.

Slot options are of two types: modifiers and filters. Modifiers apply transformations to the role text inserted into a slot, and filters enforce finer-grained matching criteria. Predicate slots have their own distinct set of options, while the other semantic roles share a common set of options. A template's slots and filters describe the necessary conditions for the template to be matched with a source sentence semantic pattern. Filters and modifiers are provided as tools for the template author to use at his or her discretion. We provide no automated methods for identifying when a given filter or modifier should be used.

Predicate slot options

The predicate filter options restrict the predicates that can match a predicate slot. With no filter options specified, any predicate is considered a match. Table 3.1 shows the complete list of filters.

Filter	Description
be	predicate must be a verb whose lemma is "be"
!be	predicate must not be a verb whose lemma is "be"
!have	predicate must not be a verb whose lemma is "have"

Table 3.1: Predicate filters

The selection of predicate filters might at first seem oddly limited. Failing to consider the functional differences between various types of verbs (particularly auxiliary and copula)

would indeed produce low-quality questions and should in fact be ignored in most cases. For example, consider the sentence in (3.3)

Flowering plants remain dormant. (3.3)

The lone predicate in this sentence is the copula *remain*. Some questions we might generate using this predicate, such as (3.4), are not particularly useful.

What do flowering plants remain? (3.4)

Fortunately, the copula verbs are not associated with an *agent*, so they do not have an `A0`. This fact can be used to recognize copula verbs by their surrounding semantic pattern, so in the broad sense, we do not need to adopt any copula-specific rules. The method for identifying roles missing from a semantic pattern will be discussed in the next section.

The one exception to the above rule is any copula verb whose lemma is *be*. The `be` and `!be` filters allow the presence or absence of such a predicate to be detected. This capability is useful for two reasons. First, the presence of such a predicate gives us an inexpensive way to generate definition questions, even if the source text is not written in the form of a definition. Although this will over-generate definition questions, non-predicate filters can be used to add additional mitigating constraints. Second, requiring the absence of such a predicate allows us to actively avoid generating certain kinds of ungrammatical questions. Whether using one of these predicates results in ungrammatical questions depends on the wording of the underlying template, so we provide the `!be` filter for the template author to use as needed.

Like copula verbs, auxiliary verbs are often not suitable for question generation. Fortunately, many auxiliary verbs are also modal and are assigned the label `AM-MOD` and so do not form predicate frames of their own. Instead, they are included in the frame of the predicate they modify. In other cases auxiliary verbs are not modal, such as in (3.5).

So far, scientists have not been able to predict the long term effects of this wobble. (3.5)

In this case, the auxiliary *have* is treated as a separate predicate, but importantly, the span of its `A1` includes the predicate *been*. We provide a non-predicate filter to prevent generation when this overlap is present.

The `!have` filter is motivated by the observation that the predicate *have* can appear as a full, non-copula verb (with an `A0` and `A1`) but often does not yield high-quality questions.

Modifier	Tense	Modifier	Person
lemma	lemma (dictionary form)	fps	first person singular
pres	present	sps	second person singular
prespart	present participle	tps	third person singular
past	past	fpp	first person plural
pastpart	past participle	spp	second person plural
perf	perfect	tpp	third person plural
pastperf	past perfect		
pastperfpart	past perfect participle		

Table 3.2: Predicate modifiers

Predicate modifiers allow the template author to explicitly force a change in conjugation. See Table 3.2 for the complete set of predicate modifiers. The `lemma` modifier can appear on its own. However, all other conjugation changes must specify both a *tense* and a *person*. If no modifiers are used, the predicate is copied as-is from the source sentence. Although *perfect* is an aspect rather than a tense, MorphAdorner[1], which we use to conjugate predicates, defines it as a tense, so we have implemented it as a tense filter.

Non-predicate slot options

The filters for non-predicate slots impose additional syntactic and named entity restrictions on any matching role text. For slots appearing outside the template's question string, a `null` filter can be applied which explicitly requires that the corresponding semantic role not be present in a source sentence semantic pattern. As with predicate filters, the absence of any non-predicate filters results in the mere presence of the corresponding semantic role being sufficient for matching. See Table 3.3 for the complete list of non-predicate filters.

The choice of filters again requires some explanation. The `null` and `!nv` filters were foreshadowed above. For slots appearing outside the template's question string, the `null` filter explicitly requires that the corresponding semantic role not be present in a source sentence semantic pattern. An `A0` slot paired with the `null` filter is the mechanism alluded to earlier that allows for the recognition of copula verbs without the need to examine the predicate itself. The `!nv` filter can be used to prevent ungrammatical questions. We observe

[1] http://morphadorner.northwestern.edu

Filter	Description
null	The predicate frame must not contain a semantic role of this slot's type.
!nv	The span of the role text must not contain a predicate.
dur	The role text must contain a named entity of type DURATION.
date	The tole text must contain a named entity of type DATE.
!date	The role text must not contain a named entity of type DATE.
loc	The role text must contain a named entity of type LOCATION.
ne	The role text must contain a named entity. Any type suffices.
misc	The role text must contain a named entity of type MISC
comp	The role text must contain a comparison.
!comma	The role text must not contain a comma.
singular	The role text must contain a singular noun.
plural	The role text must contain a plural noun.

Table 3.3: Non-predicate filters

that if a role span does include a predicate, resulting questions are often ungrammatical due to the conjugation of that predicate. Applying this filter to the `A1` of a predicate prevents generation from a predicate frame whose predicate is a non-modal auxiliary verb.

The named entity filters (`dur`, `!dur`, `date`, `loc`, `ne`, and `misc`) are those most relevant to the corpus we have used to evaluate our approach and thus the easiest to experiment with effectively. Because named entities are used only for filtering, expanding the set of named entity filters is a trivial task.

The filters `comp`, `!comma`, `singular`, and `plural` are syntax-based filters. With the exception of `!comma`, these filters force the examination of the part-of-speech (POS) tags to detect the desired features. The `singular` and `plural` filters let templates be tailored to singular and plural arguments in any desired way, beyond simply selecting appropriate auxiliary verbs. The type of comparison we search for when the `comp` filter is used is quite specific. We search for phrases that describe conditions that are atypical. These can be seen in phrases such "unusually weak," "unseasonably warm," "strangely absent," and so on. These phrases are present when a word whose POS tag is `RB` (adverb) is followed by a word whose tag is `JJ` (adjective). Detecting these can allow us to generate questions such as those seen in (3.6) and (3.7).

What data would indicate X? (3.6)

How do the conditions that cause X differ from normal conditions? (3.7)

Here, *X* would be replaced by an appropriate semantic argument, most likely an `A1`. Although this heuristic does produce both false positives and false negatives, other syntactic features such as comparative adverbs and comparative adjectives are less semantically constrained.

We see two situations in which a comma appears within the span of a single semantic role. The first situation occurs when a list of nouns is serving the role, such as in (3.8).

Climate change includes changes in precipitation, temperature, and pressure. (3.8)

The noun phrase *changes in precipitation, temperature, and pressure* is the `A1` of the predicate *includes*. In cases where a question is only appropriate for single concept (e.g. temperature) rather than a set of concepts, the `!comma` filter prevents such a question from being generated from the sentence above. This has implications for role text containing appositives, the second situation in which a comma appears within a single role span. Such roles are rejected when `!comma` is used. This is not ideal, as removing appositives does not cause semantic roles to be lost from a semantic pattern. Future work will address this problem.

The non-predicate modifiers (Table 3.4) serve two purposes: to create more fluent questions and to remove non-essential text. Note that the `-tpp`, which forces the removal of trailing prepositional phrases, can have undesired results when applied to certain modifier roles, such as `AM-LOC`, `AM-MNR`, and `AM-TMP`, when they appear in the template question string. These modifiers often contain only a prepositional phrase, and in such cases, `-tpp` will result in an empty string being placed into the template.

The non-predicate modifiers (Table 3.4) serve two purposes: to create more fluent questions and to remove non-essential text.

Modifier	**Effect**
-lpp	If the initial token is a preposition, it is removed.
-tpp	If the role text ends with a prepositional phrase, that phrase is removed
-ldt	If the initial token is a determiner, it is removed.

Table 3.4: Non-predicate modifiers

Among the template-based approaches discussed in the previous chapter, NLGML [5] is the most similar to our own. Our question strings are very similar to NLGML question

templates, because they guide the task of surface-form realization. Our slots are very similar to NLGML variables, because they define insertion points for source text. Our slot options function in much the same way as NLGML functions, because they modify source sentence text. A primary difference that distinguishes our work from NLGML is that we use semantic rather than syntactic pattern matching. We also use a more compact representation that uses familiar operators such as "!" for logical negation and "-" for deletion.

3.1.3 Our QG system

Figure 3.3 shows the architecture and data flow of our QG system. One of the most important things to observe about this architecture is that the templates are an external input. They are in no way coupled to the system and can be modified as needed without any system modifications.

Compared to most other approaches, we perform very little pre-processing. Syntax-based methods in particular have been motivated to perform sentence simplification, because their methods are more likely to generate meaningful questions from short, succinct sentences. We have chosen not to perform any sentence simplification. This decision was motivated by the observation that common methods of sentence simplification can eliminate useful semantic content. For example, Kalady et al.[20] claim that prepositional phrases are often not fundamental to the meaning of a sentence, so they remove them when simplifying a sentence. However, as Figure 3.4 shows, a prepositional phrase can contain important semantic information. In that example, removing the prepositional phrase causes temporal information to be lost.

One pre-processing step we do perform is pronominal anaphora resolution (using [9]). Even though we do not split complex sentences and therefore do not create new sentences in which pronouns are separated from their antecedents, this kind of anaphora resolution remains an important step in limiting the number of vague questions.

