Exploring Methods for Generating and Evaluating Skill Targeted Reading Comprehension Questions

by

Spencer McIntosh von der Ohe

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

Master of Science

Department of Computing Science

University of Alberta

Abstract

It takes skilled teachers a significant amount of time and effort to create high quality reading comprehension questions, often making it impractical to target a particular reader's weaknesses. Recently, language models have been proposed as a tool to help teachers fill this gap, allowing these teachers to generate questions targeting specific skill types.

In this thesis, we propose SoftSkillQG, a new soft-prompt based language model for generating skill targeted reading comprehension questions that does not require any manual effort to target new skills. We compare SoftSkillQG against a variety of strong baselines and show that it outperforms existing techniques on four out of five question quality metrics for the SBRCS dataset and human evaluation of Context Specificity on the QuAIL dataset. However, on the QuAIL dataset, T5 WTA [12], a previously proposed method using manually created prompts, outperforms SoftSkillQG in terms of perplexity and these same five metrics.

We investigate why SoftSkillQG performs poorly relative to T5 WTA, a method using manually created "hard" prompts, on the QuAIL dataset by examining both the data size and prompt initialization on SoftSkillQG's performance. We show that dataset size may be affecting performance, but augmenting training with silver data from the SQuAD dataset did not improve performance. On the other hand, initializing the prompt of SoftSkillQG using the same prompt as T5 WTA yielded nearly the same perplexity on the QuAIL dataset.

Finally, we perform a first of its kind analysis using the human annotations from our previous experiments to compare five different methods for evaluating sets of generated questions. We find that: MS-Jaccard4 [11] best captures the diversity of a set of questions, Best Reference Evaluation [12] aligns mostly closely with human judgement of Answerability; Cartesian Product evaluation aligns most closely with Context-Specificity; and Fréchet BERT Distance [11] aligns mostly closely with Fluency.

Preface

This thesis is an original work by Spencer McIntosh von der Ohe. This thesis is part of the study "Evaluating computer generated language," No. 00112093, which obtained ethics approval from the University of Alberta Research Ethics Board.

Machines take me by surprise with great frequency.

– Alan Turing, 1950.

Acknowledgements

I would like to thank Dr. Alona Fyshe for guiding me on my journey creating this thesis. I have learned so much from you over the last three years working with you. You have also always been patient and encouraging. I could not ask for a better advisor and mentor.

Additionally, I would like to thank Dr. Lili Mou and Dr. Denilson Barbosa for agreeing to be on my defense committee and taking the time to read this thesis.

As well, I am grateful for all of the help and feedback provided by members of Dr. Fyshe's research group. In particular, I would like to thank Dr. Bilal Ghanem for providing me guidance in starting this project, Dhruv Mullick for helping with human evaluation, and Dr. Alex Murphy for providing guidance and feedback on research for this thesis.

I would also like to thank Dr. Antonie Bodley for her feedback on this thesis.

Furthermore, I would like to thank my family: my mom, Tracy von der Ohe; my dad, Jackson von der Ohe; and my sister, Mason von der Ohe, for their support while working on this thesis.

Finally, the research in the thesis was supported by Alberta Innovates, the Natural Sciences and Engineering Research Council of Canada (NSERC), the Walter H Johns Graduate Fellowship, the Digital Research Alliance of Canada,s and the Alberta Machine Intelligence Institute (Amii).

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Reading is a critical skill for many aspects of daily life, including staying informed, communicating with others, and many career paths.

1.1 Motivation

However, children can struggle with reading comprehension, leading to detrimental effects on their studies and future success [37].

Reading comprehension questions are one way to evaluate and improve reading comprehension skills, but quality questions require significant effort by teachers to create, making it impractical to create targeted questions for a single student's weaknesses. One proposed solution to this problem is to formulate reading comprehension question generation as a sequence-to-sequence text generation task and use large language models to generate reading comprehension questions [11, 12, 26].

However, methods for automatically generating questions still do not achieve the same quality as questions created by skilled teachers. Additionally, many of the methods that come close require an answer to already be available before the question is created [26, 28, 42, 44, 47]. This requirement leads to extra work for teachers and may limit the type of questions asked. Other methods require manual effort to train limiting their ability to be applied to new skill types and languages [12, 39]. In this thesis, we try to address these limitations and narrow the gap in quality between automatically generated and human created questions.

1.2 Contributions

We propose, SoftSkillQG, a new question generation model that uses soft-prompt to control the types of questions generated. SoftSkillQG outperforms existing models on four out of five automatic evaluation metrics on the SBRCS dataset and Context-Specificity on the QuAIL dataset. However, T5 WTA [12], a method using manually created prompts outperforms SoftSkillQG across the same five metrics and perplexity on the QuAIL dataset.

As a results, we performed some additional experiments to help determine why SoftSkillQG does not perform as well on the QuAIL dataset. We first look at data size and show that although data size may be affecting the performance, performing additional training on a silver dataset does not improve the performance of SoftSkillQG. Next, we look at the initialization of SoftSkillQG's prompts and show that if SoftSkillQG is initialized using the manually created prompts of T5 WTA, it achieves nearly the same performance as T5 WTA.

Finally, we perform as first of its kind comparison of various question evaluation methods. We correlated each method with three aspects of human evaluation and self-similarity. We find that: MS-Jaccard4 [11] best captures the diversity of a set of questions, Best Reference evaluation [12] aligns mostly closely with human judgement of answerability; Cartesian Product evaluation aligns most closely with Context Specificity; and Fréchet BERT Distance [11] evaluation aligns mostly closely with Fluency.

We summarize our primary contributions as follows:

1. SoftSkillQG, a new method of generating targeted reading comprehension questions that does not require any manual steps and outperforms existing methods on the SBRCS dataset

2. A new comparison of methods for generating reading comprehension questions

3. An analysis of the weaknesses of SoftSkillQG and other soft-prompt based methods on the question generation task

4. An alternative method of initializing soft-prompts that can potentially resolve to these weaknesses

5. A first of its kind comparison of five diverse methods to automatically evaluate generated reading comprehension questions

Additionally, we answer the following research questions:

- Can we create targeted comprehension questions without using manually crafted prompts?

- Can we create these questions in a way that improves on previously reported techniques?

- Is there a single evaluation metric that work best to capture question quality or are multiple evaluation metrics required to capture different aspects of question quality?

1.3 Overview

In this chapter, we explain the motivation for this work, outline the contributions presented, and provide a brief overview of topics covered in each chapter of this thesis.

In Chapter 2, we provide background information required to understand later chapters and place this thesis in relation to existing work in the field.

In Chapter 3, we first outline five baseline methods to generate skill targeted reading comprehension questions. Then, we present SoftSkillQG and compare all methods using perplexity, five automatic evaluation methods and three aspects of human judgement.

In Chapter 4, we test the effect of dataset size on soft-prompt based methods like SoftSkillQG.

In Chapter 5, we test the effect of initialization on the perplexity of SoftSkillQG.

In Chapter 6, we explore five diverse methods for evaluating sets of reading comprehension questions and compare their ability to both capture diversity and align with three aspects of human judgement: Answerability, Fluency, and Context Specificity.

Finally, in Chapter 7, we summarize the preceding chapters. We then outline some limitations of this thesis and directions for future work.

Chapter 2

Background

This chapter provides relevant background information required to understand the methodology in later sections. It is divided into six sections: Language Modeling, Perplexity, Autoregressive Text Generation, Prompting, Reading Comprehension Questions, Question Generation, and finally, Question Evaluation.

2.1 Language Modeling

A language model is a statistical model that predicts the probability that a sequence of text will be observed, given another sequence of text has been observed, usually immediately preceding, or following the predicted sequence. Language models can be used for a variety of tasks, including text generation, text classification, and information retrieval. However, for this work, we focus exclusively on text generation.

In order for a language model to predict the probability of observing some text, the text must first be broken up into smaller units. In Natural Language Processing (NLP), these units are called *tokens*. Tokens are typically words but can also be pieces of words, whitespace, punctuation, or single characters. We show an example of a sentence broken into tokens in Figure 2.1.

2.1.1 T5 Language Model

For this thesis, we focus on methods that use the T5 language model [30]. T5 is a language model that takes a sequence of tokens as input and then

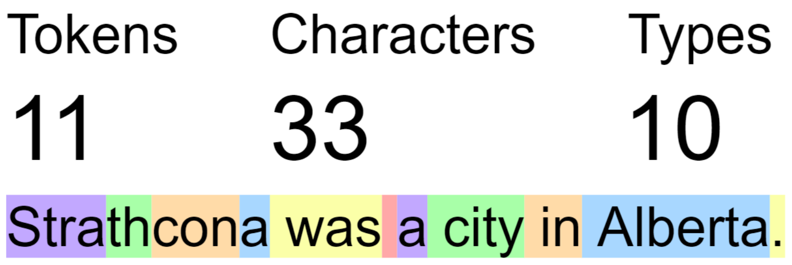


Figure 2.1: Tokenization of "Strathcona was a city in Alberta." by the T5 tokenizer [30]. Each color shows one of the eleven tokens. Notice that "Strathcona" is divided into four tokens, "Stra", "th", "con" and "a". Also, see that whitespace can either be part of another token, as in " was", or its own token, as demonstrated in the token after "was". Notice that "." is its own token. Finally, see that the token type "a" appears twice.

predicts probabilities for a separate output sequence. The T5 model has two main parts: the encoder and the decoder. The encoder takes the tokens in the input sequence as input and outputs a vector representation of the input [38]. The decoder then takes any tokens already in the output sequence and the vector representation from the encoder as input and outputs the predicted probability for the next token in the output sequence [38]. The T5 model is often used for tasks where it is easy to separate the input and output sequences, like in translation, summarization, and question answering, because it leads to higher performance with the same computation cost [30] compared to similar models that only use a decoder. See Figure 2.2 for an illustration of the T5 model.

2.2 Perplexity

Language models are often evaluated using a metric called perplexity. Perplexity is the reciprocal of the probability of a sequence of tokens normalized to the length of the sequence [18].

$$\text{Perplexity}(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i|w_1, ..., w_{i-1})}} \tag{2.1}$$

Equation 2.1 shows the formula to calculate perplexity for a given sequence W. N represents the number of tokens in W. w_i represents the i-th token in

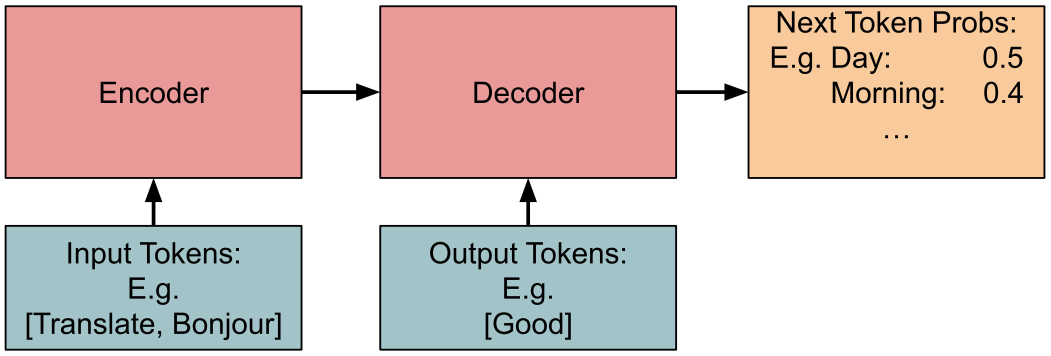


Figure 2.2: T5 Model. The blue boxes represent tokens, the red boxes represent parts of the language model, and the orange box represents the output probabilities from the model. Notice that the input tokens are separate from the output tokens and that the decoder can provide the probability for the next token in a partially complete output sequence.

W. $P(w_i|w_1, ..., w_{i-1})$ represents the probability that w_i is the next token in the sequence, given that the first $i-1$ tokens in the sequence are w_1 to w_{i-1}. Notice that the N-th root normalized the value based on the length of the sequence so that the value is not necessarily smaller for longer sequences.

Although perplexity can provide an estimate of a model's performance on a variety of tasks, perplexity does not always align with a language model's performance on every task. This difference necessitates task-specific evaluation methods. We describe methods specifically for evaluating reading comprehension question generation in Section 2.7.

2.3 Autoregressive Text Generation

Language models predict probabilities about how likely a token is to appear at a certain point in the text. However, language models cannot be used to generate text directly; an algorithm must be used to convert the language model probabilities to tokens. The most common way of converting these probabilities to generated text is with *autoregressive text generation*. In autoregressive text generation, tokens are generated one after another, starting with the first token in the sequence. Each token is generated by conditioning on previous tokens to obtain the probabilities of each possible next token. Then, the next token in the sequence is selected by sampling from this probability distribution. Generation typically continues until a specific token or sequence of tokens is reached, indicating that the generation should stop or until a maximum length is reached. We now discuss several methods of sampling text using a language model probability distribution.

2.3.1 Greedy Search

The simplest autoregressive text generation algorithm is greedy search. In greedy search, the most probable next token is selected for each next token until the end of the sequence. This approach is computationally inexpensive but can often lead to suboptimal and repetitive sequences. See Figure 2.3 for an example of greedy search.

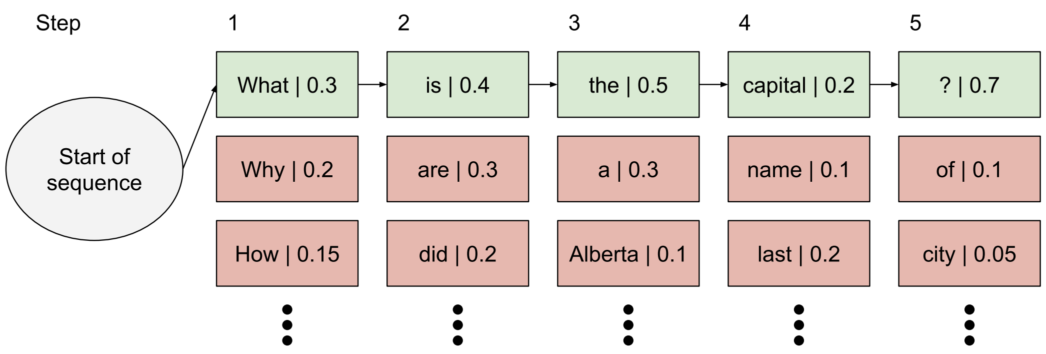


Figure 2.3: Greedy Search. The green boxes show the selected tokens at each step. The red boxes show tokens that cannot possibly be selected. Notice that greedy search always selects the highest probability token at each step.

2.3.2 Beam Search

One of the most commonly used autoregressive text generation algorithms is beam search. Beam search starts by selecting the most probable N starting tokens, where N (beam size) is a hyperparameter. Each of these tokens will become the start of a sequence. During each iteration, the most probable N next tokens are added to the end of each sequence, resulting in N^2 total sequences. Then, the N^2 sequences are reduced to the N most probable sequences for use in the next iteration. This continues until the end of the sequence is reached for each of the N sequences. Beam search generally produces higher quality generated text than greedy search. However, it is more computationally expensive and requires more memory to run. See Figure 2.4 for an example of beam search.

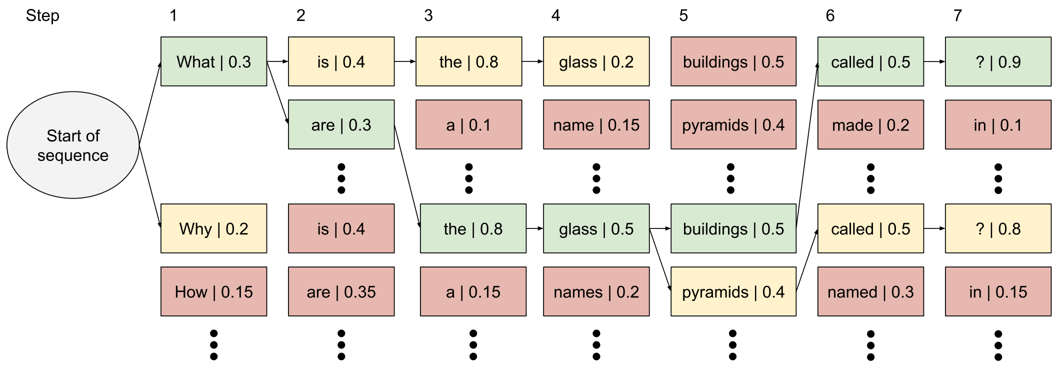


Figure 2.4: Beam Search with an N (beam size) value of 2. The green boxes show the tokens selected for the final sequence at each step. The yellow boxes show explored branches that were cut because they had a lower probability than the selected branch. The red boxes show tokens that cannot possibly be selected. Notice that there are always 2 branches active at the same time in this example. This is because the beam size is set to 2.

2.3.3 Nucleus Sampling

Nucleus sampling [15] has recently emerged as an alternative to beam search since it is less computationally expensive, less memory intensive, and can produce more diverse sequences than beam search while maintaining similar text quality. In Nucleus sampling at each step, all possible next tokens are sorted into a list with the highest probability tokens first. Next, a set of tokens is constructed by adding each token in the ordered list until the sum of the probabilities exceeds a hyperparameter p. Then, the next token is sampled based on the redistributed probabilities of tokens in the set. This process continues until the end of the sequence. See Figure 2.4 for an example of Nucleus Sampling.

