Generation of Student Questions for Inquiry-based Learning

Kevin Ros, Maxwell Jong, Chak Ho Chan, ChengXiang Zhai
University of Illinois at Urbana-Champaign
{kjros2, mjong3, chchan2, czhai}@illinois.edu

Abstract

Asking questions during a lecture is a central part of the traditional classroom setting which benefits both students and instructors in many ways. However, no previous work has studied the task of automatically generating student questions based on explicit lecture context. We study the feasibility of automatically generating student questions given the lecture transcript windows where the questions were asked. First, we create a data set of student questions and their corresponding lecture transcript windows. Using this data set, we investigate variants of T5, a sequence-to-sequence generative language model, for a preliminary exploration of this task. Specifically, we compare the effects of training with continuous prefix tuning and pre-training with search engine queries. Question generation evaluation results on two MOOCs show that that pre-training on search engine queries tends to make the generation model more precise whereas continuous prefix tuning offers mixed results.

1 Introduction

It is difficult to understate the importance of asking questions in educational settings. Well-formed questions serve many purposes, including testing student understanding, encouraging exploration of new knowledge, guiding research directions, and developing critical thinking skills (Cotton, 1988). Question-asking also has many benefits for both students and instructors because of its implicit coupling to the context in which questions are asked. For example, instructors can use student questions as implicit feedback to gauge the difficulty of a lecture or to anticipate and update pain points in lecture content. For students, upon hearing a question, they may find it helpful to think about possible answers or to connect the question to their own thought process, thus encouraging inquiry-based learning (Edelson et al., 1999).

However, the benefits of question-asking are much harder to realize in online, asynchronous class settings compared to traditional, in-person class settings. In the latter case, the students and the instructors are co-located and generally, everyone is aware of the current context (i.e., the lecture) and the question. In the former case, the students watch the lectures independently of each other. If a student independently leverages an online search engine to answer a question, then there is no way for their peers and instructors to benefit from the question being asked.

Automatically generating realistic student questions would bring significant benefit to online, asynchronous class settings. For example, instructors could use synthetic student questions to augment lecture videos with additional material. And students could use synthetic student questions to guide studying or to test understanding. Moreover, the synthetic student questions could act as a discussion guide among students and instructors by helping them focus on difficult material.

In this paper, we study how to generate such student questions automatically from given lecture transcript windows. Despite the large amount of previous work regarding question generation (Zhang et al., 2021) (see Section 2 for a detailed review), no previous work has studied our problem setup, as in our case, the answers to the questions may not be available in the lecture transcript content and the questions themselves may not provide enough context to be understood and answered on their own. In virtually all of the data sets used in the existing work, the answers to the questions to be generated are generally assumed to be either directly available or indirectly inferable from the text context. To facilitate the study of this new application scenario of question generation, we create a new data set by collecting and using two MOOC (Massive Open Online Course) transcripts along with 536 questions asked by students

Proceedings of the 15th International Conference on Natural Language Generation, pages 186 - 195

of the MOOCs. Each question includes the corresponding MOOC lecture timestamp window for when the question was asked, thus enabling us to evaluate various context-based question generation approaches.

As an initial investigation of this new task, we focus our exploration on the question generation performance of the generative language model T5 (Raffel et al., 2019) in various settings, leaving a full exploration of different models as future work. Motivated by our small number of training examples, we explore the performance effects of continuous prefix tuning, which has been shown to perform well on natural language generation tasks in low-data settings (Li and Liang, 2021). Additionally, we examine the effects of using docTTTTTquery, a T5 model pre-trained with search engine query generation (Nogueira et al., 2019a), on student question generation. Specifically, we investigate the following research questions:

RQ1: How does pre-training on search engine query generation affect student question generation performance?

RQ2: How does continuous prefix tuning affect student question generation performance?

We find that pre-training on search engine queries tends to make the generation models more precise and that continuous prefix tuning tends to outperform traditional fine-tuning (albeit with mixed significance testing results). Overall, we conclude that it is feasible and promising to use modern machine learning and natural language processing techniques to automatically generate student questions from explicitly-mentioned lecture context in low-data settings.

2 Related Work

Question Generation (Rus et al., 2010; Mazidi and Tarau, 2016) has been extensively studied, initially in the context of generating questions for educational purposes (Mitkov et al., 2006; Kurdi et al., 2020), later with broader application contexts beyond education, such as question answering (Duan et al., 2017) and conversational agents (Wang et al., 2018a).