Each source sentence is tokenized and annotated with POS tags, named entities, lemmata, and its SRL parse. We generate the SRL parse [11] in order to extract a set of predicate frames. Questions are generated from individual predicate frames rather than entire sentences (unless the sentence contains only one predicate frame). Given a sentence, the semantic pattern of each of its predicate frames is compared against that of each template. Algorithm 1 describes the process of matching a single predicate frame (pf) to a single template (t). For each slot in the template's semantic pattern, we check for the corresponding

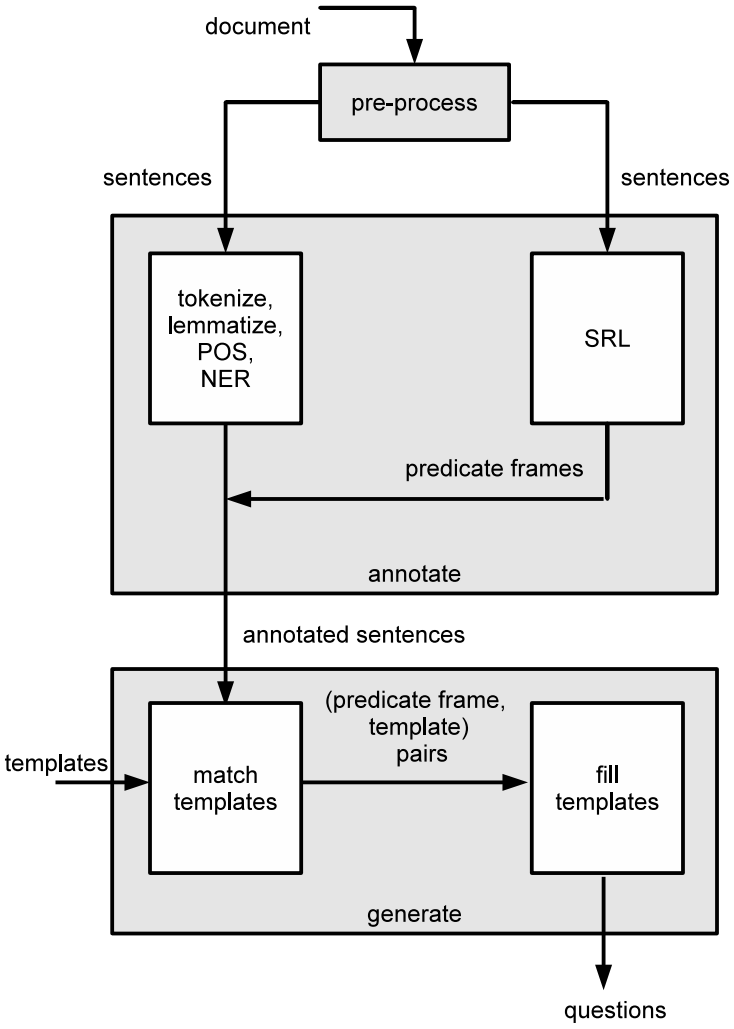

Figure 3.3: System architecture and data flow

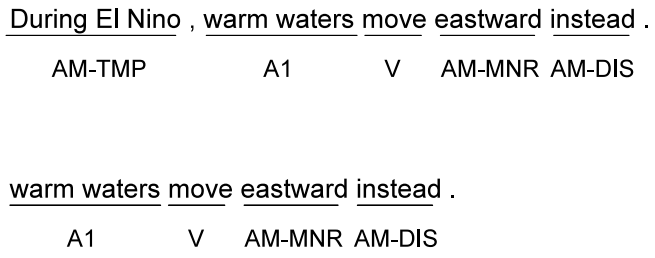

Figure 3.4: Semantic information can be lost during sentence simplification. Removing the prepositional phrase from the first sentence leaves the simpler second sentence, but the `AM-TMP` modifier is lost.

semantic role in the predicate frame. If it is not present in the predicate frame, we check if the template slot uses the `null` filter. If the predicate frame does have the necessary semantic role, we proceed to verify that the template slot's filters are also satisfied. Although it is not stated in Algorithm 1, the sentence-level tokenization, lemmata, named entities and POS tags are checked as needed according to the template's slot filters. If a predicate frame and template are matched, they are passed to Algorithm 2, which fills template slots with role text to produce a question. For each slot in the template's question string, the corresponding role text from the predicate frame is extracted, modified as dictated by the slot's modifiers, and inserted into the question string. Even in the absence of modifiers, all role text receives some additional processing before being inserted into its corresponding slot. These additional steps include the removal of colons and the things they introduce and the removal of text contained in parentheses. We observe that these extra steps lead to questions that are more meaningful. Overlapping predicate frames might result in duplicate questions. To avoid duplicates we keep only the first occurrence of a question.

Using slots and filters, we can now create some interesting templates and see the questions they yield. Table 3.5 shows some templates ($\mathbf{T}$) that match the sentence in Figure 3.1 and the questions ($\mathbf{Q}$) that result. This example shows a mixture of general and domain-specific templates. The second template has a question string containing the words *on the environment.* This is an example of a template tailored to the global warming domain.

Algorithm 1 patternMatch(pf,t)

```
# pf is a source sentence predicate frame
# t is a template
# we attempt to match the semantic pattern of pf with that of t
for all slot ∈ t do
    if pf does not have a slot of type slot.type then
        if null ∉ slot.filters then
            return false
        end if
    else
        for all filter ∈ slot.filters do
            if pf.role does not match filter then
                return false
            end if
        end for
    end if
end for
return true
```

Algorithm 2 templateFill(pf,t)

```
# pf is a source sentence predicate frame
# t is a template
# pf's semantic pattern matches that of t
question_text ← t.question_string
for all slot ∈ t.question_string.slots do
    role_text ← text of the slot.type role from pf
    for all modifier ∈ slot.modifiers do
        applyModifier(role_text,modifier)
    end for
    In question_text, replace slot with role_text
end for
return question_text
```

A template's question string is what makes it general or domain-specific. All three of the templates produce questions not achievable by merely rearranging the surface form of the original sentence and replacing a particular phrase with a question word. It should also be noticed that despite the relative length and complexity of the original sentence, we can generate syntactically and semantically well-formed questions. Although the questions that are generated are not answerable from the original sentence, they were judged answerable from the source document in our evaluation. That is, answering the questions requires the reader to refer to other parts of the document. In other words, the questions are not strictly factoids. The ability to generate questions that require the learner to consult other parts of the text is due to the flexibility of the templates. The template author can write templates that produce questions that are deliberately broader in scope than mere factoids.

As recently as 12,500 years ago, the Earth was in the midst of a glacial age referred to as the Last Ice Age.
T: How would you describe [A2 -lpp misc]? **Q**: How would you describe the Last Ice Age?
T: Summarize the influence of [A1 -lp !comma !nv] on the environment. **Q**: Summarize the influence of a glacial age on the environment.
T: What caused [A2 -lpp !nv misc]? ## [A0 null] **Q**: What caused the Last Ice Age?

Table 3.5: A few sample templates and questions

Although semantic patterns allow us to constrain question generation, they do not alleviate the problem of over-generation. Even if we are able to generate questions that are grammatical and meaningful (see Chapter 4), we are still faced with the fact that we generate a great many questions for a given document, far more than a learner would be inclined to answer. An evaluation of our system's output and how we deal with the resulting abundance of questions is the subject of the next chapter.

Chapter 4

Evaluation

4.1 Corpus

We evaluated our system using a corpus of 25 documents specifically designed to model a Grade 10 Science curriculum on climate change and global warming, with discourse and vocabulary appropriate for that age group.[1] The corpus contained 565 sentences and approximately 9000 words. These documents describe natural and man-made influences on climate. Included with the corpus was a glossary of key terms in the climate science domain. The glossary was used only for evaluation purposes and did not inform the question generation algorithm in any way. We selected a ten-document subset of this corpus to use for evaluation, and the remaining documents were used to develop templates. The ten documents contained 246 sentences and approximately 4000 words. The average number of sentences per document in the evaluation set was 24.6, with the smallest document having 15 sentences and the largest having 54. Figure 4.1 shows a sample paragraph from one of the documents.

4.2 Methodology and motivation

We conducted a three-part evaluation. First, we asked a human judge to evaluate questions generated by our system. Second, we examined the performance of each of our templates

[1]This corpus was prepared by Kiran Bisra, a graduate student in the Faculty of Education at Simon Fraser University.

> Over the course of Earth's history, the climate has naturally changed many times. The temperature has cycled at various points, causing the Earth to be cooler and warmer than today's climate. As recently as 12,500 years ago, the Earth was in the midst of a glacial age referred to as the Last Ice Age. Canada and most of the northern United States, including New York City, was buried underneath ice sheets. It began 110,000 years ago when the Earth's orbit shifted causing a decrease in the amount of sunshine hitting the Earth. Gradually, the precipitation turned into snow and over hundreds of years, turned into sheets of ice. This global cooling disrupted the carbon cycle, causing a greater portion of carbon dioxide to be taken up by the oceans and stored in the frozen ground. As more and more of the carbon dioxide was removed from the atmosphere, the cooler the Earth became. But, then the Earth's orbit shifted naturally shifted again, allowing for more sunlight to fall on the Northern Hemisphere. The ice melted and started a chain reaction which has led to today's warmer climate.

Figure 4.1: A paragraph from our evaluation corpus

in the hope that this might provide insights into what makes a good template. Third, we used the results of the human evaluation to develop an automated ranking mechanism.