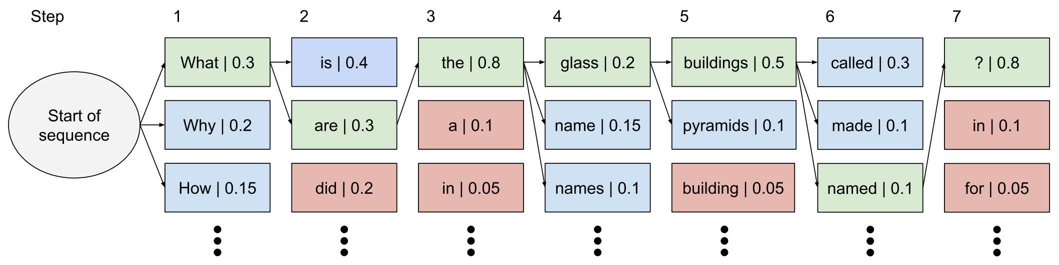


Figure 2.5: Nucleus Sampling with a p-parameter of 0.6. The green boxes show the tokens selected for the final sequence at each step. The blue boxes show tokens that were possible to be sampled. The red boxes show tokens that cannot possibly be sampled. Notice that if the probability of tokens is already greater than or equal to 0.6, it is not possible to sample the next most probable token.

2.4 Prompting

One way to control the text generated by a language model is by *prompting* the model. A model is prompted by appending tokens to the input, typically giving the model instructions, or providing a template for the model to fill in. For example, a model can be prompted with "The capital of Canada is" and then used to generate the next token in order to determine Canada's capital city. Prompting allows a language model to accomplish tasks with few or sometimes no previously seen examples [8].

2.4.1 Soft-Prompting

Prompting has traditionally been done by appending tokens from a model's vocabulary to the input sequence.

For many types of language models, it is also possible to prompt models using vector representations of tokens. These fixed length vector representations are called *embeddings*, and for many language models, including the ones used in this thesis, there is an embedding for each token in the vocabulary. Providing the embeddings for each token in a sequence to the model is equivalent to providing the tokens directly.

However, the embeddings provided to the language model do not need to correspond to tokens in the model's vocabulary. Instead, a model can learn prompt embeddings from examples of the desired output. This method of learning prompt embeddings is referred to as soft-prompt tuning, and the prompt produced is referred to as a soft-prompt [20, 21, 29]. Soft-prompting allows for better control over generated sequences, compared to using "hard" tokens from within a model's vocabulary [21, 29]. Soft-prompt methods typically only train the soft-prompt embedding weights while leaving the other weights frozen. However, as we explore in this thesis, it is also possible to train soft-prompts at the same time as other model weights.

2.5 Reading Comprehension Questions

Many children struggle with reading comprehension, an important skill for future success [37]. One way for children to evaluate and improve reading comprehension skills is by completing reading comprehension questions. However, quality questions require significant effort by teachers to create.

2.5.1 Question Generation

One potential solution to this problem is to create the questions automatically. Reading comprehension questions can be generated automatically without significant teacher effort by formulating comprehension question generation as a sequence-to-sequence text generation task [12, 26]. Before formulating the problem, it is important to understand the concepts of *contexts* and *question types* formulated below.

Contexts

The *context* is the piece of text that a student reads and must understand in order to answer a reading comprehension question. Some previous work uses story and context interchangeably [12, 43]. However, in this thesis, we use the word *context* because the context is not limited to just stories. The context can also be news articles, blog posts, excerpts, or any other text that a student can read and answer questions about.

Question Skill Types

Additionally, reading comprehension questions can measure a reader's ability to understand different aspect of the context. These abilities are referred to as skills and can be grouped together into types. For example, understanding causality, order of events, and character beliefs are all examples of skill types that reading comprehension questions can target. We refer to these question types as *skill types* in this thesis. See Table 2.1 for some examples of these skill types from the QuAIL dataset [34].

Skill Type	**Example Question**
Belief states	Why does the author think the technician inspected the wiring for free?
Causality	Why was it blazing hot?
Character identity	Who said there was nothing wrong?
Entity properties	What is probably true about the technician?
Event duration	How long after the electrician left did the author eat dinner?
Factual	Where was the fuse box located?
Subsequent state	After the end of the story, the author probably:
Temporal order	When did the technician give the author a $10,000 estimate?

Table 2.1: Skill Type Examples from the QuAIL Dataset. All questions pertain to the same short story context.

Task Formulation

We formulate the task of automatic *targeted* comprehension question generation as follows: given as input 1) a textual *context* C and 2) a desired *skill type* S for a question [12]. The goal of the task is to create a question of the type S that is answerable based on context C. The question is not considered answerable, if the required information to answer the question is missing from the context or the question includes information contradicting the context.

This formulation differs from *untargeted* comprehension question generation [13, 28, 42, 46], which does not take a skill type as input, and from *answer-based* comprehension question generation, which requires an answer in the input [26, 28, 42, 44, 47]. See Figure 2.6 for an example input and output for this task.

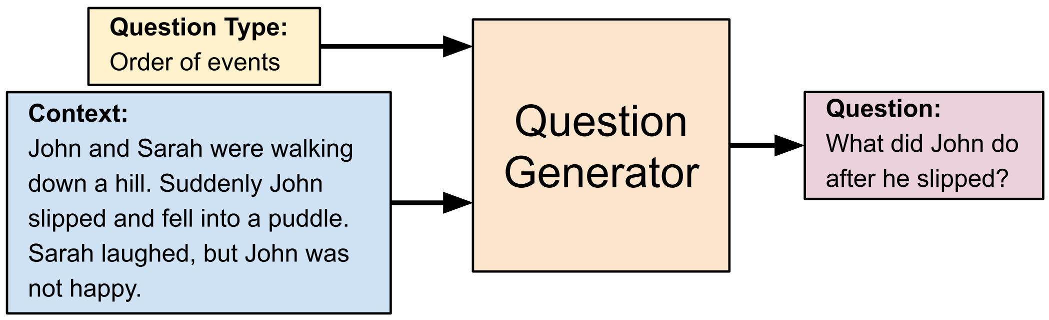

Question Type:
Order of events
Context:
John and Sarah were walking down a hill. Suddenly John slipped and fell into a puddle. Sarah laughed, but John was not happy.
Question Generator
Question:
What did John do after he slipped?

Figure 2.6: Targeted Reading Comprehension Question Generation Task Input/Output Example. Notice that the questions is specific to the context, the information required to answer the question is contained in the context, and the question matches the requested question skill type.

2.6 Question Generation Methods

There has been some previous work looking at automatically generating questions from a context. In this section, we summarize this past work and compare it to our work in this thesis.

Early research on question generation focused on template-based methods [10, 24]. This template-based approach allows for implicit control over the types of questions by selecting templates for a specific type of question. However, these techniques are less flexible compared language model based methods and additionally require significant manual effort to create.

More recently, question generation methods focused on generating questions by conditioning on preexisting answers [26, 28, 40, 42, 44, 47]. This approach is not practical for many applications. For example, in education software, it is impractical for teachers to need to create a list of potential answers for a given context or story before questions can be generated. In this thesis, we focus on the more difficult and general problem of generating questions without a provided answer, taking only the context and a requested question type as input.

The work most closely related to our own is that of Ghanem *et al.* [12], described in Section 3.1.1. They propose a method that uses tokens from the model's vocabulary to control the types of generated questions. In this work, a prompt is manually created for each question type, taking the form of a name or description given to the question type. This is in contrast to our proposed SoftSkillQG method, which automatically learns a prompt for each question type during training without needing to manually specify the text of the prompt.

Wang *et al.* [39] also explore controlling the types of questions generated using the name of the skill as a prompt to control the question type. However, they break the task of question generation into three subtasks. The first subtask is to determine a topic from the context. The second subtask is to determine what specific knowledge from the context should be tested in the topic selected from the first subtask. The final subtask is to generate a question to test the knowledge from the second subtask. This approach works well for generating questions where the answer is included in the context, but the second subtask makes it difficult to generate more inferential questions about the context, unlike our proposed method that have no such restriction.

Cao and Wang [9] use both a question-type name and a question-type template in a prompt to get a model to generate a given question type. However, this forces questions to conform to a list of templates, potentially leading to highly repetitive questions. This requirement may have little effect on the utility of the open-ended questions explored by the authors but is undesirable for reading comprehension questions where engagement is important.

Zhou *et al.* [47] group questions based on the first word in the question instead of by skill type. This allows them to set the first token in the output sequence to control the question type. This approach cannot be applied to our setting since multiple skill types can share the same prefix and each skill type is not limited to a single prefix.

Additionally, the work of Zhao *et al.* [46] introduces a system that generates questions by summarizing sections of a story. They then generate questions based on those summaries. Their system predicts question-type distributions and uses them as input to generate questions. These distributions could be adjusted to control question types. However, this control is not explored in the work of Zhao *et al.* [46].

2.7 Question Evaluation

Generated questions are typically evaluated by comparing an automatically generated *candidate question* with a set of manually created *reference questions* that have the same input context [11, 12, 44]. This comparison is usually done with common similarity metrics used for a variety of natural language processing tasks. BLEU [27], METEOR [5], and ROUGE [22] are among the most popular of these metrics.

When creating reading comprehension questions, it is useful to generate a variety of questions to help students improve a variety of skills. It also takes a student less time to answer multiple questions for a single context than to answer questions for multiple contexts, increasing the number of questions a learner can complete in a given amount of time and potentially allowing a student to learn more in the same amount of time.

However, evaluating a set of generated *candidate questions* is more challenging than evaluating a single question because common similarity metrics used for other natural language processing tasks are only made to handle a single candidate input. Naive solutions using these metrics often encourage low diversity sets of questions, which are undesirable. However, some methods have been proposed to take into account both similarity to reference questions and the diversity of text generated [2, 11, 35].

In Chapter 6, we perform a first of its kind comparison of a variety of these evaluation methods. Specifically, we compare how well each of evaluation method aligns with multiple aspects of human judgement and how well each captures the diversity of a set of questions.

Chapter 3

Models For Controlling Reading Comprehension Question Types

In this chapter, we describe a variety of existing techniques and present a new technique, SoftSkillQG, that can be used to generate targeted reading comprehension questions. We then describe the process that we use to train the models and generate new questions. Finally, we compare SoftSkillQG and the baseline models using three experiments measuring 1) perplexity, 2) five automatic question evaluation methods, and 3) three human evaluation aspects.

3.1 Baselines

We use five baseline models to generate reading comprehension questions. Three baseline models allow for control over the types of questions generated, and two do not. We include this final baseline to determine if there is degradation in the quality of questions as a result of the lack of control over the types of questions generated.

3.1.1 T5 WTA

To establish a strong baseline, we use the *T5 WTA* model introduced by Ghanem *et al.* [12] for the targeted reading comprehension question generation task.[1] T5 WTA uses a fine-tuned T5 model to generate questions and uses

[1]We obtained the same performance level as reported by Ghanem *et al.* [12] using the code provided. However, we could not replicate the exact values reported, likely due to a

a predefined prompt using tokens from the model's vocabulary to indicate the desired question skill type to the model. The question type can then be controlled by changing the prompt used to generate a question. Note that there is exactly one prompt for each question type. A full list of the prompts used for T5 WTA is available in Appendix A.6. See Figure 3.1 for an illustration of the T5 WTA model architecture.

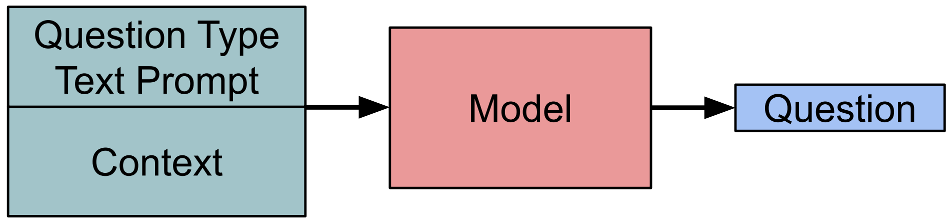


Figure 3.1: T5 WTA model architecture. The model takes two concatenated textual inputs: the question type prompt and the context. The model then produces a question of the specified type as output.

3.1.2 Soft-prompting

Soft-prompting is a method that has been used to control the output for a variety of tasks [20]. As described in Section 2.4.1, in soft-prompt tuning, the model weights are left frozen while fixed length prompt embeddings are trained. We use 100 distinct embedding tokens for each question skill type to specify the soft-prompt, since using multiple embedding tokens in a prompt has been shown to perform better than just using a single embedding token [20, 21]. We initialize the soft-prompt embeddings using token representations using the embeddings of randomly selected tokens from the model's vocabulary since this has been shown to perform better than initializing from a random distribution [20, 21]. The question type is then controlled by switching which prompt is used, similar to T5 WTA. See Figure 3.2 for an illustration of the soft-prompting model architecture.

difference in hardware or randomness during training.

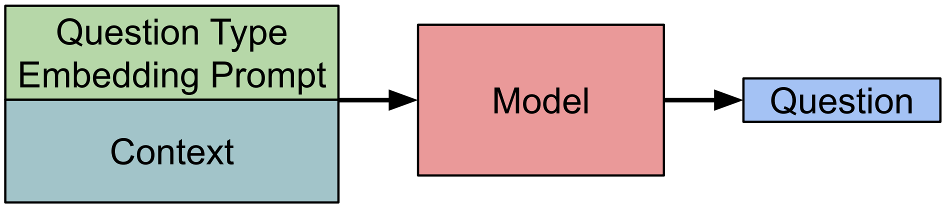


Figure 3.2: Soft-prompting model architecture. The model takes two inputs: a learned embedding representation of the question type and the context. The model is then trained to produce a question of the specified type as output.

3.1.3 Independent Mixture of Experts (IMoE)

Skill-targeted question generation can also be accomplished by training independent models for each question type, referred to as a mixture of experts [16]. For this thesis, we refer to this method of question generation as the *Independent Mixture of Experts* (IMoE) model. For IMoE, we train each independent model using data filtered to only a single skill type. Then, we can control the skill type by generating questions using a specific model trained to only generate questions of a given type.

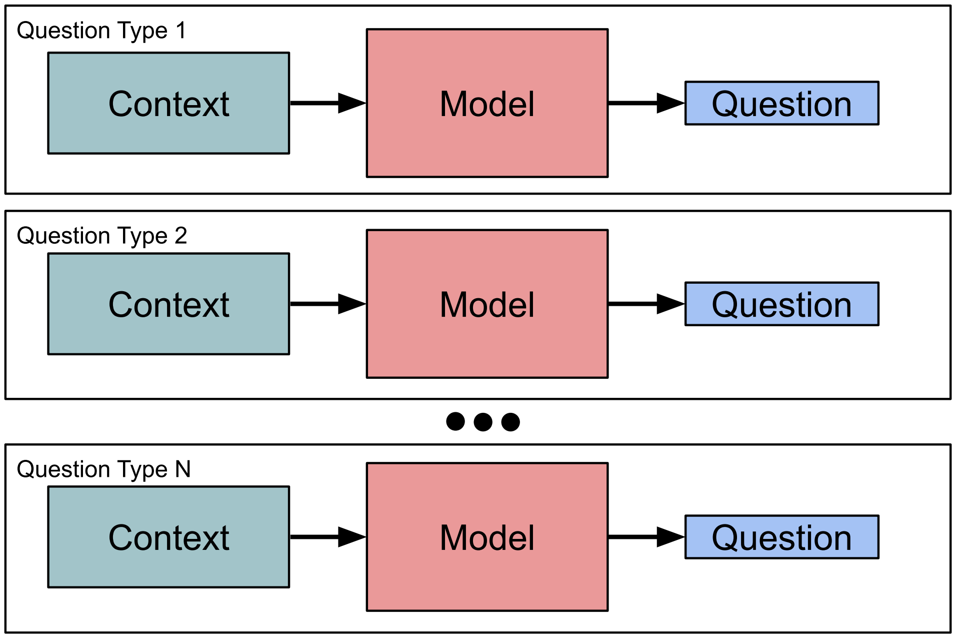


Figure 3.3: Independent Mixture of Experts (IMoE) architecture. There is a separate model for each question type that takes only a context as input and provides a question as output.

We use this model as a baseline because it does not require prompts but allows control over the types of questions generated. However, IMoE has a much larger number of parameters compared to the other models presented. See Table 3.1 for the number of parameters for each model and Figure 3.3 for an illustration of the model's architecture.

3.1.4 No Control

It is possible that targeting a skill type negatively affects the quality of questions generated. To measure the impact of targeting, we trained a single T5 model [30] without any prompt. We refer to this model as the *No Control* model. The No Control model is not conditioned to generate questions of any particular skill type. This lack of restriction could allow the model to generate more questions of types well suited to the context, potentially leading to high-quality questions. See Figure 3.4 for an illustration of the No Control model architecture.

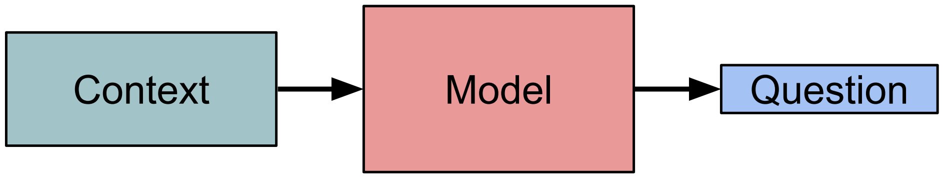


Figure 3.4: No Control model architecture. There is a single model that takes only a context as input and provides a question as output. Note that this model does not have a way to specify the question type.

3.1.5 Shared soft-prompt

To determine if the control over question types specifically benefits soft-prompting methods, we also train a soft-prompt based model without the ability to control the type of questions generated. We refer to this model as the *Shared soft-prompt* model. Shared soft-prompt is trained with the same soft-prompt for all question types and all weights are left unfrozen. See Figure 3.5 for an illustration of the Shared soft-prompt model architecture.