4.2.1 Evaluating questions

We agree in principle that human evaluation is indeed necessary for a proper evaluation of QG output. As discussed in chapter 2, an evaluation scheme involving human judges requires careful selection of first, the human judge(s), and second, the specific evaluation criteria. We examine these issues here in reverse order, because in our case, the latter directly informed the former. We wanted our evaluation to examine several dimensions. We looked at whether each question was grammatical, made sense, was vague, was answerable from the sentence from which it was generated, was answerable from the source document. and had learning value. Each of these facets was to be given a simple *yes* or *no* response from the evaluator. The first two criteria were intended to give us some way to measure the performance of our approach in terms of generating valid natural language output. The remaining criteria were intended to enable us to examine the questions from a more educationally-oriented perspective.

Had we evaluated the output of our own system as some others have done ([27] and [35]), the results would have been quite flattering, as we will discuss when we examine the evaluation results. To avoid bias, an impartial evaluator was chosen from the Faculty

of Education[2] at Simon Fraser University. Our evaluation scheme required the evaluator to have a thorough understanding of the content and educational goals of the evaluation corpus. Fortunately, we had access to the author of the corpus, who was otherwise not involved in this research.

4.2.2 Evaluating templates

We would be remiss in not evaluating the templates themselves in some way. We can gain some valuable insights that can inform the template writing process by determining how often each template was used to generate a question and how those questions fared in the manual evaluation. Is there any correlation between the number of questions a given template produces and the quality of those questions? Which templates tend to produce well-rated questions? Which templates tend to produce poorly-rated questions? These are some of the questions we attempt to answer.

4.2.3 Ranking

As we have already discussed, most previous efforts in QG have stopped at this point, performing a manual evaluation and discussing the results but not using those results to do any form of automated evaluation. While some have gone a step further and built rankers that try to estimate the relative *acceptability* of a question [19], the distinction between *acceptable* and *unacceptable* questions has not had a strong basis in pedagogy. The number of questions that are deemed acceptable can easily be much larger than the number of questions that we can reasonably expect a student would want or have time to answer. A ranking mechanism based merely on acceptability is not necessarily likely to score an educationally useful question higher than a less useful question if both are otherwise *acceptable*. Simply put, an acceptable question might not be an educationally useful question, and we want a ranking that attempts to give higher scores to the most educationally useful questions.

Initially, we experimented with binary classification rather than ranking. The results of those experiments are presented in [24]. Given the educational purpose of the questions being generated, and the difficulty in classifying the educationally useful questions with high precision and recall, ranking is more appropriate. The ranked questions will be one of the inputs to a selection algorithm. This selection algorithm must consider constraints not

[2] http://www.educ.sfu.ca

intrinsic to the task of generating questions and is thus outside the scope of this thesis. We will elaborate in our discussion of future work in the next chapter.

4.3 Evaluation before ranking

4.3.1 Question evaluation

Using 52 different templates, we generated 1472 questions from the ten documents in our corpus, an average of 5.9 questions per sentence. See Appendix A for a complete listing of the templates used. The first step in our evaluation was the human evaluation described above. Table 4.1 shows the percentage of questions that received a *yes* response for each of the evaluation criteria.

category	%
grammatical	85
makes sense	63
vague	78
answerable from sentence	14
answerable from document	20
learning value	17

Table 4.1: Evaluation results for 1472 questions generated from 10 documents

The first thing we noticed in these results was the high percentage of grammatical and semantically meaningful questions. Intuitively, it certainly sounds somewhat risky to extract a priori unknown pieces of a sentence and insert them into pre-defined templates whose syntax and semantics may or may not be compatible, but as we have demonstrated, with a few very simple filters and modifiers, we can generate many grammatical and semantically valid questions. As mentioned earlier, had we evaluated these questions ourselves, the results would have been more impressive. The percentage for vagueness would have been lower and the percentages for both answerability criteria and for learning value would have been higher.

The percentage for each of the evaluation criteria is interesting on its own, but the intersections among these criteria are more interesting. As we should expect, there is very

little overlap between questions that had learning value and those that were either ungrammatical or did not make sense. Only 15 questions were identified as having learning value despite also being either ungrammatical or not making sense. It appears that the evaluator was able to see the elements of learning value in these questions despite the noise caused by poor grammar and semantics. This is merely speculation on our part.

The relationship between vagueness and learning value is more interesting and perhaps a little surprising. While others have treated vagueness as nothing but a deficiency [19], our evaluator identified 28 questions as having learning value despite being vague. Although these questions represent less than 2% of the evaluation set, the fact that they exist at all tells us that the common assumption that vague questions are not worthwhile questions should be re-examined.

The most revealing relationships are those between the two types of answerability and their relationship to learning value. Although we generate questions from single sentences, more questions were answerable after reading an entire document than were answerable after reading only the sentence from which they were generated. There were also more questions with learning value than those that were answerable from a single sentence. Furthermore, 88 questions were identified as having learning value but deemed answerable only after reading other parts of the source document. This represents more than one-third of all questions with learning value.

4.3.2 Template evaluation

We developed each of the 52 templates used for this evaluation. Some were inspired by a small set of sample questions provided by educators for two of the documents in our development set, but most were entirely our own creation. The templates were written to explore various ways of asking questions using a diverse set of semantic patterns. Templates using very simple semantic patterns were naturally very simple and fast to write, while more complicated templates and templates using semantic roles with more ambiguous meaning took more effort.

What can we learn by looking at how each template was used by our system? Examining the template coverage of the questions might allow us to identify the qualities of templates that produced many questions or very few questions and more importantly, which templates produced many questions that have learning value. As Figure 4.2 shows, a handful of templates were clearly dominant. In fact, just five of the 52 templates accounted for 58%

of the questions generated. These templates were

(21.5%) `Summarize the influence of [A1 -lpp !comma !nv] on the environment.` (4.1)

(10.9%) `How can [A0 -lpp !nv] [V !be !have lemma] [A1]? ## [A2 null]` (4.2)

(10.8%) `What can [A0 -lpp !nv] [V !be !have lemma]? ## [A1] [A2 null]` (4.3)

(8%) `What [V !be !have pres tps] [A1 !nv]? ## [A2 null] [A0]` (4.4)

(6.8%) `What [V be] [A1 -lpp !nv !comma]?` (4.5)

These five templates account for such a large portion of the questions for the simple reason that they have very simple semantic patterns that consider only the most frequently seen semantic roles and also do not use any named entity filters. The least restrictive of these templates is (4.2), which also happens to be one of the domain-specific templates. Is there value in having domain-dependent templates with such easily-matched semantic patterns? Our evaluation of how templates covered the questions with learning value will answer this question.

We should not ignore the fact that six templates failed to generate any questions at all. These templates were

`Do the conditions that cause [A1 !comma !nv] differ from normal conditions? ## [AM_TMP comp]` (4.6)

`What data would indicate a(n) [A1 !comma !nv] event ? ## [AM_TMP comp]` (4.7)

`How do [A1 -lpp plural] compare to [A0 -lpp]? ## [AM_EXT]` (4.8)

`Which [A0 -ldt] [V !be !have prespart tps] [A1] [A2]? ## [R_A0] [AM_LOC null]` (4.9)

`Which [A0 -ldt] [V !be !have prespart tps] [A1] [AM_LOC]? ## [R_A0]` (4.10)

`Which [A0 -ldt] [V !be !have prespart tps] [A2]? ## [R_A0]` (4.11)

In contrast to the most often used templates, these six templates specify very specific or rare semantic patterns, patterns completely absent in the documents used for evaluation. These templates were developed using several documents from the original corpus that were

not included in the set used in our evaluation. Had it found a matching predicate frame, template (4.8) would have produced a question calling on the learner to make a comparison, an activity distinctly more sophisticated than level 1 in the revised Bloom taxonomy [3]. The importance of generating such questions and the apparent difficulty of doing so using a template such as (4.8) means template authors should strive to find ways to generate such questions using semantic patterns that do not contain the rare `AM-EXT` semantic role.

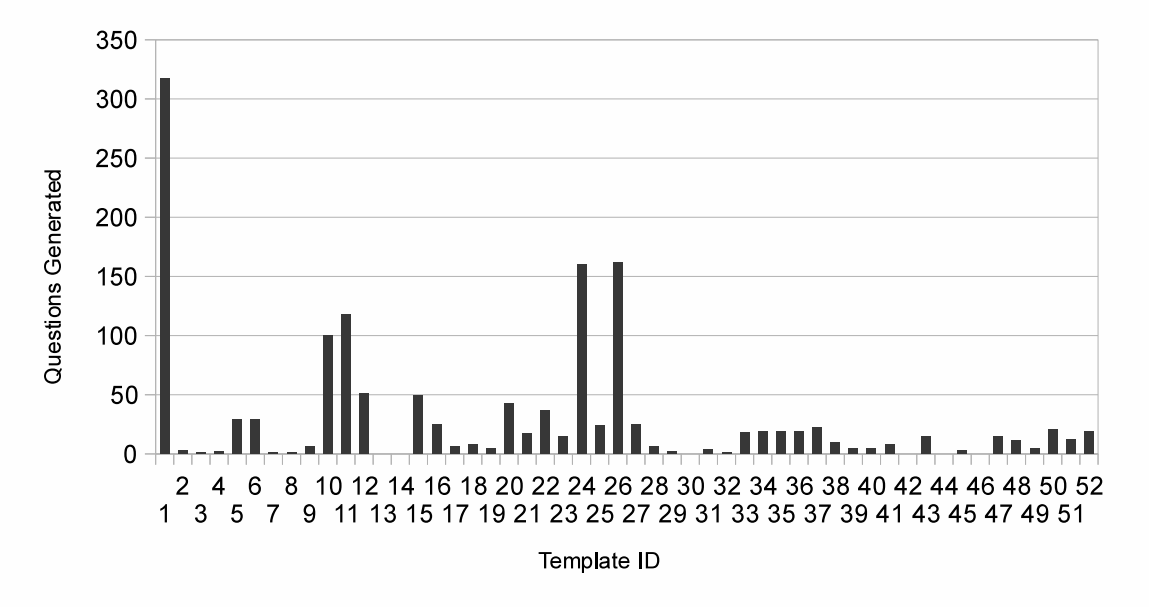

Figure 4.2: The number of questions generated using each template

We need to consider how the frequency of use of a template compares to its frequency in producing questions with learning value. While we could examine each template's contribution to the questions satisfying the other evaluation criteria, our focus has been on generating questions with learning value. Ideally, we would like to see the templates that produce the most questions also produce the most questions with learning value. Otherwise, those templates are adding a disproportionate share of the less useful questions. Figure 4.3 shows the number of questions with learning value generated by each template. As we had hoped, the top five templates in terms of questions produced also happen to be the top five

in producing questions with learning value. On average, 23% of the questions generated by a template proved to have learning value. The variance was 0.08.