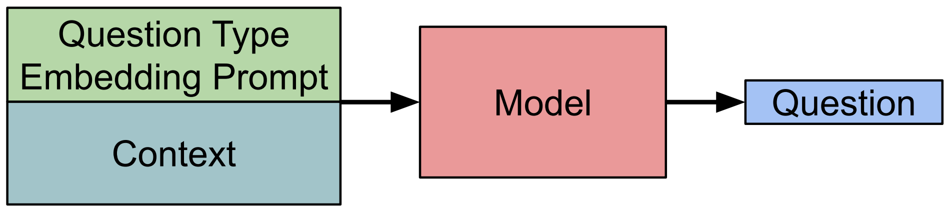


Figure 3.5: Shared soft-prompt Architecture. The model takes two inputs: a learned embedding representation of the question type and the context. The model is then trained to produce a question of the specified type as output.

3.2 Proposed Model: SoftSkillQG

In this section, we present, SoftSkillQG, our proposed model for targeted reading comprehension question generation. For *SoftSkillQG*, we use soft-prompts to control the types of questions generated by a T5 model [30]. We follow a similar approach to the soft-prompting model of Lester *et al.* [20] described in Section 3.1.2 and tune the prompt embeddings of the model to control the types of questions generated. In contrast with Lester *et al.* [20], who only trained the embedding weights, we found in early testing that training on all of the model weights resulted in slightly better performance. This method is also similar to T5 WTA described in Section 3.1.1 but uses tuned embedding tokens instead of manually selected tokens from the model's vocabulary to control the question types generated by this model.

Similar to other prompting methods, we condition the model to generate a requested skill type by prepending the input with a soft-prompt [12, 20]. This approach prevents the prompt from being truncated if the input sequence is longer than the model will allow. See Figure 3.6 for an illustration of SoftSkillQG's architecture.

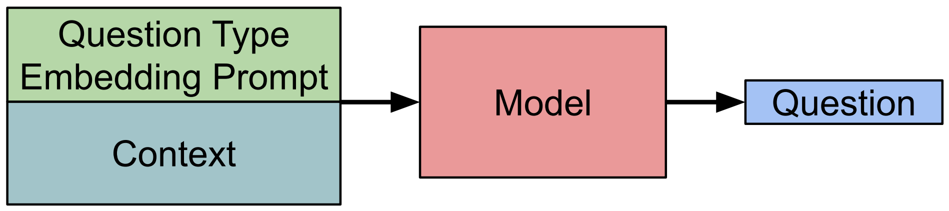


Figure 3.6: SoftSkillQG model architecture. The model takes two inputs: a learned embedding representation of the question type and the context. The model is then trained to produce a question of the specified type as output.

Method	**Prompt**	**Number of models**	**Parameters**
SoftSkillQG / Soft-prompting	20 embedding tokens	1	$B + Q \times E \times 20$
T5 WTA	A variable number of tokens from the model's vocabulary + 1 separator token	1	B
IMoE	N/A	Q	$B \times Q$
No Control	N/A	1	B
No Control with soft-prompt	20 embedding tokens	1	$B + Q \times E$

Table 3.1: Summary of Models. B is the number of parameters in the base model, E is the number of parameters for a token embedding, and Q is the number of question skill types.

3.3 Question Generation & Selection Process

In this section, we outline the steps used to generate and filter questions so that we have an equal number of candidate and reference questions for each context. This process aims to select the best questions across all available skill types for each question generation model.

In Figure 3.7, we show an overview of our process, which has 5 main steps: Generation, Scoring, Ranking, Deduplication, and Selection.

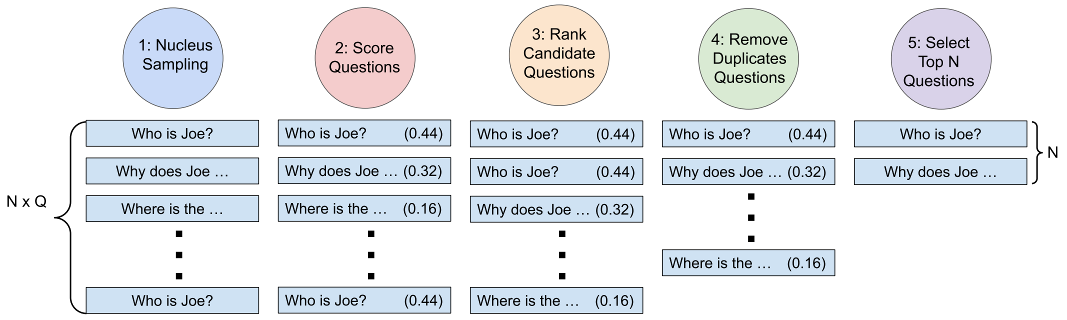


Figure 3.7: Question Selection Process. For each question generation model, we generate a set of $N \cdot Q$ questions where N is the desired number of questions to generate, and Q is the number of question skill types. Question scores are the average probability of each question token. We rank questions by score, remove duplicate questions, and finally select the top N questions.

3.3.1 Generation

To generate the questions, we perform Nucleus Sampling [15], tuning p for each model independently. We use Nucleus sampling instead deterministic auto-regressive text generation methods to allow for a diverse set of questions to be generated.

3.3.2 Scoring

During generation, we record the softmax probability of each token in the generated questions. We then score candidate questions based on the mean token probability over all tokens. We experimented with other methods for scoring questions before ranking, including median scores, and predicting question quality. However, we found simply scoring by mean token probability to be the most effective.

3.3.3 Ranking

Next, we rank questions in descending order by the scores obtained in the previous step. This ranking is later used for the deduplication and selection steps.

3.3.4 Deduplication

After ranking, we group questions into duplicate groups after removing excess whitespace. Then, we keep the questions from each group with the highest rank.

3.3.5 Selection

Finally, we select the top N questions from the set of deduplicated questions based on their ranking. We use this final set of N questions as the set of *candidate* questions in our evaluation.

3.4 Human Evaluation

We use human evaluation to determine if our automatic evaluation models align with human judgement and to accurately determine the quality of generated questions. We follow a similar process to Ghanem *et al.* [12] and perform human evaluation using Amazon Mechanical Turk to obtain annotations. We only allow annotators from countries where the majority of the population are native English speakers so that the annotators have a good understanding of each story [12]. We also require annotators to have a *Masters Qualification*, which is given to workers with a history of high success in correctly completing tasks on the platform [12].

Each annotator evaluates each question based on 3 different aspects: Answerability, Fluency, and Context Specificity.

Answerability

The first aspect, *Answerability*, determines if a question is answerable [4, 12, 14]. For example, if a question contains incorrect information or if the required information to answer the question is missing from the context, then the question is not answerable. For Answerability, we ask annotators to label each question as either *answerable* or *not answerable.*

Fluency

Second, *Fluency* determines if the question is easy to read and understand. This also encompasses whether a question is grammatically correct. For *Fluency*, we ask annotators to rate each question on a Likert scale (from 1 to 5).

Context Specificity

We also measure a new aspect, *Context Specificity* which determines if a question is relevant and specific to the presented context. For *Fluency*, we ask annotators to label each question as either *context-specific* or *not context-specific*. See Table 3.2 for examples of questions with high and low rating for each aspect.

We use three annotators to ensure high quality annotations for each aspect [12]. Each annotator is presented with multiple questions for the same story to improve efficiency. We take the most popular annotation across the three annotators for *Answerability* and *Context Specificity*. Since Fluency is numeric, we take the average rating across the three annotators instead of the most popular.

In Section 3.6.3, we use the human evaluation methods described in this section to compare automatically generated questions. Details on how the annotated questions are generated and sampled can be found in that section.

Context	John and Sarah were walking down a hill. Suddenly John slipped and fell into a puddle. Sarah laughed, but John was not happy.		
Aspect	**Answerability**	**Fluency**	**Context Specificity**
High Rating	What happened after John fell into the puddle?	Who is Sarah?	Why did Sarah laugh?
Low Rating	Where is John's rubber boot?	Who is the chacter off Sarah?	What can you conclude about the characters?

Table 3.2: Examples of questions with high and low ratings for each of the human evaluation aspects.

3.4.1 Human Evaluation Ethics

Since we are using human annotators, it is also important to consider the ethical implications of collecting the annotations. We estimated the time that it would take each annotator to complete an annotation and price the annotation tasks to pay our annotators a fair and living wage. We estimated that it would take 4 minutes to complete each annotation task and set the price to complete each task to be \$1.50 USD. This would make the earnings for a worker approximately \$22.50/hour. In addition, we required consent from each annotator, allowing us to present the data collected in this thesis. Additionally, we ensured that annotators were anonymized with no personally identifiable information. Lastly, we obtained approval from an ethics review board before performing any human evaluation.

3.5 Training Question Generation Models

In this section, we describe the datasets, hyperparameters, similarity score, implementation, and training details for our experiments.

3.5.1 Datasets

We evaluate our models using two English datasets with specified skill types. Every question in both datasets has an assigned question skill type that targets a particular reading skill, but the datasets have different sets of question skill types.

QuAIL Dataset

The *QuAIL* [34] dataset, contains approximately 15,000 questions, each with a short context and a question skill type. The contexts span a variety of domains, including fiction, news, blogs, and online story posts. The dataset has 8 question types, each targeting a different skill. QuAIL has a variable number of questions for every context, but each context has at least one question of each type. There are approximately the same number of questions of each type in the dataset, which incentivizes diversity in the generated questions.

We follow the same training, development, and test set split outlined by the dataset authors. However, we exclude unanswerable questions from the dataset since our focus is generating skill-targeting questions.

SBRCS Dataset

The Story-Based Reading Comprehension Skills (*SBRCS*) Dataset [12] contains skill-targeting reading comprehension questions to access children's knowledge of stories. SBRCS has a variable number of questions and skill types for each context. There are approximately 4,000 questions, which we split into training, development and test sets. SBRCS has 9 types of questions, each targeting a different skill, but the number of questions of each type is not evenly distributed.

3.5.2 Hyperparameters

We tune the learning rate and the Nucleus Sampling p hyperparameter for each of our experiments. For learning rates, we test values of {*5e-4*, *1e-4*, *5e-5*, *1e-5*, *5e-6*}. We use early stopping a patience values of 3 epochs with a maximum of 100 epochs. We then take the best model based on the development Multi-METEOR score. Finally, for the p hyperparameter of Nucleus Sampling [15], we test values from 0.1 to 0.9 in 0.1 increments. We select the best hyperparameters based on their performance on the development set before performing our final evaluation on the test set. The optimal hyperparameters for each model and dataset are available in Appendix A.1.

3.5.3 Similarity Scores

We use METEOR as our similarity score since it aligns more closely with human judgement on question generation tasks than other methods like ROUGE-L [22] and BLEURT [17, 36]. We use NLTK's [6] implementation of METEOR and NLTK's *word_tokenize* to tokenize sequences before scoring.

3.5.4 Implementation

For each model described in Section 3.2, we use a T5 [30] model initialized with the weights of *T5 Large LM Adapted*, which has been on first on fixing corrupted tokens and then predicting the next token in a sequence.[2] We use this T5 based model because T5 is typically only trained to fixed corrupted tokens [30]. Since it takes language models a long time to adapt from token corruption tasks to next token prediction tasks like question generation, using a model like T5 Large LM Adapted can lead to higher performance [20]. We use the *large* version of this model since it performed about the same level as larger models while outperforming smaller models in early testing. For training and sequence generation, we use the implementation provided by the HuggingFace transformers library [41] and extend the utilities provided by the Simple Transformers [31] library. We use Asai *et al.* [3]'s implementation of soft-prompting for our baseline. Our full implementation is available at `https://github.com/sazzy4o/thesis-code`.

3.5.5 Training

Here, we outline the steps that we use to train SoftSkillQG and each baseline model described in Section 3.1 with a context as input and a reference question as the expected output. First, if a given model uses a prompt, it is added to the beginning of the context. Second, we generate and filter questions using the methods we will describe in Section 3.3 to have an equal number of candidate and reference questions. Third, we evaluate the models on the development set using perplexity. We then repeat these first three steps for each hyperparameter value and select the model with the optimal hyperparameter values as determined on the development set. Finally, we evaluate the model on the test set and report the test performance.

For a fair comparison of each model, none have additional pretraining beyond the T5 Large LM Adapted checkpoints. However, additional pretraining on questions from other datasets may improve the performance of these models

[2]Available at `https://huggingface.co/google/t5-large-lm-adapt`

Model	**Mean Perplexity**	**Std. Deviation Perplexity**
T5 WTA	**7.0014**	0.2244
Soft-prompting	10.3380	0.7241
IMoE	7.2943	**0.0435**
No Control	8.1191	0.3306
Shared soft-prompt	7.8165	<u>0.0497</u>
SoftSkillQG	<u>7.2346</u>	0.1660

Table 3.3: Mean and standard deviation of perplexity values obtained for each model on the QuAIL test set. Lower values indicate that the model probabilities are better aligned with the text in the test set. Perplexity values range from 1 to $+\infty$. Best is shown in **bold**, and second best is <u>underlined</u>.

[12].

3.6 Experiments

We ran 3 experiments to determine how SoftSkillQG performs against the baseline generation models described in Section 3.2. First, we measure the perplexity each model on the QuAIL dataset. Second, we evaluate each model using the automatic evaluation scores evaluated in Section 6.2. Finally, we perform a human evaluation to measure the answerability, fluency and context-specificity of questions generated by each model.

3.6.1 Experiment 1: Perplexity Evaluation

To evaluate the models outlined, we performed an experiment to determine how well each model performs at language modeling using perplexity. We started by first training and hyperparameter tuning each of the models introduced in Section 3.2, following the training steps outlined in Section 3.5.5. Next, we calculated the average perplexity for each question in the QuAIL test set using each model. We performed this process three times with different seeds and average the perplexities to reduce the impact of randomness. We present the results of this experiment in Table 3.3.

Dataset	Model	Multi-METEOR	Best Ref.	Cartesian	Jaccard4	FBD
QuAIL	T5 WTA	**31.65**	**44.39**	12.59	43.07	8.99
	Soft-prompting	29.62	41.29	12.05	41.21	8.92
	IMoE	30.72	44.13	**12.78**	42.06	**7.90**
	No Control	30.57	39.44	11.19	**43.97**	10.47
	Shared soft-prompt	28.97	36.34	10.63	41.28	14.41
	SoftSkillQG	30.73	43.22	12.56	42.14	9.16
SBRCS	T5 WTA	14.08	15.51	11.42	18.99	31.45
	Soft-prompting	12.35	14.08	10.48	16.00	36.79
	IMoE	13.36	15.06	10.99	18.45	31.37
	No Control	**14.87**	16.74	**12.89**	18.98	36.35
	Shared soft-prompt	13.83	16.21	12.33	17.39	36.37
	SoftSkillQG	**14.87**	**16.89**	12.50	**19.12**	**29.03**

Table 3.4: Several methods of automatic evaluation applied to different question generation models. SoftSkillQG performs best or second best across all metrics for the SBRCS dataset. However, on the QuAIL dataset SoftSkillQG does not perform as well. For QuAIL, T5 WTA tends to perform best or second best across most evaluation metrics. The range of values is from 0 to 100, except for FBD, which has a range of 0 to $+\infty$. Best is shown in **bold** and second best is underlined.

3.6.2 Experiment 2: Automatic Evaluation of Question Generation Models

To help us determine which question generation model (Section 3.2) produces the highest quality questions, we performed evaluation using five different automatic evaluation methods: Best Reference, Cartesian Product, Multi-METEOR, FBD, and MS-Jaccard4. We describe each of these methods in detail in Section 6.1. First, we performed parameter tuning on the development sets independently for both the QuAIL and SBRCS datasets. Then we selected the model with the best performance on the dev set and used it to generate predictions for the test set. Finally, we computed the automatic evaluation scores for each of the generated questions and report the results in Table 3.4.

Model	**Fluency**	**Answerable**	**Context Specific**
T5 WTA	4.68	52.22%	76.67%
IMoE	**4.84**	<u>67.78%</u>	83.33%
Soft-prompting	4.7	55.56%	78.89%
No Control	4.71	**70.00%**	<u>86.67%</u>
Shared soft-prompt	4.54	54.44%	85.56%
SoftSkillQG	<u>4.73</u>	64.44%	**87.78%**
Human Created	4.69	72.22%	88.89%

Table 3.5: Human evaluation for each question generation model on the QuAIL dataset. For *Fluency* values range from 1 to 5. For *Answerability* and *Context Specificity* values range from from 0% to 100%. Best is shown in **bold** and second best is <u>underlined</u>. Notice that some question generation methods score higher than human created question for *Fluency* and *Context Specificity*.

3.6.3 Experiment 3: Human Evaluation of Question Generation Models

For our final experiment, we measured the quality of generated questions with human evaluation. We selected all 30 contexts in the QuAIL test set and sampled 3 questions per context generated by question generation model in Section 3.2 to create a set of $30 \cdot 3 \cdot 6 = 540$ questions. Then, we performed human evaluation, as described in Section 3.4. When presenting the questions to the annotators, we randomize the order of the questions and show an equal number of questions for each question type to each annotator to reduce the impact of randomness. To determine a performance ceiling, we also sampled 90 human created (gold) questions from the QuAIL test set and included them in the same evaluation. These additional human created questions bring the total number of annotated questions to 630. We show the results in Table 3.5.

3.6.4 Discussion

The results show that T5 WTA has the lowest perplexity This low perplexity indicates that the T5 WTA model assigns high probabilities to the questions in the QuAIL test set and is, therefore, well aligned with the test set. SoftSkillQG and IMoE follow closely with a slightly lower perplexity, indicating that they also align closely with the test set. The IMoE model also has a low standard deviation in mean perplexity between runs, indicating that following the IMoE training method results in a consistent model. The models without control over the question type have higher perplexities likely because they cannot utilize question type information. The original soft-prompting model also has a high perplexity because it needs to keep sub-optimal weights frozen unlike SoftSkillQG which has the ability to update all of its weights.

The proposed SoftSkillQG performs very well on the SBRSCS dataset achieving first or second on all evaluation methods. However, SoftSkillQG does not perform as well as other methods like Soft-prompting on faithfulness to a requested question type and does not perform as well as T5 WTA on the QuAIL dataset. Interestingly, the No Control model performs well on the SBRCS dataset. This may be because other methods struggle to learn question types with very limited amount of data or are forced to produce question of a certain type even when that type is not appropriate for the context. It is also important to note that only 90 questions from each model were evaluated by human annotators, so the results may vary slightly with a larger sample size.

Most models generate questions that garner *Fluency* scores close to or even above human-created questions. Automatic methods like SoftSkillQG also score very similarly to human created questions for *Context Specificity*. This suggests that there is little room to improve fluency, leaving *Answerability* as the best area to focus on for future improvement.

It is important to note that there was some disagreement between annotators. All three annotators agreed on *Fluency* for 72.2% of questions, on *Context Specificity* for 71.0% of questions, and on *Answerability* for only 46.7% of questions. Human created questions were also only annotated as answerable 72.22% of the time. This low agreement also suggests some subjectivity of *Answerability*, making it difficult to assess.

3.7 Chapter Summary

In this chapter, we presented SoftSkillQG, a new soft-prompt based method for generating reading comprehension questions and described a variety of baselines. We showed that SoftSkillQG outperforms baselines on four out of five automated metrics for the SBRCS dataset and human judgement of context-specificity on the QuAIL dataset. However, it is outperformed by T5 WTA, a method using manually crafted prompts on perplexity and automated metrics for the QuAIL dataset. To try to determine why SoftSkillQG struggles on the QuAIL dataset, we explore the effect of data set size on the performance of soft-prompt based methods like SoftSkillQG in the next chapter.

Chapter 4

Effect of Data Size on Soft-Prompting Methods

In the last chapter, we found that SoftSkillQG does not perform as well as T5 WTA on multiple measures for the QuAIL dataset. We hypothesized that the small amount of data afforded to each question type may be a contributing factor to the poor performance of SoftSkillQG and other soft-prompt based methods on the QuAIL dataset. Therefore, in this chapter, we test the effect of data size on soft-prompt based methods.

4.1 Datasets

We use two English reading comprehension datasets for our experiments in this chapter. We introduced the first dataset QuAIL in Section 3.5.1. The second dataset, SQuAD is described below.

The Stanford Question Answering Dataset (SQuAD) [33] contains over 100,000 English reading comprehension questions.[1] Each of the contexts in the SQuAD dataset is a paragraph from a Wikipedia article and provides sufficient information to answer corresponding questions. The answer to each question is contained as a span of tokens in the context.

The SQuAD dataset is partitioned into three subsets, two of which are publicly available. The training partition contains 87,599 question-answer pairs. The validation partition contains 10,570 question-answer pairs.

[1]The SQuAD dataset is available at `https://huggingface.co/datasets/squad`

It is important to note that predicting answers based on the spans of text in the context is an easier task than predicting unrestricted answers or questions since there are far fewer possible options to predict when choosing a span.

4.2 Question Type Classifier

We trained a RoBERTa-based [23] classifier to classify questions into one of the 8 skill types in QuAIL. First, we fine-tuned a pre-trained RoBERTa language model[2] to classify questions on the QuAIL training set. We then performed hyperparameter tuning on the dev set. After hyperparameter tuning, the classifier obtained 89% accuracy on the test set. The hyperparameters chosen are available in Appendix A.1.

4.3 Size Experiments

We performed two experiments to estimate the amount of data that soft-prompt based methods like SoftSkillQG require and determine if more data is required than is available for each question type in the previously explored QuAIL and SBRCS datasets.

4.3.1 Experiment 1: Effect of Dataset Size on Question Answering Performance

For our first experiment, we follow the methods of Lester *et al.* [20] and train a T5 Soft-prompt model to predict the answers to questions on the SQuAD dataset. We use the same initial model and model size as Lester *et al.* [20][3] since it has been shown to be effective for this task. We repeat the same training process using subsets of the SQuAD with different sizes: 1000, 2000, 5000, 10000, 20000, 50000, and the full 87599 training examples. The dataset sizes increase in size approximately exponentially, allowing us to the amount of data required within one order of magnitude. We use greedy search at each epoch to generate the answer to each question in the SQuAD validation

[2]Available at `deepset/roberta-large-squad2`

[3]Available at `https://huggingface.co/google/t5-xxl-lm-adapt`

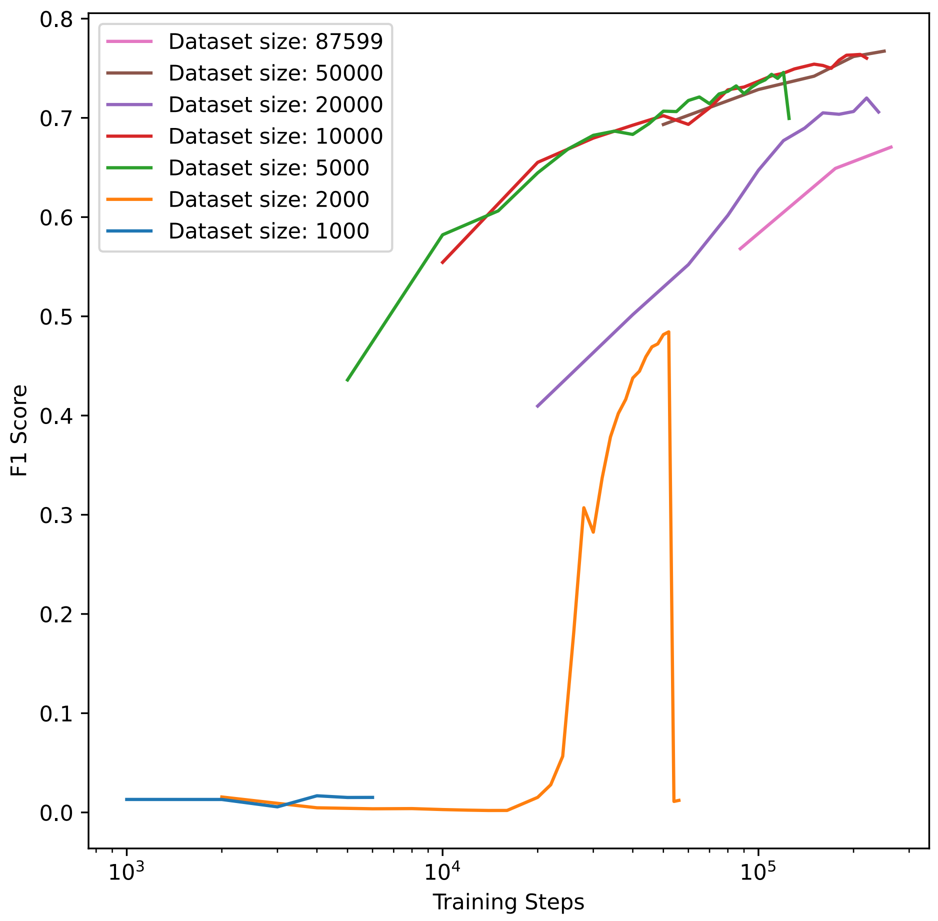


Figure 4.1: SQuAD Question Answering F1-Score Performance by Dataset Size. The number of training steps is plotted on a logarithmic scale. Notice that the soft-prompt models trained on 1000 (blue) and 2000 (orange) perform much worse than other the models.

set. We use greedy search since it is deterministic to reduce the impact of randomness. We then compute the F_1 score between the predicted answer tokens and the tokens in the true answer where the inclusion of each token is treated as a binary classification prediction. This experiment allows us to estimate the number of examples needed for question answering, which is one task where soft-prompting has previously been shown to be effective [20]. We show the results in Figure 4.1.

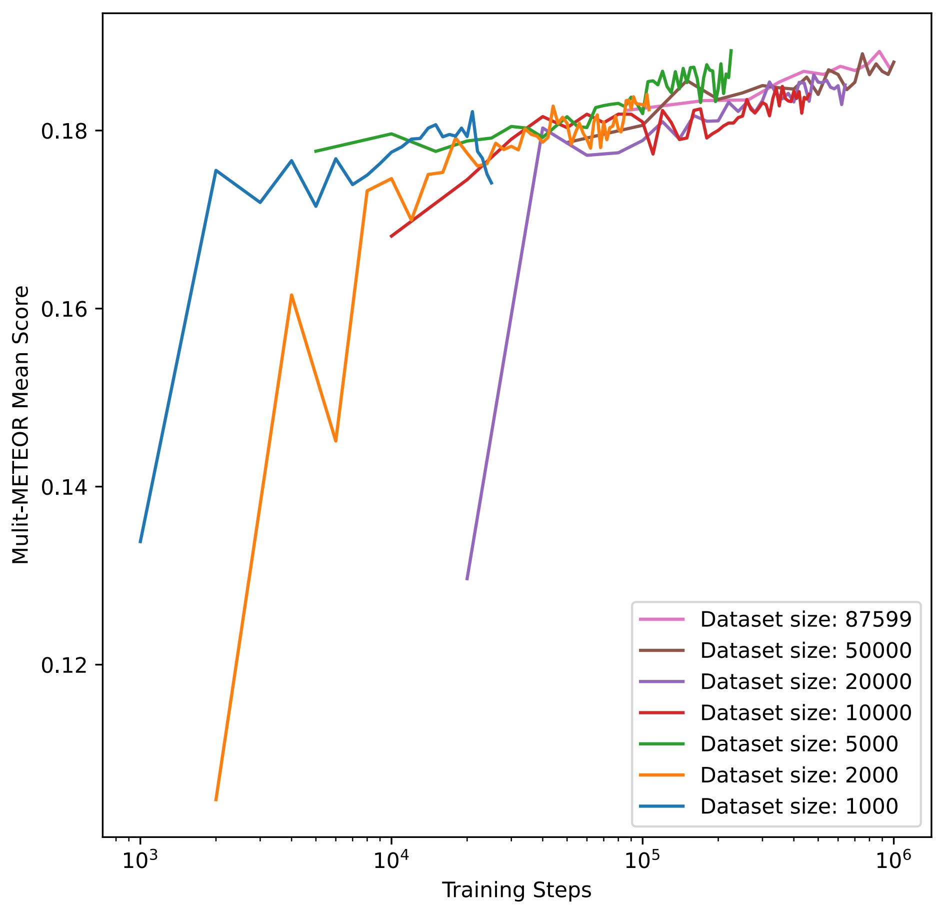


Figure 4.2: SQuAD Question Generation Multi-METEOR Performance by Dataset Size. The number of training steps is plotted on a logarithmic scale. Notice that the soft-prompt models trained on 1000 (blue) perform worse than the other models.

4.3.2 Experiment 2: Effect of Dataset Size on Question Generation Performance

In our second experiment, we follow the same procedure as the first experiment. However, instead of the model to answer questions, we train it to generate questions on the SQuAD dataset. We also use the same initial model as we used in the previous chapter in order to make the results of this experiment more comparable. Again, we test seven different sizes: 1000, 2000, 5000, 10000, 20000, 50000, and the full 87599 training examples. Similar to Experiment 1, we use greedy search to generate questions at each epoch, but we compute the mean Multi-METEOR score instead of the F_1 score. We repeated the whole training process for three seeds and plot the best seed to minimize the effect of randomness. See results in Figure 4.2.

4.3.3 Discussion

The results from the first two experiments show a notable decrease in performance when there is not enough data. In Experiment 1, about 5,000 examples are required to achieve comparable performance to models trained on more data. In Experiment 2, about 2,000 examples are required to achieve comparable performance, and 5,000 examples produced slightly better performance than much larger dataset.

Given that the QuAIL dataset has approximately 1,100 questions of each question type and the SBRCS has even fewer, the results suggest that a lack of data could be contributing to the poor performance of the soft-prompt methods on question generation.

4.4 Data Augmentation Experiment

Since the results of the previous experiments suggest that there a lack of data may be affecting SoftSkillQG performance on the QuAIL dataset, we try running an additional experiment training with additional silver training data.

4.4.1 Experiment 3: Effect of Silver Data on Question Generation Performance

Recall that in Section 3.6, we saw that T5 WTA outperformed SoftSkillQG and other soft-prompt based methods on multiple metrics for the QuAIL dataset. Therefore, for our third experiment, we want to determine if training on additional silver data can boost the performance of SoftSkillQG in scenarios where there is limited data. First, we use the question type classifier introduced in Section 4.2 to classify each question in the SQuAD dataset. Then, we use the softmax probabilities of the classifier to filter the questions to only include instances classified into a class with at least 0.8 probability.[4] The resulting

[4]We tried more restrictive filters, but they resulted in worse performance.

distribution of question types in the silver dataset is available in Figure 4.3.

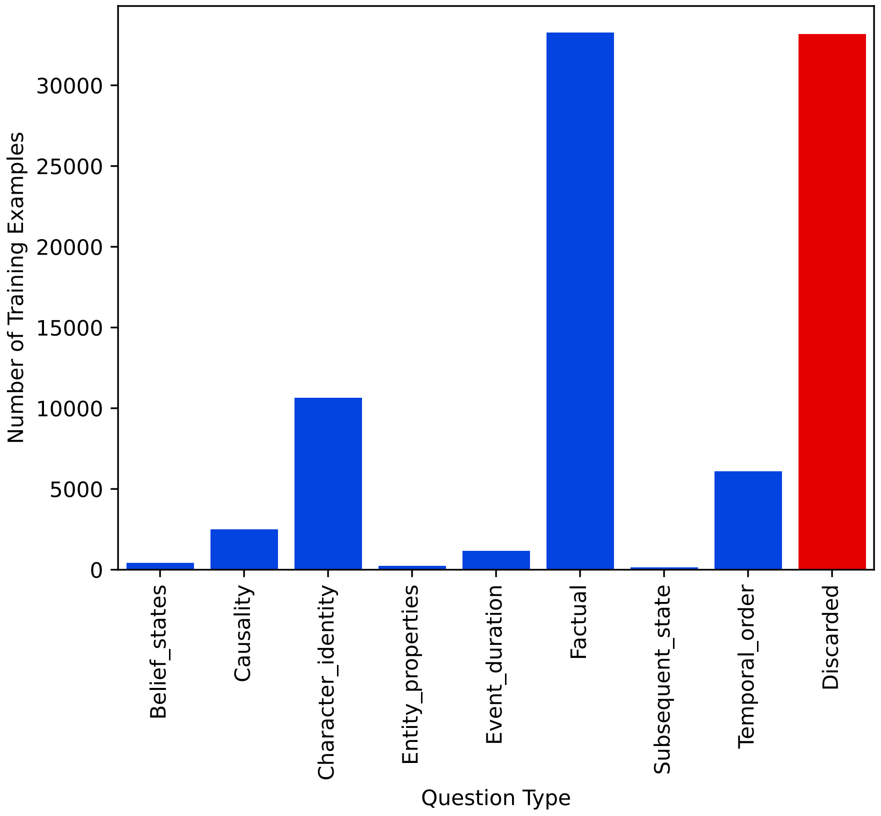


Figure 4.3: SQuAD Silver Question Type Distribution. Count of each automatically categorized question type in the SQuAD silver dataset order by number of training examples. The *Discarded* questions questions are questions that did not meet the 0.8 threshold of the RoBERTa classifier. Notice that there are more *Factual* questions than every other question type combined.

We experiment with the SoftSkillQG and T5 WTA models described in Chapter 3. We trained each model first on the SQuAD silver dataset, selecting hyperparameters based on each model's perplexity on the QuAIL train set.[5] We then trained each model on the QuAIL train set and tuned hyperparameters based on the QuAIL dev set using Multi-METEOR score after generating and selecting questions as described in Section 3.3. Finally, we generated questions for each model on the test set and compared them to the same methods finetuned only on the QuAIL training dataset. We score each

[5]SoftSkillQG performed better with all weights frozen except for the prompt weights, so this variation is used for future steps.

method based on the mean Multi-METEOR by question type. We show our results in Figure 4.4.