The template in 4.2, which corresponds to template ID 1 in Figures 4.2 and 4.3 is an example of a domain-specific template with a very simple, easily-matched semantic pattern. It generated more questions than any other template, but it also generated the third-highest number of questions with learning value. To answer our earlier question, such templates are in fact valuable.

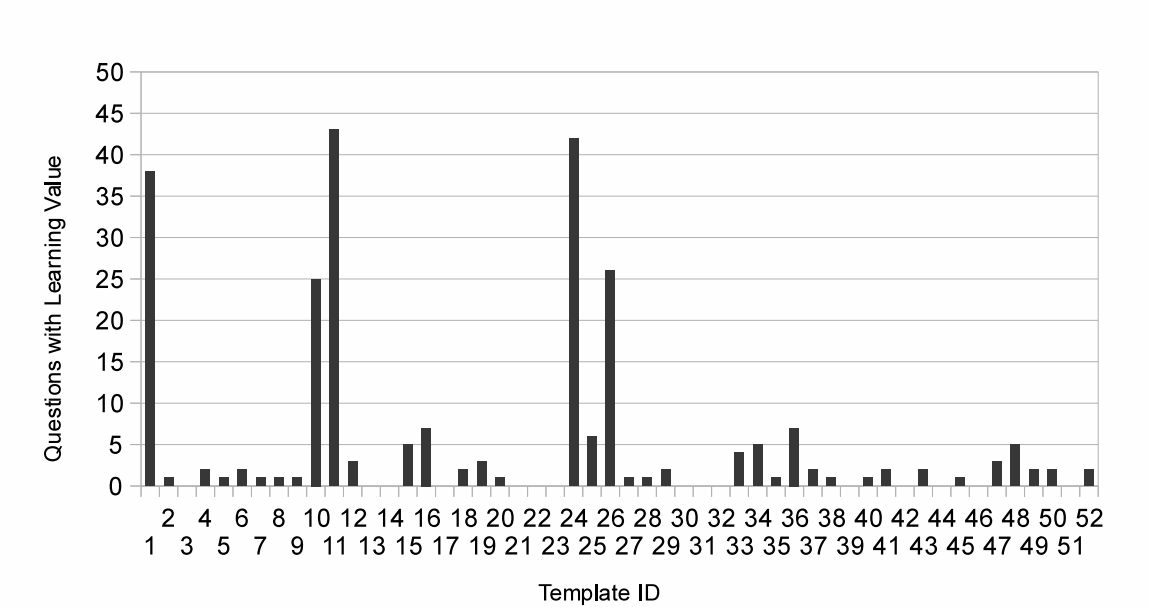

Figure 4.3: The number of questions having learning value generated using each template

Although a small subset of the templates generated the majority of the questions and the majority of the questions having learning value, we also observed a small subset of templates that produced very few questions but also produced questions with learning value 100% of the time. These templates were

```
How would you describe [A2 -lpp misc]?                                    (4.12)
```

```
How would you describe the cause of [A2 -lpp !nv misc]? ## [A0 null]      (4.13)
```

```
What caused [A2 -lpp !nv misc] ? ## [A0 null]
```
(4.14)

```
How does [A1 -lpp singular] compare to [A0 -lpp] ? ## [AM_EXT]
```
(4.15)

These would seem to be ideal templates, because they generate very few questions but generate questions with learning value. It appears as though requiring a `MISC` entity in an `A2` role in (4.12), (4.13), and (4.14) is the primary reason so few questions were generated using those templates. An entity of type `MISC` rarely appeared in an `A2` role, but when it did, it was something worth asking about. Notice that (4.15) differs from (4.8) only in that it uses the `singular` filter on the `A1` slot rather than `plural`.

4.3.3 Evaluator observations

In a post-evaluation interview, our evaluator provided several valuable insights into the performance of our system. Her first set of observations concerned the types of questions being generated. She clearly identified not only which types of questions tended to have more value but also why she felt certain types were better than others. *How* and *when* questions were generally of lesser quality in her opinion due to the writing style she used in the corpus. On the other hand, she felt *summarize*, *describe* and definition questions worked very well, because they suited the learning objectives she had in mind when writing the corpus. This qualitative assessment is corroborated by the quantitative evaluation of the template coverage of questions with learning value. The evaluator went on to say that the corpus could be rewritten to produce better questions. We obviously do not want a system that needs to have its input rewritten in order to produce high-quality questions. The much easier thing to do is to change the templates. Because they are defined outside the system, templates can easily be added, removed, or modified as needed to suit a particular domain, style of discourse, and set of desired learning outcomes.

The evaluator did make one observation that forces us to re-examine one of our assumptions. We mentioned in the previous chapter that we do not perform any sentence simplification, because semantic information can be lost in the process. However, the evaluator commented that shorter sentences seemed to yield better questions. A further study would be needed to examine the effect of adding sentence simplification.

4.4 Ranking

4.4.1 Features and model

We implemented a ranker based on a logistic regression model that estimates the probability of a question (q) having learning value (lv) given the sentence (s) and template (t) used to generate the question: $p(lv|s,t,q)$. The feature set used by our ranker included length, language model, SRL, named entity, predicate, and glossary features. The n parameters (θ) of the model were estimated by optimizing the L2-regularized log-likelihood of the training data. That is, given (4.16), we minimize (4.17). Each vector $\mathbf{x}^{(i)}$ is one of our m training samples and $y^{(i)}$ is the sample's learning value label. The regularization parameter (λ) is chosen using cross-validation on the training set.

$$h_\theta(\mathbf{x}) = \frac{1}{1 + e^{-\theta^T \mathbf{x}}} \tag{4.16}$$

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)} \log h_\theta(\mathbf{x}^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(\mathbf{x}^{(i)}))] + \frac{\lambda}{2m}\sum_{j=1}^{n}\theta_j^2 \tag{4.17}$$

We did not explicitly define features to account for vagueness. The type of vagueness we can feasibly capture at the level of a single sentence or question would be based on pronouns. The number of noun phrases containing only pronouns might be an appropriate measure of vagueness for a sentence or question [19]. Two questions arise here. First, do we want to risk penalizing templates whose question string contains pronouns? Template authors have the freedom to specify any question string they feel is appropriate, and that text could very well contain pronouns. The answer to this question is obviously *no*. This issue could perhaps be avoided altogether by examining the pronoun content of role text being inserted into the template. The second question is, can other features work together to appropriately penalize this kind of vagueness? We will return to this question when we discuss our ranking results.

Length features

These features captured the number of tokens in a question and the source sentence from which it was generated.

Language model features

Even though we have already seen that we can generate a high percentage of grammatical questions, we wanted to incorporate some measure of grammaticality into our model. Experimental results in the existing literature have differed regarding the significance of language model scores. Heilman and Smith report language model scores as being particularly helpful [19], while McConnell et al. report their language model features as being extremely helpful [26]. We wanted to investigate this for ourselves. To that end, we included a language model score for questions. We used a tri-gram language model estimated from a selection of abstracts from English Wikipedia[3], totalling 63.5 million tokens. It is a limitation of this language model that it was not estimated from a corpus containing a large number of questions.

SRL features

We extracted SRL features from both the source sentence predicate frame and the template used to generate a given question. The source sentence SRL features included token counts for the predicate and each of the roles `A0-A4`, `V`, `AM-ADV`, `AM-CAU`, `AM-DIR`, `AM-DIS`, `AM-LOC`, `AM-MNR`, `AM-MOD`, `AM-NEG`, `AM-PNC`, and `AM-TMP`. The template SRL features included binary features for each of the above roles indicating whether or not the template called for the text of that role to be used in the text of the question.

Named entity features

We included binary features for nine named entity types identified by Stanford NER [14] and SUTime [8]: `DATE`, `DURATION`, `LOCATION`, `MISC`, `NUMBER`, `ORGANIZATION`, `PERCENT`, `PERSON`, and `SET`. These features noted the presence or absence of each of these named entity types in the source sentence and the question.

Predicate features

Predicate depth is the only predicate feature we included. We wanted to examine the assumption made by [25] that predicates depth has some bearing on question quality.

[3] http://en.wikipedia.org

Glossary features

The glossary effectively gives us a simple topic model. Questions that include glossary terms should be preferred over those that do not. We extracted an *in-terms-of* graph from the glossary with directed links from each term to the terms appearing in its gloss. The glossary features used in our model were the number of glossary terms appearing in the source sentence and the question as well as the average degree of those terms in the *in-terms-of* graph.