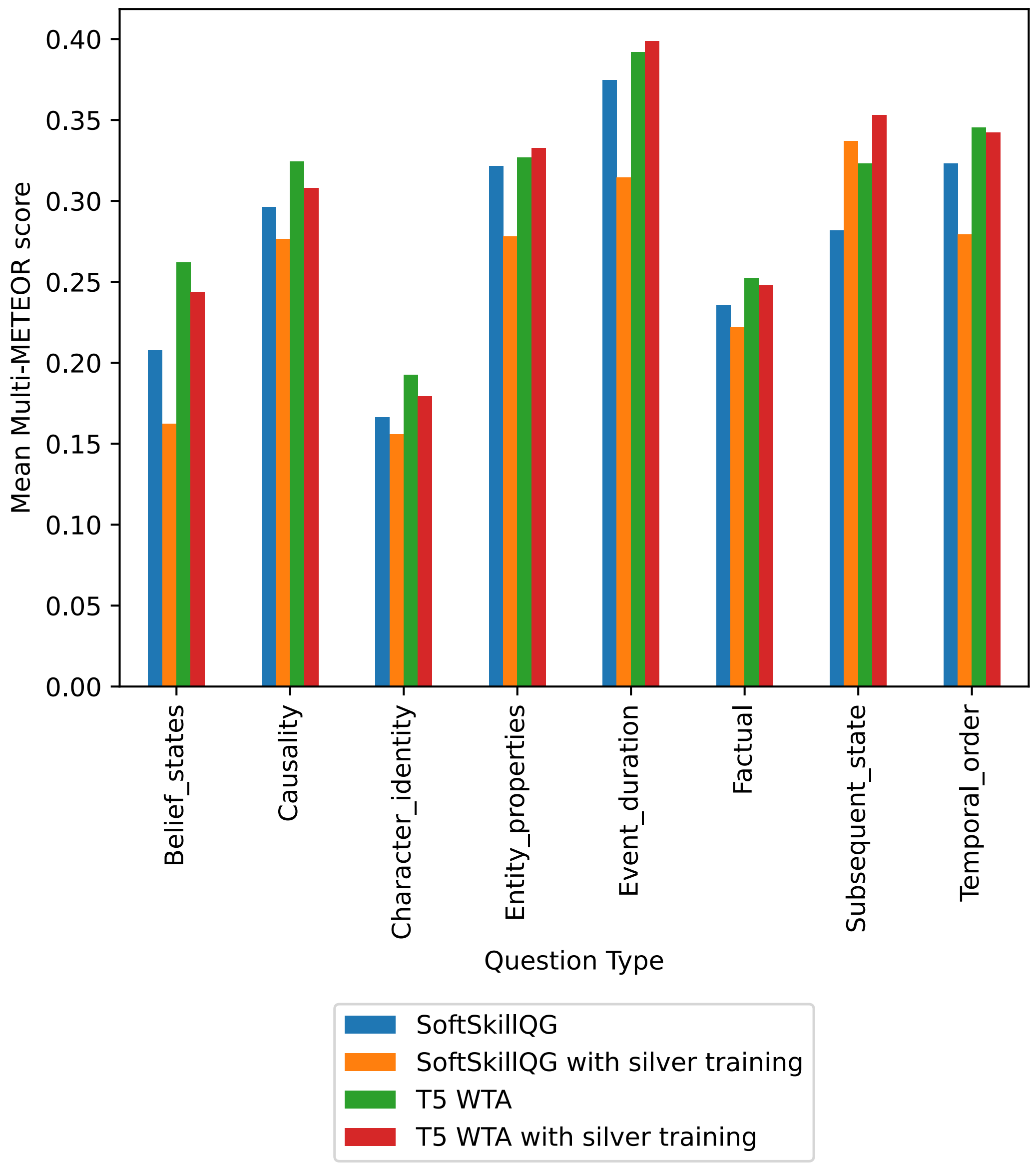


Figure 4.4: SQuAD Silver Multi-METEOR Score by Question Type. Notice that *Subsequent_state* is the only question type where SoftSkillQG sees an improvement from training on the silver dataset.

4.5 Discussion

The results in Experiment 3 show that for most question types, extra silver training did not help performance. The only question type to improve for Soft-

SkillQG was *Subsequent_state*, which has the least additional silver questions of any question type. In fact, the results show that for some types of questions training on silver data can harm performance.

However, it is important to note that SQuAD and QuAIL have some key differences, including the type of answers that are valid for each set and types of contexts used. SQuAD answers are restricted to text spans in the context, and all contexts are Wikipedia excerpts. In contrast, QuAIL answers can contain any text, not just text from the context. QuAIL also has a much more diverse set of contexts (e.g. new articles, blog posts, short stories, etc.) and it does not use Wikipedia or similar encyclopedia-excerpt contexts. Therefore, it is possible that augmenting QuAIL training with data from a more similar dataset may help improve performance. However, our results show augmenting with SQuAD data does not improve performance.

4.6 Chapter Summary

In this chapter, we performed three experiments to determine the effect of data size on soft-prompt based methods. Our results suggest that lack of data may contribute to the poor performance of soft-prompt methods on the QuAIL dataset. However, augmenting the existing QuAIL dataset with silver questions from the SQuAD dataset did not solve this data scarcity issue. As SQuAD does have some key differences with QuAIL, it is possible that training on a dataset more similar to QuAIL may still provide an improvement in performance. In the next chapter, we continue trying to determine why SoftSkillQG struggles on the QuAIL dataset by analysing the effect of SoftSkillQG's initialization on its question generation performance.

Chapter 5

Effect of Initialization on Soft-Prompting Methods

In this chapter, we continue our search to determine why SoftSkillQG performs poorly on the QuAIL dataset. We hypothesize that T5 WTA may outperform SoftSkillQG because its manually created prompts are more effective. In this chapter, we determine if the prompt initialization of SoftSkillQG contributes to this difference in performance on the QuAIL dataset.

5.1 SoftSkillQG Variants

To determine how the initialization of SoftSkillQG affects its performance relative to T5 WTA, we train five variations of SoftSkillQG to isolate the effects of initialization.

SoftSkillQG: Original

For our first variation, we use the same SoftSkillQG model we introduced in Section 3.2. The initialization variation has a prompt of 20 embedding tokens initialized using random tokens from the model's vocabulary.

SoftSkillQG: Controlling for initialization with token padding

For our second variation, we use the same SoftSkillQG model with 20 embedding tokens. However, we initialize the embeddings of the first k tokens to match the embeddings of the tokens used in T5 WTA's prompt, where k is the number of tokens in T5 WTA's prompt. Since T5 WTA has a variable number of tokens in each of its prompts, it is not possible to initialize all 20 tokens in the same way as T5 WTA. Instead, the remaining tokens in the prompt are initialized to random tokens from the model's vocabulary.

SoftSkillQG: Controlling for initialization with repeating tokens

Our third variation is nearly identical to our second variation. However, we repeat the embedding from the first k tokens to initialize the remaining tokens in the prompt instead of padding with random tokens.

SoftSkillQG: Controlling for length

For our fourth variation, we want to determine if the length of the prompt contributes to the performance of SoftSkillQG. To determine the prompt length's contribution, we initialize the prompt using random tokens, but keep the prompt lengths the same as the prompts for T5 WTA.

SoftSkillQG: Controlling for length and initialization

For our final variation, we control for both the length of the prompt and prompt initialization by initializing the soft-prompt embedding using T5 WTA prompts. Since the weights of this variation start training the same as T5 WTA, it allows us to determine whether the soft-prompts themselves are hurting performance.

5.2 Experiment

For our experiment, we train and compare models outlined in Section 5.1 and T5 WTA outlined in Section 3.1.1. We followed the same training procedure as outlined in Section 3.5.5 and training on the QuAIL dataset to generate targeted reading comprehension questions with three random seeds to reduce the effect of randomness. We then computed the perplexity for each model averaged over all seeds. We present the mean and standard deviation perplexity of each variation in Table 5.1.

Model	**Prompt Length**	**Prompt Init**	**Mean Perp-lexity**	**Std. Dev. Perp-lexity**
T5 WTA: Original	Manual	Manual	**7.0014**	0.2244
SoftSkillQG: Controlling for initialization (with repeating tokens)	20	Manual + Repeat Pad	<u>7.0056</u>	0.1300
SoftSkillQG: Controlling for length & initialization	Manual	Manual	7.0912	0.2587
SoftSkillQG: Controlling for initialization (with random padding)	20	Manual + Random Pad	7.1604	0.1682
SoftSkillQG: Original	20	Random	7.2346	0.1660
SoftSkillQG: Controlling for length	Manual	Random	7.3153	0.2630

Table 5.1: Effect of Initialization on SoftSkillQG Perplexity. Mean and standard deviation perplexity of T5 WTA and five different initialization variations of SoftSkillQG across three training seeds. Perplexity values range from 1 to $+\infty$. Best is shown in **bold** and second best is <u>underlined</u>.

5.3 Discussion

The results show that initialization has a substantial effect on the performance of SoftSkillQG. Using the same initialization as T5 WTA and then repeating tokens allow SoftSkillQG to achieve nearly the same level of performance as T5 WTA. The other two variations that use T5 WTA prompts for initializations both achieve perplexity within one standard deviation of T5 WTA. This improvement suggests that soft-prompting methods are highly sensitive to their initialization. It is possible that the initialization is so important here in part because the amount of training data is so small, which does not provide enough time and data for prompt tuning. The results also show that shortening the prompt can also hurt performance, which is consistent with previous work [20].

5.4 Chapter Summary

In this chapter, we showed that the prompt initialization of each model accounts for the of majority of the difference in performance between T5 WTA and SoftSkillQG. In fact, SoftSkillQG can achieve comparable performance to T5 WTA if we initialize the soft-prompt embeddings using T5 WTA then repeating embedding until all embeddings are initialized. In the next chapter, we compare a variety of methods for evaluating generated questions.

Chapter 6

Comparing Methods for Automatically Evaluating Generated Reading Comprehension Questions

Question generation systems are typically evaluated using a set of manually created reference questions. However, each generated question is often evaluated individually, without regard for other questions generated for the same context [44]. This approach can incentivize low diversity questions and does not align with real-world settings where systems must generate many questions per context. In this chapter, we explore multiple ways of evaluating generated questions and compare how well they capture diversity and align with multiple aspects of human judgement.

6.1 Automatic Evaluation Methods

In this section, we describe and compare five methods that can be used to evaluate generated reading comprehension questions. These methods include *Best Reference Evaluation*, *Cartesian Product Evaluation*, *Multi-METEOR Evaluation*, *Fréchet BERT Distance*, and *MS-Jaccard4 Evaluation*. We later evaluate each of the methods described here in Section 6.2.

6.1.1 Best Reference Evaluation

Best Reference Evaluation follows the methods of Ghanem *et al.* [12], recording the score of the best reference for a given candidate question. We then compute the mean of the similarity metric score for each candidate question matching the best reference, ignoring other candidate questions that might map to the same reference. For this method, a reference question can appear in multiple pairs, potentially hiding a lack of diversity. See Figure 6.1.

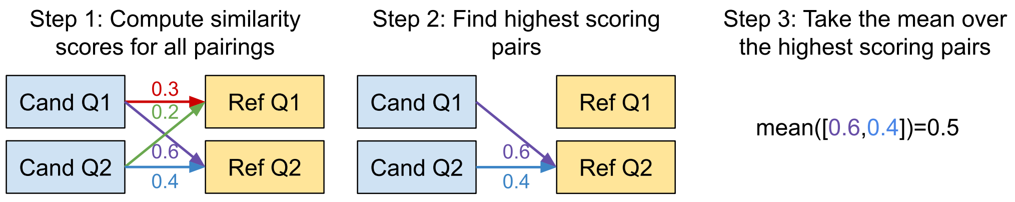

Step 1: Compute similarity scores for all pairings
Cand Q1
Cand Q2
Ref Q1
Ref Q2
0.3
0.2
0.6
0.4
Step 2: Find highest scoring pairs
Cand Q1
Cand Q2
Ref Q1
Ref Q2
0.6
0.4
Step 3: Take the mean over the highest scoring pairs
mean([0.6,0.4])=0.5

Figure 6.1: Three main steps of Best Reference Evaluation. In the first step, the similarity metric is computed for all possible pairings of candidate and reference questions. In the second step, the highest scoring pairs of candidate and reference questions are selected, with the possibility that multiple candidates can map to the same reference. In the third and final step, the Best Reference score is calculated by taking the mean over all the scores from the highest scoring pairs.

6.1.2 Cartesian Product Evaluation

Cartesian Product Evaluation is a generalization of Self-BLEU [48] to allow for the comparison of two distinct sets of text. Unlike Best Reference, the Cartesian Product Evaluation method does not try to match pairs of questions from each set. Instead, it simply takes the mean similarity metric score over all pairs in the Cartesian product of the reference questions and candidate questions for a given context. See Figure 6.2.

6.1.3 Multi-METEOR Evaluation

Multi-METEOR [11] is a recent method for evaluating multiple generated questions in a given context by considering the full generated candidate set and reference question set.

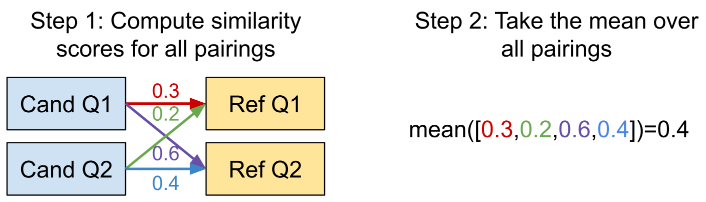

Step 1: Compute similarity scores for all pairings
Step 2: Take the mean over all pairings
Cand Q1
Cand Q2
Ref Q1
Ref Q2
0.3
0.2
0.6
0.4
mean([0.3,0.2,0.6,0.4])=0.4

Figure 6.2: Two main steps of Cartesian Product Evaluation. In the first step, the similarity metric is computed for all possible pairings of candidate and reference questions. In the second and final step, the Cartesian Product score is calculated by taking the mean over all the scores from the previous step.

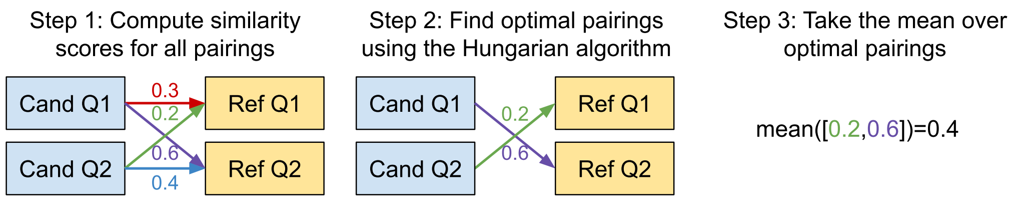

Step 1: Compute similarity scores for all pairings
Step 2: Find optimal pairings using the Hungarian algorithm
Step 3: Take the mean over optimal pairings
Cand Q1
Cand Q2
Ref Q1
Ref Q2
0.3
0.2
0.6
0.4
Cand Q1
Cand Q2
Ref Q1
Ref Q2
0.2
0.6
mean([0.2,0.6])=0.4

Figure 6.3: Three main steps of Multi-METEOR Evaluation. In the first step, the similarity metric is computed for all possible pairings of candidate and reference questions. In the second step, the optimal one-to-one pairings of candidate and reference questions are found using the Hungarian algorithm [19, 25]. In the third and final step, the Multi-METEOR score is calculated by taking the mean over all the scores from the optimal pairings.

To compute Multi-METEOR, we first calculate a similarity metric, METEOR [5], for each pair in the Cartesian product of the reference questions and the candidate questions. Second, using the Hungarian algorithm [19, 25], we find the optimal one-to-one pairings of candidate and reference questions in polynomial time. Finally, we compute the Multi-METEOR score for the set of candidate questions based on the mean similarity score over all optimal pairings. See Figure 6.3.

6.1.4 Fréchet BERT Distance (FBD)

The *Fréchet BERT Distance* (FBD) [2] determines the distance between aggregated candidate and reference set embeddings. FBD is able to capture similarity through its comparison of the mean embeddings and captures diversity by comparing the covariance matrices.

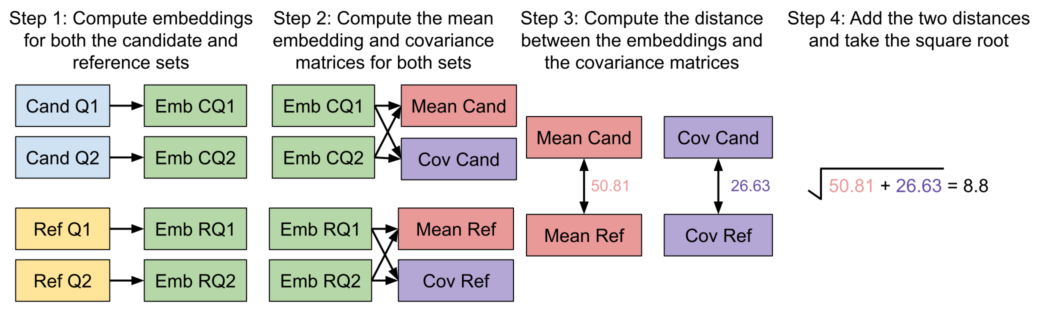

Step 1: Compute embeddings for both the candidate and reference sets
Step 2: Compute the mean embedding and covariance matrices for both sets
Step 3: Compute the distance between the embeddings and the covariance matrices
Step 4: Add the two distances and take the square root
Cand Q1
Cand Q2
Ref Q1
Ref Q2
Emb CQ1
Emb CQ2
Emb RQ1
Emb RQ2
Mean Cand
Cov Cand
Mean Ref
Cov Ref
50.81
26.63
√50.81 + 26.63 = 8.8

Figure 6.4: Four main steps of Fréchet BERT Distance (FBD) Evaluation. In the first step, the embedding vectors are computed for each candidate and reference question. In the second, the mean embedding vector and covariance matrixes are computed for both sets. In the third step, the Euclidean distance is computed between the candidate and reference mean vectors, and a distance is computed between the candidate and reference covariance matrixes. For the final step, the two distances are added together and the square root is taken to obtain the final FBD score.

As FBD is a distance measure rather than a similarity measure, we use the negative FBD for evaluation to keep it comparable to the other automatic evaluation methods. See Figure 6.4.

6.1.5 MS-Jaccard4 Evaluation

The final method we compare is *MS-Jaccard4* [2], which was created as a similarity score that could capture diversity. Similar to FBD, MS-Jaccard4 considers the sets as a whole instead of considering individual candidate-reference pairs. However, instead of using embeddings, MS-Jaccard4 compares the frequencies of sub-sequences of tokens between the candidate and reference questions. These sub-sequences are commonly referred to as *n-grams*. See Figure 6.5.