4.4.2 Results

To evaluate ranking, we tested on each of our ten documents, holding the remaining nine out for training each time. We report our cross-validation results in terms of average *precision-at-N*, the percentage of questions in our N top-ranked questions that the evaluator identified as having learning value. Table 4.2 shows results for N= 1, 10, 25, and 50. The average learned weights of our model are shown in Figure 4.4. The ranking results for one test set can be seen in Appendix B. The average precision-at-1 shows that our model is able to

N	avg. precision-at-N (%)	variance
1	70.0	2333.3
10	34.0	137.8
25	29.6	132.3
50	26.6	92.5

Table 4.2: Precision-at-N values for automatically ranked questions

accurately predict the top question 70% of the time. The high variance is due to the other 30% of cases in which the top-ranked question does not in fact have learning value.

Having come this far, we can now at least attempt to answer some of the questions posed along the way concerning what constitutes a "good" question. Our human evaluator commented that shorter sentences seemed to produce better questions. Among the 30 negatively-weighted features, only four features received more negative weight than source sentence length. This would seem to corroborates the evaluator's observation.

Note that the question language model score appears to have a positive weight, but the language model score is in fact a log probability, so this positive weight is in fact a penalty.

The language model score of a question is strongly influenced by the question string of the template that generated the question. In fact, given the relatively small size of our language model, this score is almost always determined *entirely* by the question string of the template. Although this means we cannot grammatically separate any of the questions generated by a single template, it gave our model a way to reward the more concisely-worded templates. We experimented with removing the question language model score during the training phase and re-inserting it with a large, arbitrary weight, but this produced poor results.

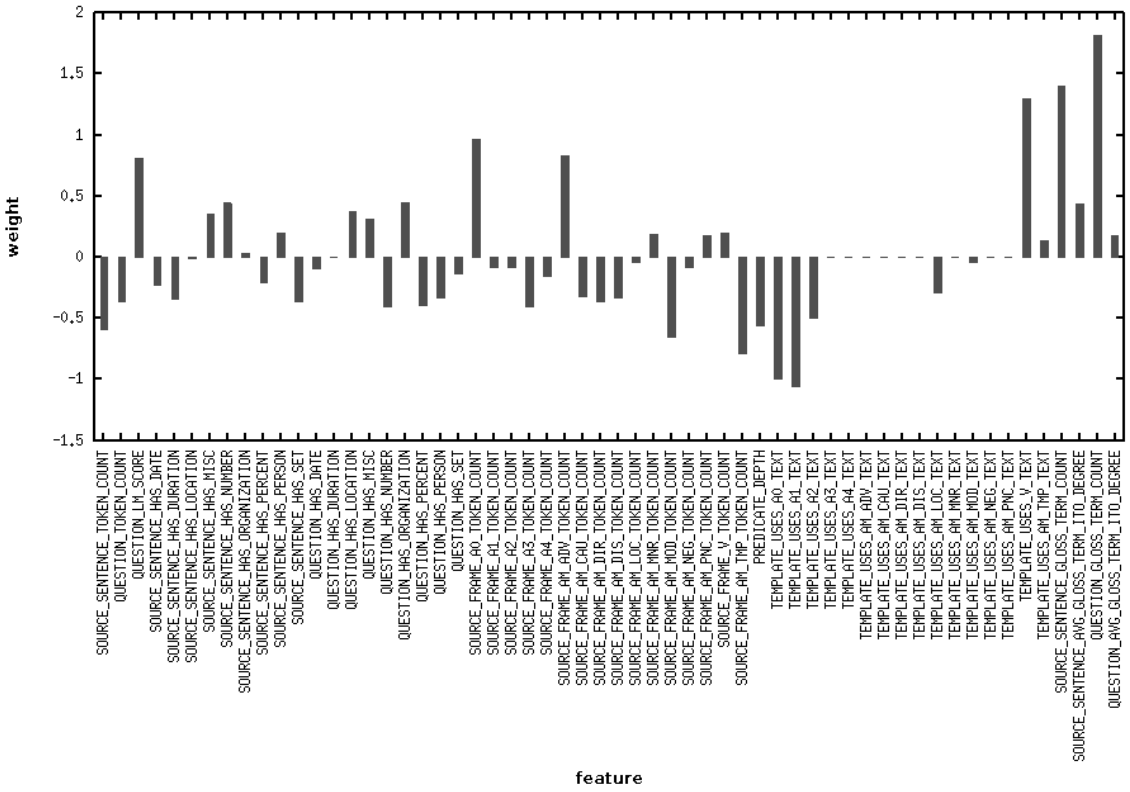

Figure 4.4: The average feature weights learned by our ranking model across ten cross-validation iterations

Another question that has lingered since Chapter 2 is the question of whether or not the depth of a predicate in a sentence influences the quality of the questions that can be generated from it. Mannem et al. believed that question quality degrades the deeper

the predicate [25], and our model justifies that belief. However, though the weight of the predicate depth feature was indeed negative, it was not the most negatively weighted feature.

It should be no surprise that the glossary features received high weights. In fact, the number of glossary terms appearing in a question received the highest weight of all features. Clearly, sentences that have glossary terms are the best candidates for question generation, and more importantly, we should always ask questions about glossary terms if we want to be confident about the learning value of those questions. That is not to say we will always generate highly-ranked questions from sentences that contain glossary terms. Given a sufficiently long sentence, the length and language model features will overpower the glossary features. This is our second piece of evidence that suggests sentence simplification would be beneficial.

The importance of glossary features in determining learning value is evident in the weights of other features. The `A0`, `A1`, and `A2` roles quite frequently appear as slots in our templates, and we see those features (`TEMPLATE_USES_A0_TEXT`, `TEMPLATE_USES_A1_TEXT`, and `TEMPLATE_USES_A2_TEXT`) rather heavily penalized. This is most likely due to a great many questions being generated using those templates and not being evaluated favourably in terms of learning value.

Earlier, we asked if it was even necessary to explicitly model vagueness. We examined the results of one cross-validation iteration and saw that of the 154 questions in that test set, 19 exhibited the kind of vagueness we can identify using the pronoun-based heuristic described earlier. The highest-ranked of these questions was ranked the 69th best among the entire test set, with an estimated probability of having learning value of 0.16. We speculate that the low ranking of these questions was due to the interaction of the `SOURCE_FRAME_A0_TOKEN_COUNT` and glossary features. A pronoun in the `A0` role is neither long nor a glossary term, so little reward is given to questions generated from such predicate frames. Although this is not conclusive proof that we do not need explicit vagueness features, it is suggestive.

4.5 Discussion

We wanted to evaluate our approach to QG in a way that was mindful of purpose for which the questions were being generated. Evaluation schemes seen in previous QG efforts have focused on the notion of *acceptability*. In some sense, we too have tried to identify *acceptable* questions, but we set a higher standard of acceptability. Whereas others have considered

questions acceptable given the absence of pre-defined deficiencies, we have taken a more pedagogically-motivated approach that examines the *learning value* of a question, which as we have seen, is not simply a matter of the question being free from syntactic and semantic deficiencies.

Based on our experience developing templates and evaluating their performance, we can draw some conclusions about how future templates should be written. We have seen that both domain-specific and general-purpose templates can produce questions with learning value, so a mixture of the two is important. We have also observed that templates with slots for the more rare semantic roles, such as `R-A0`, are of little value. On the other hand, templates that use the much more common `A0`, `A1`, and `A2` roles with named entity constrains are more useful. The latter type of template is the type we should favour going forward. The more common roles are most useful when they also contain glossary terms, so adding a glossary term filter would be beneficial.

The results in Appendix B show a few undesirable situations in which questions that appear to be very good receive very poor scores. Table 4.3 shows three such cases. When looking at these scores, it is important to consider that the model assigns very low scores to most of the questions. In fact, only the top six ranked questions have a score greater than 0.5. For this reason, a question's relative ranking is more important to consider than its absolute score. Although the first two questions in Table 4.3 have low scores, they are both ranked relatively high.

Rank	**Question**	**Model Score**
16	What can a large meteor impact cause?	0.39169998984520582
31	What produces carbon dioxide?	0.29965670956287466
152	Summarize the influence of heat on the environment.	0.033673471086601951

Table 4.3: Good questions with poor scores

What we should be most concerned about are situations demonstrated by the third question, *Summarize the influence of heat on the environment.*, which is ranked as the third-worst question from the 154 question document. The sentence used to generate this question is as shown in (4.18).

Once in the atmosphere, these substances absorb heat from the sun instead of letting it through to the Earth. (4.18)

Several features contribute to the question's very low score. First of all, (4.18) is fairly long, and sentence length is among the more strongly penalized features. In addition, the predicate frame that generates the question includes *Once in the atmosphere* as an `AM-TMP`, and the `SOURCE_FRAME_AM_TMP_TOKEN_COUNT` feature is also among the most negatively-weighted features. Finally, the fact that the template used to generate the question uses the source frame's `A1` text, *heat*, which is not a glossary term, renders the question irredeemable as far as the ranking model is concerned.

The lesson to be learned here is that there is more that needs to be done to evaluate our approach to ranking. While our results are interesting, a larger study using a larger corpus and multiple evaluators is needed. A larger set of candidate features and an ablation study like that done by Heilman and Smith [19] might help us identify a better set of features for ranking. As noted by our evaluator, the question type also plays a key role in determining learning value. In our approach the question type is determined entirely by a template's question string. We have not accounted for this in our ranking model. However, closer collaboration with educators can help refine templates in order to reduce the variance in template quality.