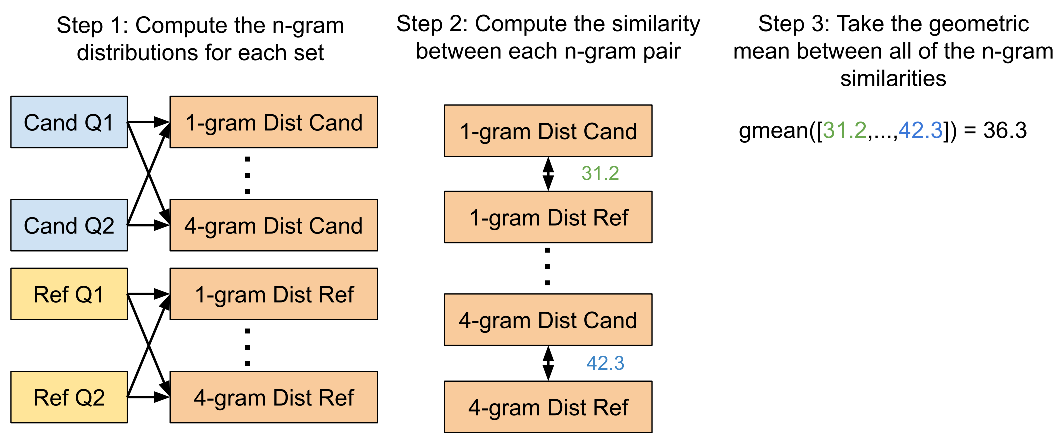

Step 1: Compute the n-gram distributions for each set
Step 2: Compute the similarity between each n-gram pair
Step 3: Take the geometric mean between all of the n-gram similarities
Cand Q1
Cand Q2
Ref Q1
Ref Q2
1-gram Dist Cand
4-gram Dist Cand
1-gram Dist Ref
4-gram Dist Ref
1-gram Dist Cand
31.2
1-gram Dist Ref
4-gram Dist Cand
42.3
4-gram Dist Ref
gmean([31.2,...,42.3]) = 36.3

Figure 6.5: Three main steps of MS-Jaccard4 Evaluation. In the first step, the n-gram counts from 1 to 4 are collected from both the candidate questions and the reference questions to obtain 4 n-gram distributions for each set. In the second step, the similarities are computed between each pair of n-gram distributions. Finally, the geometric mean is taken between all 4 similarities to obtain the final MS-Jaccard4 score.

6.2 Automatic Evaluation Experiments

For this section, we run two experiments to compare different methods of automatic evaluation. First, we evaluate how well each method captures diversity. Second, we determine how well each method aligns with human judgement.

6.2.1 Experiment 1: Capturing Diversity in Automatic Evaluation Methods

To evaluate how well diversity is captured by each automatic evaluation method, we first used each of the question generation methods described in Section 3.2 to generate questions for each of the contexts in the QuAIL test set. We then computed a score for each context generation method pair using each automatic evaluation method described in Section 6.1. To measure the diversity of a question set, we first calculated the self-similarity score described in Section 3.5.3. A higher self-similarity score means the question set is less diverse. We then correlated the automatic evaluation method scores with the self-similarity score for each question set, using both Pearson and Spearman correlations. A lower correlation indicates that the automatic evaluation method assigns low

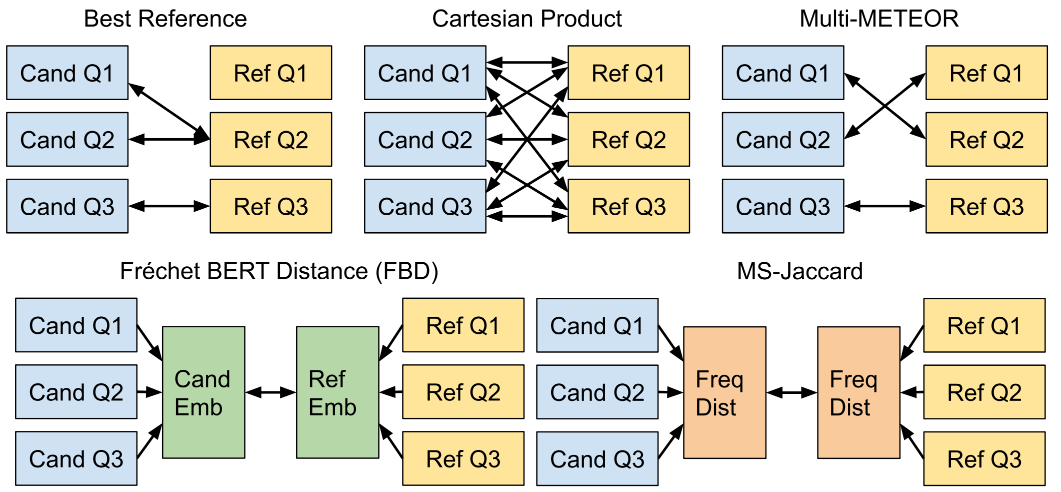


Figure 6.6: Automatic Evaluation Methods. Candidate questions are model-generated; reference questions are those associated with the same context in a dataset. In Best Reference, we match each candidate question to a reference question but allow multiple candidates to match to the same reference (N total matches). In Cartesian Product, we match every candidate question with every reference question (N^2 total matches). In Multi-METEOR, we find the best 1:1 match between candidate and reference questions (N total matches). For each of these three methods, we report the average score over all matches. Only the Multi-METEOR score computes the best 1-1 match between candidate and reference questions. Instead of comparing individual question pairs, Fréchet BERT Distance (FBD) compared aggregated embeddings between the candidate and reference question sets. Similarly, MS-Jaccard compares the n-gram frequency distributions between the two sets.

scores to low diversity question sets (i.e., those sets with high self-similarity). Table 6.1 shows the results of this experiment.

6.2.2 Experiment 2: Alignment with Human Judgement of Automatic Evaluation Methods

For our second experiment, we perform an analysis of how well each automatic evaluation method correlated with human judgement. For this analysis, we used the human annotations obtained in Section 3.6.3. We assign Answerability values of 1 and 0 for *answerable* and *not answerable* repetitively and assign Context Specificity values of 1 and 0 for *context-specific* and *not context-specific*. Finally, we computed Spearman's rank correlation between the human evaluation aspect scores and the automated evaluation method

Method	**Pearson**	**Spearman**
Best Reference	0.3008	0.3543
Cartesian	0.3846	0.4940
Multi-METEOR	-0.0832	0.0218
FBD	<u>-0.0752</u>	<u>-0.0887</u>
MS-Jaccard4	**-0.3950**	**-0.3390**

Table 6.1: Correlation between automatic evaluation matching methods and self-similarity. Lower values indicate a preference for a more diverse question set. MS-Jaccard4 has a lower correlation than the other methods, indicating that it better captures the diversity of a question set. The range of values is from -1 to 1. Lowest is shown in **bold**, and second lowest is <u>underlined</u>.

scores. We used Spearman's rank correlation instead of Pearson correlation because it does not assume the relationship between variables is linear [7]. Figure 6.7 shows the results of this experiment.

6.3 Discussion

The results show that Cartesian Product evaluation is the most correlated with human judgement for *Context Specificity* and the only evaluation method to have a positive correlation with all three human evaluation aspect. This result suggest that Cartesian Product may be the best metrics for cases where questions that reference specific information from the context are preferred. However, Cartesian Product does not capture diversity as well as MS-Jaccard4 and FBD. MS-Jaccard4 correlates the least with self-similarity and therefore captures diversity best, but does not correlate very well with any of the human evaluation aspects. FBD on the other hand has the highest correlation with *Fluency* and has a high correlation with *Answerability.* Therefore, when diversity of the questions generated is important then FBD is likely the best method to evaluate generated questions. Finally, Best Reference evaluation correlates most closely with human evaluation of *Answerability*, so it may be most useful in scenarios where questions are generated and selected in a completely automated manner. These results also emphasize the need for multiple automated metrics and human evaluation when evaluating question generation methods.

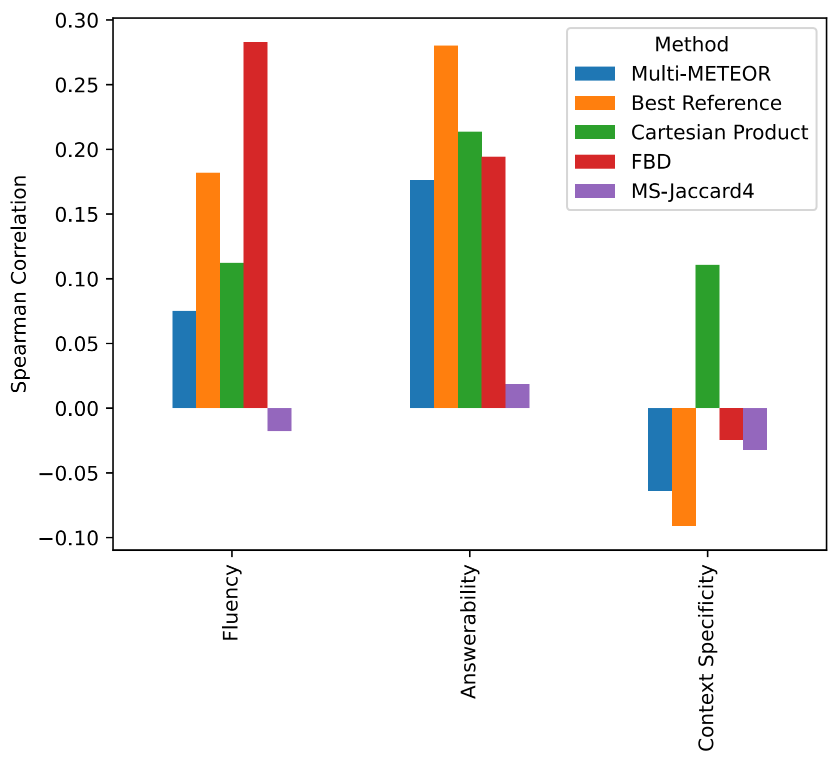


Figure 6.7: Correlation between each of the automatic evaluation methods and mean human annotator scores for *Fluency*, *Context Specificity*, and *Answerability*, as well as the mean over all three. Higher values indicate closer alignment with human judgement. FBD has the highest correlation with *Fluency*. Cartesian Product has the highest correlation with *Context Specificity*. Best Reference has the highest correlation with *Answerability*. The range of values is from -1 to 1.

6.4 Chapter Summary

In this chapter, we examined five methods for evaluating generated reading comprehension questions: Best Reference, Cartesian Product, Multi-METEOR, FBD, and MS-Jaccard4. We showed that: MS-Jaccard4 best captures the diversity of a set of questions, Cartesian Product best aligns with human judgement of Context Specificity, Best Reference best aligns with human judgement of Answerability, and Fréchet BERT Distance (FBD) most closely aligns human judgement of Fluency. These results emphasis the need to use multiple diverse evaluation metrics when evaluating questions. In the next chapter, we summarize the findings of this thesis, discuss some limitations, and describe some directions for future work.

Chapter 7

Conclusion, Limitations & Future Work

For this final chapter, we discuss the conclusion and limitations of our work. Additionally, we present potential directions for future work.

7.1 Conclusion

In this thesis, we explored different methods for generating and evaluating skill targeted reading comprehension questions.

We proposed a new question generation method, SoftSkillQG that uses soft-prompts to control the types of questions that it generates so no manual effort is required to create prompts to target new skills. We show that SoftSkillQG outperforms existing methods across four out of five automatic evaluation metrics on the SBRCS dataset and on human evaluation of Context-Specificity. However, T5 WTA, a method that uses manually created prompts, achieves higher perplexity and automated evaluation scores on the QuAIL dataset. We found the performance of T5 WTA surprising compared to more flexible SoftSkillQG. As a result, we performed additional analysis to determine the source of this unexpected difference in performance. We showed that a lack of training data and sub-optimal initializations both likely contribute to this difference. However, when SoftSkillQG was initialized using the same manually created prompts as T5 WTA, it was able to achieve nearly the same perplexity on the QuAIL dataset.

Finally, we compared a number of methods for evaluating sets of generated questions. We showed that: MS-Jaccard4 best captures the diversity of a set of questions, Cartesian Product best aligns with human judgement of Context Specificity, Best Reference best aligns with human judgement of Answerability, and Fréchet BERT Distance (FBD) most closely aligns human judgement of Fluency. Based on these findings, we emphasised need to use multiple diverse automated evaluation methods when evaluating tasks like question generation.

7.2 Limitations

There are important limitations to our work. Firstly, our soft-prompt method requires a significant amount of computational resources. Without parallelization, it takes approximately 2 weeks using an Nvidia V100 GPU to perform hyperparameter tuning. However, once hyperparameters are found, each model can be trained in less than 48 hours. Secondly, because we use the T5 language model, which limits the input to 512 tokens, our methods will only work for short contexts. Although there has been some work exploring methods to support longer contexts [1, 46], this direction is beyond the scope of this thesis. Thirdly, we only evaluated our methods on English datasets, so it is possible that our results will not generalize to other datasets or other languages. Fourthly, it is possible to have questions target multiple skills. However, for this thesis, we focus on simpler questions that only target a single skill. Finally, this work focuses only on question generation and not question answering. We believe that question generation can be useful on its own. However, it may be more helpful in applications like education software to also have an answer available. In this case, there are already robust question answering systems exceeding human performance [45] on the SQuAD2.0 [32] benchmark that could potentially be combined with our work to create tools to aid teachers in the classroom with developing the reading skills of their students.

7.3 Future Work

There are many possible extensions to this work. One potential continuation of this work we would like to explore is control over multiple aspects of the question generated. For example, we could use techniques like SoftSkillQG to control both the skill that a question targets and the question difficulty simultaneously. Another potential area that we would like to explore is multilingual question generation. There is a significant amount of reading comprehension questions and other literacy resources in English. However, there are many languages that do not have the same level of resources as English. It would be greatly beneficial to new readers of these languages to have access to have a similar breadth of reading comprehension questions. Lastly, we would like to test our proposed methods in a real-world setting to see if they provide a similar level of benefit to readers as manually created reading comprehension questions.

We hope that this thesis helps deepen understanding of skill targeted reading comprehension and generation tasks as a whole. Additionally, we are optimistic that insights into the sensitivity of soft-prompt methods to dataset size and initialization will help improve understanding in the field. Finally, we hope that this work will help improve access to reading comprehension resources for children learning to read, and ultimately help them achieve their reading goals.

References

[1] S. Alhashedi, N. M. Suaib, and A. Bakri, *Arabic automatic question generation using transformer model*, EasyChair Preprint no. 8588, 2022.

[2] D. Alihosseini, E. Montahaei, and M. Soleymani Baghshah, "Jointly measuring diversity and quality in text generation models," in *Proceedings of the Workshop on Methods for Optimizing and Evaluating Neural Language Generation*, Minneapolis, Minnesota: Association for Computational Linguistics, Jun. 2019, pp. 90–98. DOI: 10.18653/v1/W19-2311. [Online]. Available: https://aclanthology.org/W19-2311.

[3] A. Asai, M. Salehi, M. Peters, and H. Hajishirzi, "ATTEMPT: Parameter-efficient multi-task tuning via attentional mixtures of soft prompts," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, Dec. 2022, pp. 6655–6672. DOI: 10.18653/v1/2022.emnlp-main.446. [Online]. Available: https://aclanthology.org/2022.emnlp-main.446.

[4] P. Azevedo, B. Leite, H. L. Cardoso, D. C. Silva, and L. P. Reis, "Exploring nlp and information extraction to jointly address question generation and answering," *Artificial Intelligence Applications and Innovations*, vol. 584, pp. 396–407, 2020.

[5] S. Banerjee and A. Lavie, "METEOR: An automatic metric for MT evaluation with improved correlation with human judgments," in *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, Ann Arbor, Michigan: Association for Computational Linguistics, Jun. 2005, pp. 65–72. [Online]. Available: https://aclanthology.org/W05-0909.

[6] S. Bird, E. Klein, and E. Loper, *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.", 2009.

[7] S.-D. Bolboaca and L. Jäntschi, "Pearson versus spearman, kendall's tau correlation analysis on structure-activity relationships of biologic active compounds," *Leonardo Journal of Sciences*, vol. 5, no. 9, pp. 179–200, 2006.

[8] T. Brown *et al.*, "Language models are few-shot learners," in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33, Curran Associates, Inc., 2020, pp. 1877–1901. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[9] S. Cao and L. Wang, "Controllable open-ended question generation with a new question type ontology," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Online: Association for Computational Linguistics, Aug. 2021, pp. 6424–6439. DOI: 10.18653/v1/2021.acl-long.502. [Online]. Available: https://aclanthology.org/2021.acl-long.502.

[10] W. Chen, G. Aist, and J. Mostow, "Generating questions automatically from informational text," in *Proceedings of AIED 2009 Workshop on Question Generation*, 2009, pp. 17–24. [Online]. Available: https://www.cs.cmu.edu/~listen/pdfs/QG2009-Wei-informational-text-questions-final.pdf.

[11] J. R. Chowdhury, D. Mahata, and C. Caragea, "On the evaluation of answer-agnostic paragraph-level multi-question generation," *ArXiv*, vol. abs/2203.04464, 2022.

[12] B. Ghanem, L. Lutz Coleman, J. Rivard Dexter, S. von der Ohe, and A. Fyshe, "Question generation for reading comprehension assessment by modeling how and what to ask," in *Findings of the Association for Computational Linguistics: ACL 2022*, Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 2131–2146. DOI: 10.18653/v1/2022.findings-acl.168. [Online]. Available: https://aclanthology.org/2022.findings-acl.168.

[13] V. Harrison and M. Walker, "Neural generation of diverse questions using answer focus, contextual and linguistic features," in *Proceedings of the 11th International Conference on Natural Language Generation*, Tilburg University, The Netherlands: Association for Computational Linguistics, Nov. 2018, pp. 296–306. DOI: 10.18653/v1/W18-6536. [Online]. Available: https://aclanthology.org/W18-6536.