Chapter 5

Conclusion

This thesis has presented a novel approach to generating questions from text that combines the shallow semantics of SRL with the flexibility of question templates. Although SRL and templates have each been used by other approaches to QG, we have combined the two in a novel, interesting way. We wanted to demonstrate that templates, which have fallen out of favour in the literature, still have value. We believe we have succeeded. Templates provide a natural way to combine domain-specific and domain-independent generation in a single framework. Our evaluation methodology focused on assessing questions from an educational perspective. We conducted a pedagogically-motivated evaluation that identified questions that were not merely free from obvious deficiencies but in fact had real pedagogical utility. The evaluation revealed that we were in fact able to generate some questions that were distinctly above level 1 in the Bloom taxonomy. Summarization questions in particular are at the level of "understanding" in the taxonomy.

QG from text is far from a solved problem. There is much more to explore in terms of algorithms and evaluation. In terms of the research presented in this thesis, the next necessary step would be the selection algorithm mentioned in the previous chapter. While a simple selection algorithm could have been examined in this thesis, such an algorithm would not be appropriate given the context in which the questions we have generated are to be used. An algorithm that selects based on a coverage metric might not provide the most educationally suitable set of questions. This research will be extended and integrated into the nStudy system [36]. This combined system will be adaptive to the needs of the learner. In the context of learners with varying levels of concept mastery exploring learning materials from diverse sources, a selection algorithm needs to take a more comprehensive

approach. One factor would be verbosity. Of the total number of questions we generate for a given document, how many should we give to the learner? This might depend on the learner's perceived mastery of the concepts described in the document. If the system had access to the learner's answers to previous questions, could we use that information to identify questions that would most benefit the learner? The full design of a selection algorithm must be educationally-informed. User studies needed to properly collect data for such an algorithm and evaluate its effectiveness are well outside the scope of this thesis.

Further work might also investigate methods for making template development easier. A partially-automated, human in-loop method for template authoring would be useful. We might also want to investigate methods to automatically generate templates. One possible approach would be to solicit human-generated questions and attempt to learn a set of templates from sentence-question pairs.

Although more work can be done to improve the quality of questions generated from single sentences, leveraging multiple sentences for QG has the potential to provide a quantum leap in the pedagogic utility. As a first step in that direction, we have begun experimenting with a multi-sentence QG method that combines templates, a glossary, and discourse connectives. Inter-sentential discourse connectives, such as *for example*, *therefore*, *however*, and *furthermore*, provide an inexpensive and reasonably robust way to identify groups of sentences that we can and should use to generate questions. Agarwal et al.[1] demonstrate a system that uses discourse connectives for multi-sentence QG, but their approach does not truly integrate multi-sentence content into questions. Once they identify the arguments associated with the connective, `Arg1` and `Arg2`, they use syntactic transformations to produce questions from one and only one of those arguments. Furthermore, these questions are of pre-defined type based on the type of connective. We envision a method in which templates have `Arg1` and/or `Arg2` slots. The glossary would allow us to ask more targeted questions given the appearance of glossary terms in `Arg1` and/or `Arg2`. It is difficult to envision a method of generating questions from single sentences that could frequently generate questions at the higher Bloom levels.

Aside from evaluating our approach using a larger corpus and multiple human judges, more can be done in terms of evaluation. As noted in the previous chapter, much more thorough evaluation of ranking features is needed. Aside from this, in-situ educational studies are also needed. These studies might include a Turing test style of study in which subjects are given a mixture of human and machine-generated questions and asked to rank them.

More important, though, is the need for a study of the effectiveness of our automatically-generated questions. We might consider a study using three groups of learners studying a particular piece of text. One group would receive no questions, another would receive only human-generated questions, and the third group would receive only automatically-generated questions. Performance on a test might shed some light on the issue of effectiveness.

Bibliography

[1] Manish Agarwal, Rakshit Shah, and Prashanth Mannem. Automatic question generation using discourse cues. In *Proceedings of the 6th Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–9. Association for Computational Linguistics, 2011.

[2] Husam Ali, Yllias Chali, and Sadid A Hasan. Automation of question generation from sentences. In *Proceedings of QG2010: The Third Workshop on Question Generation*, pages 58–67, 2010.

[3] Lorin W Anderson, David R Krathwohl, and Benjamin Samuel Bloom. *A taxonomy for learning, teaching, and assessing*. Longman, 2005.

[4] Benjamin S Bloom, MD Engelhart, Edward J Furst, Walker H Hill, and David R Krathwohl. Taxonomy of educational objectives: Handbook i: Cognitive domain. *New York: David McKay*, 19:56, 1956.

[5] Zhiqiang Cai, Vasile Rus, Hyun-Jeong Joyce Kim, Suresh C. Susarla, Pavan Karnam, and Arthur C. Graesser. Nlgml: A markup language for question generation. In Thomas Reeves and Shirley Yamashita, editors, *Proceedings of World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education 2006*, pages 2747–2752, Honolulu, Hawaii, USA, October 2006. AACE.

[6] Aimee A. Callender and Mark A. McDaniel. The benefits of embedded question adjuncts for low and high structure builders. *Journal Of Educational Psychology (2007)*, pages 339–348, 2007.

[7] Xavier Carreras and Lluís Màrquez. Introduction to the conll-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 152–164. Association for Computational Linguistics, 2005.

[8] Angel X Chang and Christopher D Manning. Sutime: a library for recognizing and normalizing time expressions. *Language Resources and Evaluation*, 2012.

[9] Eugene Charniak and Micha Elsner. Em works for pronoun anaphora resolution. In *Proceedings of the 12th Conference of the European Chapter of the Association for*

Computational Linguistics, pages 148–156. Association for Computational Linguistics, 2009.

[10] Wei Chen. Understanding mental states in natural language. In *Proceedings of the Eighth International Conference on Computational Semantics*, pages 61–72. Association for Computational Linguistics, 2009.

[11] Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

[12] Ann Copestake, Dan Flickinger, Carl Pollard, and Ivan A Sag. Minimal recursion semantics: An introduction. *Research on Language and Computation*, 3(2-3):281–332, 2005.

[13] Nell K Duke and P David Pearson. Effective practices for developing reading comprehension. *What research has to say about reading instruction*, 3:205–242, 2002.

[14] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating non-local information into information extraction systems by gibbs sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 363–370. Association for Computational Linguistics, 2005.

[15] Arthur C Graesser, Shulan Lu, George Tanner Jackson, Heather Hite Mitchell, Mathew Ventura, Andrew Olney, and Max M Louwerse. Autotutor: A tutor with dialogue in natural language. *Behavior Research Methods, Instruments, & Computers*, 36(2):180–192, 2004.

[16] Arthur C Graesser and Natalie K Person. Question asking during tutoring. *American educational research journal*, 31(1):104–137, 1994.

[17] Michael Heilman and Noah A Smith. Question generation via overgenerating transformations and ranking. Technical report, DTIC Document, 2009.

[18] Michael Heilman and Noah A Smith. Extracting simplified statements for factual question generation. In *Proceedings of QG2010: The Third Workshop on Question Generation*, pages 11–20, 2010.

[19] Michael Heilman and Noah A Smith. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617. Association for Computational Linguistics, 2010.

[20] Saidalavi Kalady, Ajeesh Elikkottil, and Rajarshi Das. Natural language question generation using syntax and keywords. In *Proceedings of QG2010: The Third Workshop on Question Generation*, pages 1–10, 2010.

[21] Dan Klein and Christopher D Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 423–430. Association for Computational Linguistics, 2003.

[22] David R Krathwohl. A revision of bloom's taxonomy: An overview. *Theory into practice*, 41(4):212–218, 2002.

[23] Roger Levy and Galen Andrew. Tregex and tsurgeon: tools for querying and manipulating tree data structures. In *LREC 2006*, 2006.

[24] David Lindberg, Fred Popowich, John Nesbit, and Phil Winne. Generating natural language questions to support learning on-line. In *Proceedings of the 14th European Workshop on Natural Language Generation*. Association for Computational Linguistics, 2013. To appear.

[25] Prashanth Mannem, Rashmi Prasad, and Aravind Joshi. Question generation from paragraphs at upenn: Qgstec system description. In *Proceedings of QG2010: The Third Workshop on Question Generation*, pages 84–91, 2010.

[26] Claire Cooper McConnell, Prashanth Mannem, Rashmi Prasad, and Aravind Joshi. A new approach to ranking over-generated questions. In *Proceedings of the AAAI Fall Symposium on Question Generation*, 2011.

[27] Jack Mostow and Wei Chen. Generating instruction automatically for the reading strategy of self-questioning. In *Proceedings of the 2009 conference on Artificial Intelligence in Education: Building Learning Systems that Care: From Knowledge Representation to Affective Modelling*, pages 465–472. IOS Press, 2009.

[28] Rodney D Nielsen. Question generation: Proposed challenge tasks and their evaluation. In *Proceedings of the Workshop on the Question Generation Shared Task and Evaluation Challenge, Arlington, Virginia*, 2008.

[29] Martha Palmer, Daniel Gildea, and Paul Kingsbury. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106, 2005.

[30] Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James Martin, and Dan Jurafsky. Shallow semantic parsing using support vector machines. In *Proceedings of HLT/NAACL*, page 233, 2004.

[31] Ehud Reiter and Robert Dale. Building applied natural language generation systems. *Natural Language Engineering*, 3(1):57–87, 1997.

[32] Vasile Rus and Arthur C Graesser. The question generation shared task and evaluation challenge. In *Workshop on the Question Generation Shared Task and Evaluation Challenge, Final Report, The University of Memphis: National Science Foundation*, 2009.

[33] Vasile Rus, Brendan Wyse, Paul Piwek, Mihai Lintean, Svetlana Stoyanchev, and Cristian Moldovan. Overview of the first question generation shared task evaluation challenge. In *Proceedings of the Third Workshop on Question Generation*, pages 45–57, 2010.

[34] Liana Stanescu, Cosmin Stoica Spahiu, Anca Ion, and Andrei Spahiu. Question generation for learning evaluation. In *Computer Science and Information Technology, 2008. IMCSIT 2008. International Multiconference on*, pages 509–513. IEEE, 2008.

[35] Andrea Varga and Le An Ha. Wlv: A question generation system for the qgstec 2010 task b. In *Proceedings of QG2010: The Third Workshop on Question Generation*, pages 80–83, 2010.

[36] Philip H Winne and Allyson F Hadwin. nstudy: Tracing and supporting self-regulated learning in the internet. In *International handbook of metacognition and learning technologies*, pages 293–308. Springer, 2013.

[37] John H Wolfe. Automatic question generation from text-an aid to independent study. In *ACM SIGCUE Outlook*, volume 10, pages 104–112. ACM, 1976.

[38] Brendan Wyse and Paul Piwek. Generating questions from openlearn study units. In *AIED 2009: 14 th International Conference on Artificial Intelligence in Education Workshops Proceedings*, 2009.

[39] Xuchen Yao and Yi Zhang. Question generation with minimal recursion semantics. In *Proceedings of QG2010: The Third Workshop on Question Generation*, pages 68–75. Citeseer, 2010.

Appendix A

Templates

This appendix lists the templates we used to evaluate our approach. The template IDs are used in Figures 4.2 and 4.3.

ID	Template
1	Summarize the influence of [A1 -lpp !comma !nv] on the environment .
2	How long can [A1 !nv] last ? ## [AM_TMP dur] [A2 null]
3	How long can [A0 -lpp !nv] last ? ## [AM_TMP dur] [A1 null]
4	How would you describe [A2 -lpp misc] ?
5	What is the geographic scope of [A1 -lpp !nv] ? ## [AM_LOC]
6	Where can we find [A1 -lpp !nv] ? ## [AM_LOC]
7	How would you describe the cause of [A2 -lpp !nv misc] ? ## [A0 null]
8	What caused [A2 -lpp !nv misc] ? ## [A0 null]
9	Where would you find [A1 !nv] ? ## [A0 loc]
10	What [V be] [A1 -lpp !nv !comma] ?
11	What [V !be !have pres tps] [A1 !nv] ? ## [A2 null] [A0]
12	What [V !be !have pres tps] [A1 !nv] [A2 !nv] ? ## [A0 null] [A0]
13	Do the conditions that cause [A1 !comma !nv] differ from normal conditions ? ## [AM_TMP comp]
14	What data would indicate a(n) [A1 !comma !nv] event ? ## [AM_TMP comp]
15	When can [A1 -lpp !nv] [V !be lemma] ? ## [AM_TMP !date] [AM_MNR null]
16	Why is [A1 singular !nv] important ? ## [AM_ADV]
17	Why are [A1 plural !nv] important ? ## [AM_ADV]
18	What happens when [A1 singular] is [V !be !have pastpart tps] ? ## [AM_PNC]
19	What happens when [A1 plural] are [V !be !have pastpart tps] ? ## [AM_PNC]
20	Is [A1 singular] rare ? ## [AM_TMP !date] [AM_MNR null]
21	Are [A1 plural] rare ? ## [AM_TMP !date] [AM_MNR null]
22	How often does [A1 singular] occur ? ## [AM_TMP !date] [AM_MNR null] [V !be]
23	How often do [A1 plural] occur ? ## [AM_TMP !date] [AM_MNR null] [V !be]

ID	Template
24	What can [A0 -lpp !nv] [V !be !have lemma] ? ## [A1] [A2 null]
25	What can [A0 -lpp !nv] [V !be !have lemma] [A2] ? ## [A1]
26	How can [A0 -lpp !nv] [V !be !have lemma] [A1] ? ## [A2 null]
27	How can [A0 -lpp !nv] [V !be !have lemma] [A1] [A2] ?
28	What can you say about the nature or behaviour of [A1] [AM_TMP] ? ## [A0 null] [AM_MNR]
29	How does [A1 -lpp singular] compare to [A0 -lpp] ? ## [AM_EXT]
30	How do [A1 -lpp plural] compare to [A0 -lpp] ? ## [AM_EXT]
31	Describe the factor(s) that contribute to [A1] . ## [AM_CAU !nv] [A2 null]
32	Describe the factor(s) that contribute to [A1] [V prespart fps] [A2] . ## [AM_CAU]
33	How are [A1 !nv -lpp], [A2 !nv -lpp], and [A0 !nv -lpp] related ?
34	What can cause [A1 !nv -lpp] to be [V !be !have pastpart fps] ? Describe the consequences of this ? ## [AM_ADV] [A0 null]
35	What can cause [A1 !nv -lpp] [V !be !have lemma] ? What are some of the consequences of this ? ## [AM_ADV] [A0 null]
36	What can cause [A1 !nv -lpp] to [V !be !have lemma] ? What are some of the consequences of this ? ## [AM_ADV] [A0 null]
37	What allows [A0 -lpp !nv] to [V !be !have lemma] [A1 !nv] [A2] ? Describe the consequences of this . ## [AM_LOC null] [A0]
38	What allows [A0 -lpp !nv] to [V !be !have lemma] [A1 !nv] [AM_LOC] ? Describe the consequences of this . ## [A0]
39	In what way do [A0 -lpp !nv plural] [V lemma] [A1 !nv] ? ## [AM_MNR]
40	In what way does [A0 -lpp !nv singular] [V lemma] [A1 !nv] ? ## [AM_MNR]
41	How would you summarize [A2 -lpp !nv] ? ## [A1 null] [A0]
42	Which [A0 -ldt] [V !be !have prespart tps] [A1] [A2] ? ## [R_A0] [AM_LOC null]
43	Which [A0 -ldt] [V !be !have prespart tps] [A1] ? ## [R_A0] [AM_LOC null]
44	Which [A0 -ldt] [V !be !have prespart tps] [A1] [AM_LOC] ? ## [R_A0]
45	Which [A0 -ldt] [V !be !have prespart tps] [A2] ? ## [R_A0] [AM_LOC null]
46	Which [A0 -ldt] [V !be !have prespart tps] [A2] ? ## [R_A0]
47	Describe the important role played by [A1 -lpp] . ## [AM_PNC]
48	Name a [A1 -ldt singular] . ## [R_A1]
49	Name some [A1 -ldt plural] . ## [R_A1]
50	What phenomenon/process is demonstrated by [A0 -lpp !nv] ? ## [AM_DIS]
51	[A0] [V prespart fps] [A1 !nv] is an example of what important process ? ## [AM_DIS]
52	How [AM_MOD] [A0 -lpp] [V lemma] [A1 !nv] ?

Appendix B

Ranking Results

We present the ranking results for one of our documents. The "Learning Value" column shows the evaluator's judgement for each question (0=does not have learning value, 1=has learning value).

Rank	Question	Model Score	Learning Value
1	What is the layer of the earth's atmosphere directly above the troposphere?	0.80011982376894009	1
2	What is volcanic eruptions?	0.78496258914024186	0
3	Name a layer of the earth's atmosphere directly above the troposphere.	0.67256487767585249	1
4	Summarize the influence of the layer of the earth's atmosphere directly above the troposphere on the environment.	0.65356604517393901	0
5	Summarize the influence of volcanic eruptions on the environment.	0.63715261970225157	1
6	What is a major greenhouse gas?	0.50368724970145273	1
7	What can meteor impacts spew up?	0.49925230284088745	0
8	What creates the ash and aerosols?	0.48417585002394892	1
9	What can the Laki volcanic eruptions create?	0.46684988617659856	1
10	What can the Eyjafjallajokull volcanic eruptions in Iceland gain?	0.45229057012100271	0
11	What notices an unusually cool summer and severe winter?	0.44834931173837467	0

Rank	Question	Model Score	Learning Value
12	How can the Laki volcanoes send a large quantity of ash and sulfur gases up into the stratosphere, which is the layer of the earth's atmosphere directly above the troposphere?	0.44189330589639608	0
13	What spews up dust, debris and gases into the atmosphere?	0.42273626719673857	1
14	What produces the carbon dioxide?	0.40244732577269343	0
15	What can these aerosols absorb?	0.39899568594912671	1
16	What can a large meteor impact cause?	0.39169998984520582	1
17	How can volcanoes release carbon dioxide, which is a major greenhouse gas?	0.37312513389327201	0
18	What absorbs sunlight?	0.3715990161386783	1
19	How can the Laki volcanic eruptions create the ash and aerosols?	0.36143001391698637	0
20	What can the sulfur gases create?	0.35159363683329259	1
21	How can some scientists believe that a large meteor impact caused the extinction of the dinosaurs by drastically altering the Earth's climate?	0.34819274828913466	1
22	Meteor impacts spewing up dust, debris and gases into the atmosphere is an example of what important process?	0.34734293086377288	0
23	How can meteor impacts spew up dust, debris and gases into the atmosphere?	0.34501193686086562	0
24	How may disasters, or catastrophic events, that are large enough have an impact on the earth's climate?	0.34390601371977031	0
25	Summarize the influence of a major greenhouse gas on the environment.	0.3261960296318267	0
26	What can volcanoes release?	0.32025506642981405	1
27	Name a major greenhouse gas.	0.31890677609586104	1
28	How can these aerosols absorb sunlight?	0.30626917487730754	0
29	What can the Laki volcanoes send?	0.3002259092233881	0
30	Summarize the influence of the ash and aerosols on the environment.	0.30003852825791205	1