[14] V. Harrison and M. Walker, "Neural generation of diverse questions using answer focus, contextual and linguistic features," in *Proceedings of the 11th International Conference on Natural Language Generation*, Tilburg University, The Netherlands: Association for Computational Linguistics, Nov. 2018, pp. 296–306. DOI: 10.18653/v1/W18-6536. [Online]. Available: https://aclanthology.org/W18-6536.

[15] A. Holtzman, J. Buys, L. Du, M. Forbes, and Y. Choi, "The curious case of neural text degeneration," in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, OpenReview.net, 2020. [Online]. Available: `https://openreview.net/forum?id=rygGQyrFvH`.

[16] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991. DOI: `10.1162/neco.1991.3.1.79`.

[17] T. Ji, C. Lyu, G. J. F. Jones, L. Zhou, and Y. Graham, "Qascore—an unsupervised unreferenced metric for the question generation evaluation," *Entropy*, vol. 24, 2022.

[18] D. Jurafsky and J. H. Martin, *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, Third Edition Draft. Jan. 2023.

[19] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics (NRL)*, vol. 52, 1955.

[20] B. Lester, R. Al-Rfou, and N. Constant, "The power of scale for parameter-efficient prompt tuning," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, Nov. 2021, pp. 3045–3059. DOI: `10.18653/v1/2021.emnlp-main.243`. [Online]. Available: `https://aclanthology.org/2021.emnlp-main.243`.

[21] X. L. Li and P. Liang, "Prefix-tuning: Optimizing continuous prompts for generation," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Online: Association for Computational Linguistics, Aug. 2021, pp. 4582–4597. DOI: `10.18653/v1/2021.acl-long.353`. [Online]. Available: `https://aclanthology.org/2021.acl-long.353`.

[22] C.-Y. Lin, "ROUGE: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*, Barcelona, Spain: Association for Computational Linguistics, Jul. 2004, pp. 74–81. [Online]. Available: `https://www.aclweb.org/anthology/W04-1013`.

[23] Y. Liu *et al.*, "Roberta: A robustly optimized BERT pretraining approach," *CoRR*, vol. abs/1907.11692, 2019. arXiv: `1907.11692`. [Online]. Available: `http://arxiv.org/abs/1907.11692`.

[24] J. Mostow and W. Chen, "Generating instruction automatically for the reading strategy of self-questioning," in *Proceedings of the 14th International Conference on Artificial Intelligence in Education*, vol. 200, Jan. 2009, pp. 465–472. DOI: `10.3233/978-1-60750-028-5-465`.

[25] J. R. Munkres, “Algorithms for the assignment and transportation problems,” *Journal of The Society for Industrial and Applied Mathematics*, vol. 10, pp. 196–210, 1957.

[26] L. Murakhovs’ka, C.-S. Wu, T. Niu, W. Liu, and C. Xiong, *Mixqg: Neural question generation with mixed answer types*, 2021. arXiv: `2110.08175 [cs.CL]`.

[27] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “BLEU: A method for automatic evaluation of machine translation,” en, in *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL ’02*, Philadelphia, Pennsylvania: Association for Computational Linguistics, 2001, p. 311. DOI: `10.3115/1073083.1073135`. [Online]. Available: `http://portal.acm.org/citation.cfm?doid=1073083.1073135` (visited on 06/23/2023).

[28] G.-M. Park, S.-E. Hong, and S.-B. Park, “Post-training with interrogative sentences for enhancing BART-based Korean question generator,” in *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, Online only: Association for Computational Linguistics, Nov. 2022, pp. 202–209. [Online]. Available: `https://aclanthology.org/2022.aacl-short.26`.

[29] G. Qin and J. Eisner, “Learning how to ask: Querying LMs with mixtures of soft prompts,” in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online: Association for Computational Linguistics, Jun. 2021, pp. 5203–5212. DOI: `10.18653/v1/2021.naacl-main.410`. [Online]. Available: `https://aclanthology.org/2021.naacl-main.410`.

[30] C. Raffel *et al.*, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *Journal of Machine Learning Research*, vol. 21, no. 140, pp. 1–67, 2020. [Online]. Available: `http://jmlr.org/papers/v21/20-074.html`.

[31] T. C. Rajapakse, *Simple transformers*, 2019. [Online]. Available: `https://github.com/ThilinaRajapakse/simpletransformers`.

[32] P. Rajpurkar, R. Jia, and P. Liang, “Know what you don’t know: Unanswerable questions for squad,” *CoRR*, vol. abs/1806.03822, 2018. arXiv: `1806.03822`. [Online]. Available: `http://arxiv.org/abs/1806.03822`.

[33] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, “SQuAD: 100,000+ questions for machine comprehension of text,” in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, J. Su, K. Duh, and X. Carreras, Eds., Austin, Texas: Association for Compu-

tational Linguistics, Nov. 2016, pp. 2383–2392. DOI: 10.18653/v1/D16-1264. [Online]. Available: https://aclanthology.org/D16-1264.

[34] A. Rogers, O. Kovaleva, M. Downey, and A. Rumshisky, "Getting closer to AI complete question answering: A set of prerequisite real tasks," in *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, AAAI Press, 2020, pp. 8722–8731. [Online]. Available: https://aaai.org/ojs/index.php/AAAI/article/view/6398.

[35] M. S. Schlichtkrull and W. Cheng, "Evaluating for diversity in question generation over text," *CoRR*, vol. abs/2008.07291, 2020. arXiv: 2008.07291. [Online]. Available: https://arxiv.org/abs/2008.07291.

[36] T. Sellam, D. Das, and A. Parikh, "BLEURT: Learning robust metrics for text generation," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 7881–7892. DOI: 10.18653/v1/2020.acl-main.704. [Online]. Available: https://aclanthology.org/2020.acl-main.704.

[37] J. Stockard and K. Engelmann, "The development of early academic success: The impact of direct instruction's reading mastery.," *Journal of Behavior Assessment and Intervention in Children*, vol. 1, no. 1, p. 2, 2010.

[38] A. Vaswani *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, I. Guyon *et al.*, Eds., vol. 30, Curran Associates, Inc., 2017. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

[39] X. Wang, B. Liu, S. Tang, and L. Wu, "SkillQG: Learning to generate question for reading comprehension assessment," in *Findings of the Association for Computational Linguistics: ACL 2023*, Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 13 833–13 850. DOI: 10.18653/v1/2023.findings-acl.870. [Online]. Available: https://aclanthology.org/2023.findings-acl.870.

[40] X. Wang, B. Liu, S. Tang, and L. Wu, "SkillQG: Learning to generate question for reading comprehension assessment," in *Findings of the Association for Computational Linguistics: ACL 2023*, Toronto, Canada: Association for Computational Linguistics, Jul. 2023, pp. 13 833–13 850. DOI: 10.18653/v1/2023.findings-acl.870. [Online]. Available: https://aclanthology.org/2023.findings-acl.870.

[41] T. Wolf *et al.*, "Transformers: State-of-the-art natural language processing," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. DOI: 10.18653/v1/2020.emnlp-demos.6. [Online]. Available: https://aclanthology.org/2020.emnlp-demos.6.

[42] J. Xu, Y. Sun, J. Gan, M. Zhou, and D. Wu, "Leveraging structured information from a passage to generate questions," *Tsinghua Science and Technology*, vol. 28, no. 3, pp. 464–474, 2023. DOI: 10.26599/TST.2022.9010034.

[43] Y. Xu *et al.*, "Fantastic questions and where to find them: FairytaleQA – an authentic dataset for narrative comprehension," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, S. Muresan, P. Nakov, and A. Villavicencio, Eds., Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 447–460. DOI: 10.18653/v1/2022.acl-long.34. [Online]. Available: https://aclanthology.org/2022.acl-long.34.

[44] R. Zhang, J. Guo, L. Chen, Y. Fan, and X. Cheng, "A review on question generation from natural language text," *ACM Trans. Inf. Syst.*, vol. 40, no. 1, Sep. 2021, ISSN: 1046-8188. DOI: 10.1145/3468889. [Online]. Available: https://doi.org/10.1145/3468889.

[45] Z. Zhang, J. Yang, and H. Zhao, "Retrospective reader for machine reading comprehension," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 16, pp. 14 506–14 514, May 2021. DOI: 10.1609/aaai.v35i16.17705. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/17705.

[46] Z. Zhao, Y. Hou, D. Wang, M. Yu, C. Liu, and X. Ma, "Educational question generation of children storybooks via question type distribution learning and event-centric summarization," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Dublin, Ireland: Association for Computational Linguistics, May 2022, pp. 5073–5085. DOI: 10.18653/v1/2022.acl-long.348. [Online]. Available: https://aclanthology.org/2022.acl-long.348.

[47] W. Zhou, M. Zhang, and Y. Wu, "Question-type driven question generation," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, Hong Kong, China: Association for Computational Linguistics, Nov. 2019, pp. 6032–6037. DOI: 10.18653/v1/D19-1622. [Online]. Available: https://aclanthology.org/D19-1622.

[48] Y. Zhu *et al.*, "Texygen: A benchmarking platform for text generation models," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, ser. SIGIR '18, Ann Arbor, MI, USA: Association for Computing Machinery, 2018, pp. 1097–1100, ISBN: 9781450356572. DOI: `10.1145/3209978.3210080`. [Online]. Available: `https://doi.org/10.1145/3209978.3210080`.

Appendix A

Implementation Details

A.1 Optimal Hyperparameters

We show the optimal hyperparameter for the QuAIL dataset in Table A.1 and the optimal hyperparameter for the SBRCS dataset in Table A.2. We used a learning rate of *1e-05* for the RoBERTa classifier and used an AdamW optimizer for all models.

Hyperparameters	Model	Seed
p = 0.5, Learning rate = 1e-5	SoftSkillQG	1
p = 0.6, Learning rate = 1e-5	SoftSkillQG	2
p = 0.3, Learning rate = 5e-5	SoftSkillQG	3
p = 0.5, Learning rate = 5e-6	T5 WTA	1
p = 0.5, Learning rate = 5e-6	T5 WTA	2
p = 0.6, Learning rate = 5e-6	T5 WTA	3
p = 0.5, Learning rate = 5e-6	IMoE	1
p = 0.5, Learning rate = 1e-4	IMoE	2
p = 0.4, Learning rate = 1e-4	IMoE	3
p = 0.7, Learning rate = 5e-6	No Control	1
p = 0.7, Learning rate = 5e-6	No Control	2
p = 0.7, Learning rate = 5e-6	No Control	3
p = 0.7, Learning rate = 1e-4	Shared Soft-prompt	1
p = 0.6, Learning rate = 1e-4	Shared Soft-prompt	2
p = 0.7, Learning rate = 1e-4	Shared Soft-prompt	3
p = 0.5, Learning rate = 5e-1	Soft-prompting	1
p = 0.4, Learning rate = 5e-1	Soft-prompting	2
p = 0.3, Learning rate = 2.5e-1	Soft-prompting	3

Table A.1: Optimal hyperparameters on the QuAIL development set.

Hyperparameters	Model	Seed
p = 0.3, Learning rate = 1e-5	SoftSkillQG	1
p = 0.1, Learning rate = 5e-5	SoftSkillQG	2
p = 0.4, Learning rate = 1e-5	SoftSkillQG	3
p = 0.6, Learning rate = 5e-4	T5 WTA	1
p = 0.1, Learning rate = 1e-4	T5 WTA	2
p = 0.5, Learning rate = 5e-5	T5 WTA	3
p = 0.6, Learning rate = 5e-4	IMoE	1
p = 0.3, Learning rate = 5e-4	IMoE	2
p = 0.2, Learning rate = 5e-4	IMoE	3
p = 0.2, Learning rate = 1e-5	No Control	1
p = 0.3, Learning rate = 1e-5	No Control	2
p = 0.3, Learning rate = 5e-6	No Control	3
p = 0.3, Learning rate = 1e-4	Shared Soft-prompt	1
p = 0.2, Learning rate = 5e-4	Shared Soft-prompt	2
p = 0.2, Learning rate = 1e-4	Shared Soft-prompt	3
p = 0.3, Learning rate = 2.5e-1	Soft-prompting	1
p = 0.3, Learning rate = 2.5e-1	Soft-prompting	2
p = 0.7, Learning rate = 5e-1	Soft-prompting	3

Table A.2: Optimal hyperparameters on the SBRCS development set.

A.2 Seeds

For each experiment utilizing seeds, the same three seeds are used: 1, 2, and 3. See the code at `https://github.com/sazzy4o/thesis-code` to see how the seeds are implemented.

A.3 Annotator Agreement

We commute the Fleiss' kappa value between annotators for *Fluency*, *Answerable*, and *Context Specific* to determine the agreement between annotators. We present these values in Table A.3

Fluency	**Answerability**	**Context Specificity**
0.1922	0.2609	0.3773

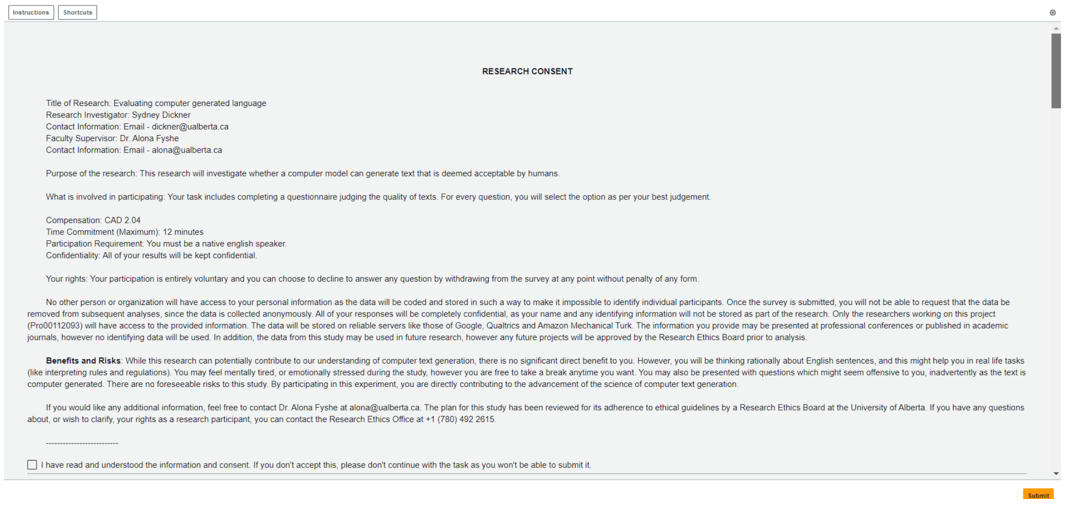
Table A.3: Fleiss' Kappa for Human Annotations. Values indicate fair agreement between annotators for *Answerability* and *Context Specificity*. *Fluency* has a lower kappa value. However, Fleiss' kappa does not consider the rank order of annotations, which may make *Fluency* appear to have less agreement. Higher values indicate greater agreement.

A.4 Human Evaluation Instructions

We have included some screenshots that include the instructions in Figure A.1. Note that we group multiple annotations for the same context into the one Human Intelligence Task (HIT).

We also include the following full-text version of the instructions shown to annotators:

> Title of Research: Evaluating computer generated language
>
> Research Investigator: Sydney Dickner
>
> Contact Information: Email - dickner@ualberta.ca
>
> Faculty Supervisor: Dr. Alona Fyshe
>
> Contact Information: Email - alona@ualberta.ca
>
> Purpose of the research: This research will investigate whether a computer model can generate text that is deemed acceptable by humans.
>
> What is involved in participating: Your task includes completing a questionnaire judging the quality of texts. For every question, you will select the option as per your best judgement.
>
> Compensation: CAD 2.04
>
> Time Commitment (Maximum): 12 minutes
>
> Participation Requirement: You must be a native english speaker.
>
> Confidentiality: All of your results will be kept confidential.
>
> Your rights: Your participation is entirely voluntary and you can choose to decline to answer any question by withdrawing from the survey at any point without penalty of any form.
>
> No other person or organization will have access to your personal information as the data will be coded and stored in such a way to make it impossible to identify individual participants. Once the survey is submitted, you will not be able to request that the data



RESEARCH CONSENT

Title of Research: Evaluating computer generated language
Research Investigator: Sydney Dickner
Contact Information: Email - dickner@ualberta.ca
Faculty Supervisor: Dr. Alona Fyshe
Contact Information: Email - alona@ualberta.ca

Purpose of the research: This research will investigate whether a computer model can generate text that is deemed acceptable by humans.

What is involved in participating: Your task includes completing a questionnaire judging the quality of texts. For every question, you will select the option as per your best judgement.

Compensation: CAD 2.04
Time Commitment (Maximum): 12 minutes
Participation Requirement: You must be a native english speaker.
Confidentiality: All of your results will be kept confidential.