Rank	Question	Model Score	Learning Value
31	What produces carbon dioxide?	0.29965670956287466	1
32	In what way do these aerosols absorb sunlight?	0.29943462236042107	0
33	What was the average temperature of the entire northern hemisphere?	0.29724308170417924	0
34	What grounds many flights in Europe?	0.29287721434354053	0
35	When can carbon dioxide produce?	0.2891765786491135	0
36	Summarize the influence of the Earth's climate on the environment.	0.28203172723487097	0
37	Summarize the influence of an unusually cool summer and severe winter on the environment.	0.28061268501716752	0
38	What can daily air traffic in Europe produce?	0.27952279656434231	0
39	Summarize the influence of an impact on the earth's climate on the environment.	0.27948730082707612	0
40	What can a volcano produce?	0.27707981415427463	0
41	How can the sulfur gases create substances called aerosols?	0.26646769717828173	0
42	Name some disasters, or catastrophic events.	0.26447610722825771	1
43	What can cause the heat from the Earth's surface back towards the sun to be reflected? Describe the consequences of this?	0.25991932467309853	0
44	What emits the ash and aerosols?	0.25776787227055137	0
45	What can cause the heat from the Earth's surface back towards the sun reflect? What are some of the consequences of this?	0.25681789125486648	0
46	What can cause the heat from the Earth's surface back towards the sun to reflect? What are some of the consequences of this?	0.2537407769193466	0
47	What can this cloud block out?	0.24508787454177167	1
48	Summarize the influence of the carbon dioxide on the environment.	0.2449531019834319	0
49	Summarize the influence of the climate on the environment.	0.2412531117354737	0

Rank	Question	Model Score	Learning Value
50	How can this cloud block out some of the sunlight that would have normally reached the Earth's surface?	0.23940727229157985	1
51	What phenomenon/process is demonstrated by meteor impacts?	0.23564730429641992	1
52	Why is an unusually cool summer and severe winter important?	0.22981934845220936	0
53	What phenomenon/process is demonstrated by the Eyjafjallajokull volcanic eruptions in Iceland?	0.2244123268660197	0
54	What can these aerosols release?	0.21775923836794142	1
55	Summarize the influence of sunlight on the environment.	0.21559834460914976	0
56	Summarize the influence of a large effect upon the Earth's climate on the environment.	0.2140906877638353	0
57	What can some scientists believe?	0.20959291005086753	0
58	What can the sunlight reach?	0.20901215419461505	0
59	What causes the extinction of the dinosaurs?	0.20608311784606692	1
60	How can the Eyjafjallajokull volcanic eruptions in Iceland gain worldwide attention?	0.19814068824288814	0
61	How can a volcano produce the carbon dioxide?	0.19701429397482298	0
62	The Eyjafjallajokull volcanic eruptions in Iceland gaining worldwide attention is an example of what important process?	0.19669484413064134	0
63	How does the warming effect of the carbon dioxide produced by a volcano compare to the cooling effect of the ash and aerosols it emits?	0.18246073550592176	1
64	What can the volcano produce?	0.17991698259646924	0
65	What was the average temperature?	0.17897344653193578	0
66	What reflects the heat from the Earth's surface back towards the sun these substances?	0.17847572232220119	0
67	Summarize the influence of carbon dioxide on the environment.	0.17088500910879664	1
68	How can this effect have a large effect upon the Earth's climate?	0.16922463893156969	1

Rank	Question	Model Score	Learning Value
69	What can it emit?	0.16376062149114373	0
70	Summarize the influence of the heat from the Earth's surface back towards the sun on the environment.	0.1632672847434152	0
71	When can the Earth's surface reach?	0.16220190524043401	0
72	What reaches the Earth's surface?	0.16166111806721839	1
73	How can a large meteor impact cause the extinction of the dinosaurs?	0.16147877665050578	1
74	In what way does a large meteor impact cause the extinction of the dinosaurs?	0.15714339564212287	0
75	What can humans produce?	0.15616697956000736	0
76	Summarize the influence of many flights in Europe on the environment.	0.15505566935389184	0
77	When can the average temperature reduce?	0.15261402929058651	1
78	What was this phenomenon?	0.13969100352111138	0
79	Summarize the influence of substances on the environment.	0.13934196707716151	0
80	Summarize the influence of the sulfur gases on the environment.	0.13909530983083584	1
81	What phenomenon/process is demonstrated by volcanoes?	0.13839371846796975	0
82	When can less than one-half the amount produce?	0.13629796766859423	0
83	Why is the average temperature of the entire northern hemisphere important?	0.13526870448316267	0
84	How can a volcano produce carbon dioxide?	0.13485283840507972	0
85	Which sunlight reached the Earth's surface?	0.13241836879475877	0
86	What can cause the average temperature to be reduced? Describe the consequences of this?	0.13048393474340655	1
87	When can the average temperature of the entire northern hemisphere reduce?	0.13014817315183938	1

Rank	Question	Model Score	Learning Value
88	Why is the heat from the Earth's surface back towards the sun important?	0.12987967549703361	0
89	Which created an ash cloud grounded many flights in Europe?	0.12927106445751632	0
90	What can Benjamin Franklin note?	0.12912454088555417	0
91	What notes this phenomenon?	0.128879978973548	0
92	How can the sunlight reach the Earth's surface?	0.12874341032425593	0
93	What can cause the average temperature reduce? What are some of the consequences of this?	0.12865846724094868	0
94	Is carbon dioxide rare?	0.12853847988134706	0
95	What can cause the average temperature to reduce? What are some of the consequences of this?	0.12685481218217251	1
96	How can the volcano produce carbon dioxide?	0.12490363622410464	0
97	What produces the levels?	0.1231283262892312	0
98	How often does carbon dioxide occur?	0.12066794864385513	0
99	What produces less than one-half the amount?	0.12047747455339795	0
100	What releases it?	0.11855464048769801	0
101	What gains worldwide attention?	0.11788729190529432	0
102	How would the sunlight reach the Earth's surface?	0.11783445813722454	0
103	How can it emit the ash and aerosols?	0.11393128305343485	0
104	Summarize the influence of the extinction of the dinosaurs on the environment.	0.11082012095560594	0
105	Where can we find the sulfur gases?	0.1071948415507392	0
106	It emitting the ash and aerosols is an example of what important process?	0.10438865637726287	0
107	What was this?	0.10423205021220736	0
108	What is the geographic scope of the sulfur gases?	0.1041358790270504	0
109	When can it release?	0.10406157863581054	0
110	When can this put?	0.10160759022174179	0

Rank	Question	Model Score	Learning Value
111	How would the sunlight have normally?	0.090752722257673918	0
112	How can these aerosols release it?	0.089520521187072405	0
113	What reduces the average temperature by only 5 C.?	0.088284307334926274	0
114	Summarize the influence of the Earth's surface on the environment.	0.083055867398202726	0
115	What can these substances absorb?	0.078168708899445716	1
116	What phenomenon/process is demonstrated by some scientists?	0.077112368789635941	0
117	Summarize the influence of the average temperature on the environment.	0.076958006318902053	0
118	What reduces the average temperature of the entire northern hemisphere by 1 C?	0.075579294197257785	1
119	How can daily air traffic in Europe produce less than one-half the amount?	0.071144265867157427	0
120	Summarize the influence of an ash cloud on the environment.	0.068888234378247273	1
121	Summarize the influence of the levels on the environment.	0.068831279186724034	0
122	Summarize the influence of this phenomenon on the environment.	0.068726440570840833	0
123	What absorbs heat?	0.068447238159565743	1
124	Summarize the influence of less than one-half the amount on the environment.	0.068214733679731535	0
125	What allows these aerosols to release it back into space? Describe the consequences of this.	0.067676441419871947	0
126	Is the Earth's surface rare?	0.066110377884486315	0
127	Summarize the influence of the average temperature of the entire northern hemisphere on the environment	0.064777879398227475	0
128	What phenomenon/process is demonstrated by it?	0.063574429235729368	0
129	Summarize the influence of worldwide attention on the environment.	0.06254243187235467	0

Rank	Question	Model Score	Learning Value
130	Summarize the influence of normally on the environment.	0.062308559081989373	0
131	How often does the Earth's surface occur?	0.061791339525831601	0
132	Is the average temperature rare?	0.061173633213365677	0
133	Why is the average temperature important?	0.059955654899936761	0
134	How often does the average temperature occur?	0.057157376481638093	0
135	What puts this in perspective?	0.056807106097310857	0
136	Is less than one-half the amount rare?	0.054118497354707681	0
137	Summarize the influence of it on the environment.	0.052381886057258201	0
138	Is the average temperature of the entire northern hemisphere rare?	0.051352710806300092	0
139	Summarize the influence of effect on the environment.	0.051253100709156924	0
140	How often does less than one-half the amount occur?	0.050540498819851808	0
141	Summarize the influence of this on the environment.	0.049453812420735731	0
142	How can humans produce the levels?	0.049419395692805207	0
143	How often does the average temperature of the entire northern hemisphere occur?	0.047948300546133776	0
144	What is the geographic scope of it?	0.043922097006760481	0
145	Where can we find it?	0.043702733322861807	0
146	How can Benjamin Franklin note this phenomenon?	0.04302906360484824	0
147	Is it rare?	0.042573901167765167	0
148	In what way does Benjamin Franklin note this phenomenon?	0.041715610914216918	0
149	How often does this occur?	0.039840569785308873	0
150	How often does it occur?	0.03972710499887272	0
151	Is this rare?	0.037033583597906324	0
152	Summarize the influence of heat on the environment.	0.033673471086601951	1
153	How can these substances absorb heat?	0.026284578719297895	0
154	In what way do these substances absorb heat?	0.025468644844977642	0