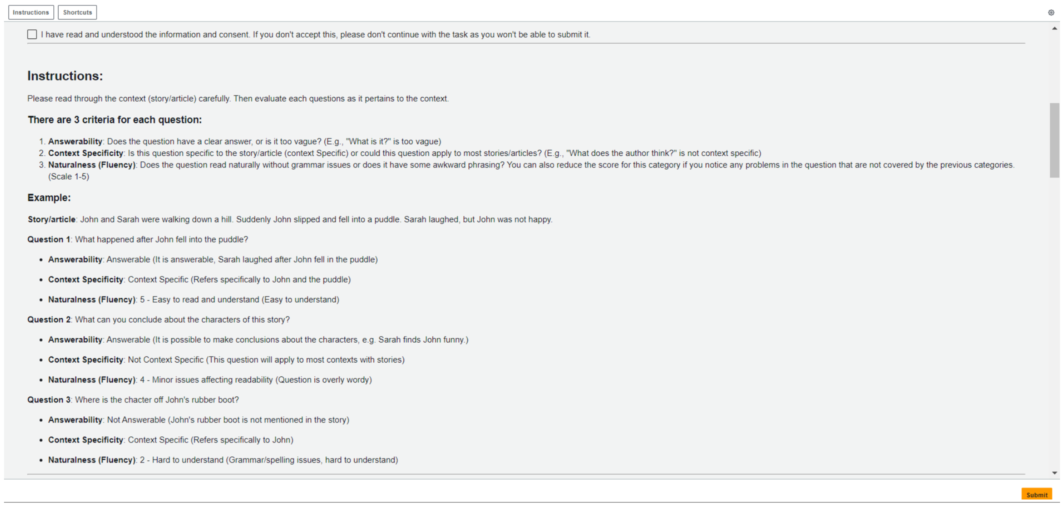
Your rights: Your participation is entirely voluntary and you can choose to decline to answer any question by withdrawing from the survey at any point without penalty of any form.

No other person or organization will have access to your personal information as the data will be coded and stored in such a way to make it impossible to identify individual participants. Once the survey is submitted, you will not be able to request that the data be removed from subsequent analyses, since the data is collected anonymously. All of your responses will be completely confidential, as your name and any identifying information will not be stored as part of the research. Only the researchers working on this project (Pro00112093) will have access to the provided information. The data will be stored on reliable servers like those of Google, Qualtrics and Amazon Mechanical Turk. The information you provide may be presented at professional conferences or published in academic journals, however no identifying data will be used. In addition, the data from this study may be used in future research, however any future projects will be approved by the Research Ethics Board prior to analysis.

Benefits and Risks: While this research can potentially contribute to our understanding of computer text generation, there is no significant direct benefit to you. However, you will be thinking rationally about English sentences, and this might help you in real life tasks (like interpreting rules and regulations). You may feel mentally tired, or emotionally stressed during the study, however you are free to take a break anytime you want. You may also be presented with questions which might seem offensive to you, inadvertently as the text is computer generated. There are no foreseeable risks to this study. By participating in this experiment, you are directly contributing to the advancement of the science of computer text generation.

If you would like any additional information, feel free to contact Dr. Alona Fyshe at alona@ualberta.ca. The plan for this study has been reviewed for its adherence to ethical guidelines by a Research Ethics Board at the University of Alberta. If you have any questions about, or wish to clarify, your rights as a research participant, you can contact the Research Ethics Office at +1 (780) 492 2615.

☐ I have read and understood the information and consent. If you don't accept this, please don't continue with the task as you won't be able to submit it.



☐ I have read and understood the information and consent. If you don't accept this, please don't continue with the task as you won't be able to submit it.

Instructions:

Please read through the context (story/article) carefully. Then evaluate each questions as it pertains to the context.

There are 3 criteria for each question:

1. **Answerability**: Does the question have a clear answer, or is it too vague? (E.g., "What is it?" is too vague)
2. **Context Specificity**: Is this question specific to the story/article (context Specific) or could this question apply to most stories/articles? (E.g., "What does the author think?" is not context specific)
3. **Naturalness (Fluency)**: Does the question read naturally without grammar issues or does it have some awkward phrasing? You can also reduce the score for this category if you notice any problems in the question that are not covered by the previous categories. (Scale 1-5)

Example:

Story/article: John and Sarah were walking down a hill. Suddenly John slipped and fell into a puddle. Sarah laughed, but John was not happy.

Question 1: What happened after John fell into the puddle?

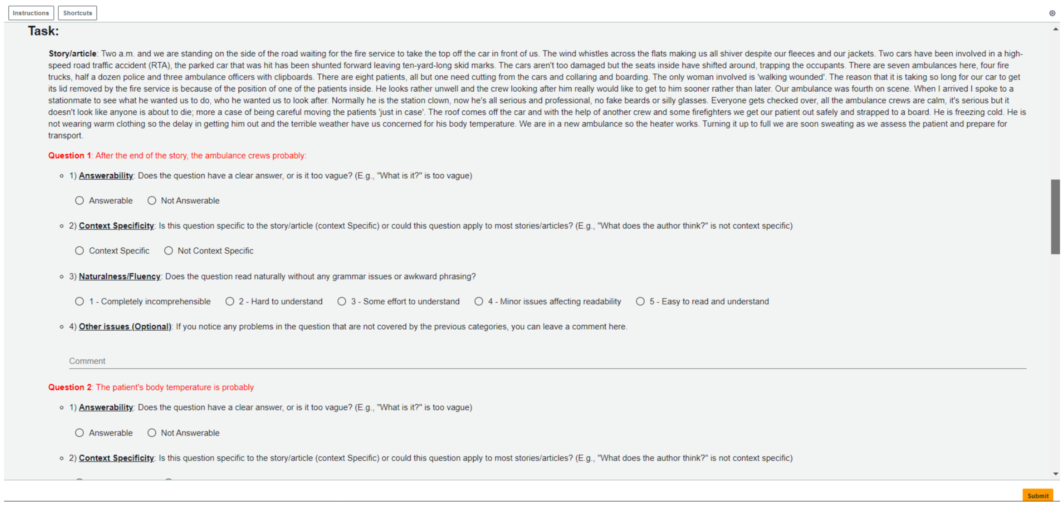
- **Answerability**: Answerable (It is answerable, Sarah laughed after John fell in the puddle)
- **Context Specificity**: Context Specific (Refers specifically to John and the puddle)
- **Naturalness (Fluency)**: 5 - Easy to read and understand (Easy to understand)

Question 2: What can you conclude about the characters of this story?

- **Answerability**: Answerable (It is possible to make conclusions about the characters, e.g. Sarah finds John funny.)
- **Context Specificity**: Not Context Specific (This question will apply to most contexts with stories)
- **Naturalness (Fluency)**: 4 - Minor issues affecting readability (Question is overly wordy)

Question 3: Where is the chacter off John's rubber boot?

- **Answerability**: Not Answerable (John's rubber boot is not mentioned in the story)
- **Context Specificity**: Context Specific (Refers specifically to John)
- **Naturalness (Fluency)**: 2 - Hard to understand (Grammar/spelling issues, hard to understand)



Task:

Story/article: Two a.m. and we are standing on the side of the road waiting for the fire service to take the top off the car in front of us. The wind whistles across the flats making us all shiver despite our fleeces and our jackets. Two cars have been involved in a high-speed road traffic accident (RTA), the parked car that was hit has been shunted forward leaving ten-yard-long skid marks. The cars aren't too damaged but the seats inside have shifted around, trapping the occupants. There are seven ambulances here, four fire trucks, half a dozen police and three ambulance officers with clipboards. There are eight patients, all but one need cutting from the cars and collaring and boarding. The only woman involved is 'walking wounded'. The reason that it is taking so long for our car to get its lid removed by the fire service is because of the position of one of the patients inside. He looks rather unwell and the crew looking after him really would like to get to him sooner rather than later. Our ambulance was fourth on scene. When I arrived I spoke to a stationmate to see what he wanted us to do, who he wanted us to look after. Normally he is the station clown, now he's all serious and professional, no fake beards or silly glasses. Everyone gets checked over, all the ambulance crews are calm, it's serious but it doesn't look like anyone is about to die; more a case of being careful moving the patients 'just in case'. The roof comes off the car and with the help of another crew and some firefighters we get our patient out safely and strapped to a board. He is freezing cold. He is not wearing warm clothing so the delay in getting him out and the terrible weather have us concerned for his body temperature. We are in a new ambulance so the heater works. Turning it up to full we are soon sweating as we assess the patient and prepare for transport.

Question 1: After the end of the story, the ambulance crews probably:

- 1) **Answerability**: Does the question have a clear answer, or is it too vague? (E.g., "What is it?" is too vague)

  ○ Answerable ○ Not Answerable

- 2) **Context Specificity**: Is this question specific to the story/article (context Specific) or could this question apply to most stories/articles? (E.g., "What does the author think?" is not context specific)

  ○ Context Specific ○ Not Context Specific

- 3) **Naturalness/Fluency**: Does the question read naturally without any grammar issues or awkward phrasing?

  ○ 1 - Completely incomprehensible ○ 2 - Hard to understand ○ 3 - Some effort to understand ○ 4 - Minor issues affecting readability ○ 5 - Easy to read and understand

- 4) **Other issues (Optional)**: If you notice any problems in the question that are not covered by the previous categories, you can leave a comment here.

  Comment

Question 2: The patient's body temperature is probably

- 1) **Answerability**: Does the question have a clear answer, or is it too vague? (E.g., "What is it?" is too vague)

  ○ Answerable ○ Not Answerable

- 2) **Context Specificity**: Is this question specific to the story/article (context Specific) or could this question apply to most stories/articles? (E.g., "What does the author think?" is not context specific)

Figure A.1: Screenshots of annotation instructions

be removed from subsequent analyses, since the data is collected anonymously. All of your responses will be completely confidential, as your name and any identifying information will not be stored as part of the research. Only the researchers working on this project (Pro00112093) will have access to the provided information. The data will be stored on reliable servers like those of Google, Qualtrics and Amazon Mechanical Turk. The information you provide may be presented at professional conferences or published in academic journals, however no identifying data will be used. In addition, the data from this study may be used in future research, however any future projects will be approved by the Research Ethics Board prior to analysis.

Benefits and Risks: While this research can potentially contribute to our understanding of computer text generation, there is no significant direct benefit to you. However, you will be thinking rationally about English sentences, and this might help you in real life tasks (like interpreting rules and regulations). You may feel mentally tired, or emotionally stressed during the study, however you are free to take a break anytime you want. You may also be presented with questions which might seem offensive to you, inadvertently as the text is computer generated. There are no foreseeable risks to this study. By participating in this experiment, you are directly contributing to the advancement of the science of computer text generation.

If you would like any additional information, feel free to contact Dr. Alona Fyshe at alona@ualberta.ca. The plan for this study has been reviewed for its adherence to ethical guidelines by a Research Ethics Board at the University of Alberta. If you have any questions about, or wish to clarify, your rights as a research participant, you can contact the Research Ethics Office at +1 (780) 492 2615.

[ ] I have read and understood the information and consent. If you don't accept this, please don't continue with the task as you won't be able to submit it.

Instructions:

Please read through the context (story/article) carefully. Then evaluate each questions as it pertains to the context.

There are 3 criteria for each question:

Answerability: Does the question have a clear answer, or is it too vague? (E.g., "What is it?" is too vague)

Context Specificity: Is this question specific to the story/article (context Specific) or could this question apply to most stories/articles? (E.g., "What does the author think?" is not context specific)

Naturalness (Fluency): Does the question read naturally without grammar issues or does it have some awkward phrasing? You can also reduce the score for this category if you notice any problems in the question that are not covered by the previous categories. (Scale 1-5)

Example:

Story/article: John and Sarah were walking down a hill. Suddenly John slipped and fell into a puddle. Sarah laughed, but John was not happy.

Question 1: What happened after John fell into the puddle?

Answerability: Answerable (It is answerable, Sarah laughed after John fell in the puddle)

Context Specificity: Context Specific (Refers specifically to John and the puddle)

Naturalness (Fluency): 5 - Easy to read and understand (Easy to understand)

Question 2: What can you conclude about the characters of this story?

Answerability: Answerable (It is possible to make conclusions about the characters, e.g. Sarah finds John funny.)

Context Specificity: Not Context Specific (This question will apply to most contexts with stories)

Naturalness (Fluency): 4 - Minor issues affecting readability (Question is overly wordy)

Question 3: Where is the chacter off John's rubber boot?

Answerability: Not Answerable (John's rubber boot is not mentioned in the story)

Context Specificity: Context Specific (Refers specifically to John)

Naturalness (Fluency): 2 - Hard to understand (Grammar/spelling issues, hard to understand)

Task:

Story/article: [Context]

Question 1: [Question 1]

1) Answerability: Does the question have a clear answer, or is it too vague? (E.g., "What is it?" is too vague)

2) Context Specificity: Is this question specific to the story/article (context Specific) or could this question apply to most stories/articles? (E.g., "What does the author think?" is not context specific)

3) Naturalness/Fluency: Does the question read naturally without any grammar issues or awkward phrasing?

4) Other issues (Optional): If you notice any problems in the question that are not covered by the previous categories, you can leave a comment here.

Question 2: [Question 2]

1) Answerability: Does the question have a clear answer, or is it too vague? (E.g., ”What is it?” is too vague)

2) Context Specificity: Is this question specific to the story/article (context Specific) or could this question apply to most stories/articles? (E.g., ”What does the author think?” is not context specific)

3) Naturalness/Fluency: Does the question read naturally without any grammar issues or awkward phrasing?

4) Other issues (Optional): If you notice any problems in the question that are not covered by the previous categories, you can leave a comment here.

Question 3: [Question 3]

1) Answerability: Does the question have a clear answer, or is it too vague? (E.g., ”What is it?” is too vague)

2) Context Specificity: Is this question specific to the story/article (context Specific) or could this question apply to most stories/articles? (E.g., ”What does the author think?” is not context specific)

3) Naturalness/Fluency: Does the question read naturally without any grammar issues or awkward phrasing?

4) Other issues (Optional): If you notice any problems in the question that are not covered by the previous categories, you can leave a comment here.

Question 4: [Question 4]

1) Answerability: Does the question have a clear answer, or is it too vague? (E.g., ”What is it?” is too vague)

2) Context Specificity: Is this question specific to the story/article (context Specific) or could this question apply to most stories/articles? (E.g., ”What does the author think?” is not context specific)

3) Naturalness/Fluency: Does the question read naturally without any grammar issues or awkward phrasing?

4) Other issues (Optional): If you notice any problems in the question that are not covered by the previous categories, you can leave a comment here.

Question 5: [Question 5]

1) Answerability: Does the question have a clear answer, or is it too vague? (E.g., ”What is it?” is too vague)

2) Context Specificity: Is this question specific to the story/article (context Specific) or could this question apply to most stories/articles? (E.g., ”What does the author think?” is not context specific)

3) Naturalness/Fluency: Does the question read naturally without any grammar issues or awkward phrasing?

4) Other issues (Optional): If you notice any problems in the question that are not covered by the previous categories, you can leave a comment here.

Question 6: [Question 6]

1) Answerability: Does the question have a clear answer, or is it too vague? (E.g., ”What is it?” is too vague)

2) Context Specificity: Is this question specific to the story/article (context Specific) or could this question apply to most stories/articles? (E.g., ”What does the author think?” is not context specific)

3) Naturalness/Fluency: Does the question read naturally without any grammar issues or awkward phrasing?

4) Other issues (Optional): If you notice any problems in the question that are not covered by the previous categories, you can leave a comment here.

Question 7: [Question 7]

1) Answerability: Does the question have a clear answer, or is it too vague? (E.g., ”What is it?” is too vague)

2) Context Specificity: Is this question specific to the story/article (context Specific) or could this question apply to most stories/articles? (E.g., "What does the author think?" is not context specific)

3) Naturalness/Fluency: Does the question read naturally without any grammar issues or awkward phrasing?

4) Other issues (Optional): If you notice any problems in the question that are not covered by the previous categories, you can leave a comment here.

A.5 Automatic Evaluation Calculations

We present the formulas required to calculate the automatic evaluation methods in Table A.4.

Method(s)	**Formula**	**Variables**
Best Reference & Cartesian Product & Multi-METEOR	$\frac{\sum_{\vec{p} \epsilon P} METEOR(p_1,p_2)}{length(P)}$	P = Pairs selected by matching method p_1 = Reference question in pair $\vec{p}$ p_2 = Candidate question in pair $\vec{p}$ C_2 = Candidate question embedding $METEOR$ = The METEOR function
Fréchet BERT Distance (FBD)	$\sqrt{\begin{matrix} \lvert\lvert m_1 - m_2 \rvert\rvert_2^2 \\ +Tr(C_1 + C_2 - 2(C_1C_2)^{\frac{1}{2}}) \end{matrix}}$	m_1 = Mean reference question embedding m_2 = Mean candidate question embedding C_1 = Reference question embedding covariance matrix C_2 = Candidate question embedding covariance matrix Tr = Trace of the matrix (Sum of values along the main diagonal)
MS-Jaccard4	$score_n = \frac{\sum_{g \epsilon G_n} min\{C_n(g,S_1),C_n(g,S_2)\}}{\sum_{g \epsilon G_n} max\{C_n(g,S_1),C_n(g,S_2)\}}$ MS-Jaccard4 $= \left(\prod_{n=1}^{4} score_n\right)^{\frac{1}{4}}$	S_1 = Set of reference questions S_2 = Set of candidate questions G_n = Set of n-grams in $S_1 \cup S_2$ (of length n) $C_n(g,S)$ = Normalized count of n-gram, g in set S

Table A.4: Calculations for each of the question generation evaluation methods.

A.6 T5 WTA Prompts

We present the prompts used by the T5 WTA model for both the QuAIL and SBRCS datasets in Table A.5.

Dataset	**Prompt**
QuAIL	Belief States </s>
	Causality </s>
	Character Identity </s>
	Entity Properties </s>
	Event Duration </s>
	Factual </s>
	Subsequent State </s>
	Temporal Order </s>
SBRCS	Basic Story Elements </s>
	Character Traits </s>
	Close Reading </s>
	Figurative Language </s>
	Inferring </s>
	Predicting </s>
	Summarizing </s>
	Visualizing </s>
	Vocabulary </s>

Table A.5: T5 WTA Prompts for the QuAIL and SBRCS datasets.