The survey (Pan et al., 2019) provides a detailed discussion of the major data sets used in recent work on neural question generation and the different levels of questions supported by those data sets, concluding that the current methods cannot work well for generating deep questions. Virtually all the existing data sets have been generated based on answers in the provided text context (i.e., answer-aware (Zhang et al., 2021)) with perhaps only one exception, which is the LearningQ data set (Chen et al., 2018), where the problem formulation does not include the use of answer when generating a question (i.e., answer-agnostic (Zhang et al., 2021)). Our work is closest to (Chen et al., 2018) in that our formulation of question generation is also answer-agnostic. However, the questions included in the LearningQ data set have been filtered to ensure that the questions included are context-complete. In other words, a question in the LearningQ data set must contain sufficient contextual information on its own to enable other learners to answer the question. Because the structure of our data set explicitly guarantees a reference to the point in the lecture where the question was asked, we can keep questions which don't provide much context themselves (e.g., "Could you be more specific?"). This coupling between lecture and question provides the basis for a new application scenario of question generation where the generated questions are meant to encourage inquiry-based learning (Edelson et al., 1999) for students consuming online lectures. Thus, our data set and approach facilitate a study of how to generate interesting open-ended deep questions using lecture context.

(Ko et al., 2020) collected questions without answers from readers of news articles. They asked study participants to read the first paragraph of various news articles one sentence at a time. If the participant had a question about the sentence, they were instructed to highlight the location of the sentence (e.g., a word or phrase) and write down their question. They collected approximately 19,000 questions and corresponding contexts. However, many of the questions tended to be simpler than the ones collected for our study or by (Chen et al., 2018), and they tended to be answerable by the following sentences in the paragraphs. Moreover, the context sizes selected in (Ko et al., 2020) were smaller than the student-selected lecture windows in our data set.

The survey (Zhang et al., 2021) provides an up-to-date comprehensive review of different lines of work with detailed categorization of the task formulation and comparison of the major approaches

including both rule-based approaches (Lindberg et al., 2013) and modern neural network-based approaches (Du et al., 2017; Duan et al., 2017; Lewis et al., 2019). In our work, because the data set has a small number of training examples, while the question structures can be quite complex, we focused on exploring the use of general pre-trained language models (T5) and prefix tuning / pre-training on search engine queries to address the technical challenges.

More generally, the notion of question generation has been studied from the perspective of information retrieval (Nogueira et al., 2019a,b). Here, the authors expanded documents with the queries for which the documents would be relevant. For a given search query, a document's relevance score was computed using both the document's content and its respective generated queries. We leverage their fine-tuned query generation model, docTTTTTquery, as the basis for answering **RQ1**.

Previous work has found that learners engage heavily with in-video quizzes (Kovacs, 2016). However, such quizzes are usually designed manually. The methods that we explore can be potentially used to automatically generate in-video questions to enhance learner engagement. Automatic generation of quiz questions for testing learners' knowledge has also been attempted. For example, Wang et al. introduced QG-Net, a recurrent neural network-based model that can generate quiz questions from educational content (Wang et al., 2018b). Although such questions were also generated based on educational content, their answers were generally available in the educational content from which the questions are generated; in contrast, in our work, the answers to those questions generated are generally not available directly in the educational content.

3 The Lecture-Question Data Set

Because our exact problem setup has not been studied before, there does not exist any data set that we can use for our experiments. Thus, we created a data set from student questions previously submitted to two MOOCs available on Coursera, titled *Text Retrieval and Search Engines* and *Text Mining and Analytics* . The students were enrolled in a class which used the two MOOCs as the major lectures. There are a total of 90 lecture across the MOOCs, and each lecture contains a complete transcript of the audible instruction, with frequent and regular timestamps. Students were asked to submit any questions that they had about the MOOC lecture content, and to include a reference to the lecture name and timestamps where the question occurred. The question submission template was as follows:

<Lecture name, start time, end time, question>

Both of the MOOC transcripts and the submitted student questions are in English.

3.1 Preprocessing and Data Availability

To clean the data, we filtered out all questions with malformed annotations or missing timestamps. Next, we attempted to map each question to the respective lecture transcript window. The overall cleaning process resulted in a data set of 536 (lecture window text, question text) pairs. This is the data set that we used to quantify the performance of question generation. The original and filtered data sets, as well as the complete MOOC transcripts, are available on GitHub.[1] We obtained IRB approval and permission from the MOOC author to release the anonymous data. Note that we removed any student-identifiable information from the data set.

3.2 Basic Properties of the Student Questions

This section describes the characteristics of the student questions from our filtered data set. For the 536 questions, the mean number of words per question is 18, and the median number of words is 15. A similar skew is also present in the corresponding lecture windows. The mean number of words in each lecture window is 210, and the median number of words is 132. Moreover, the mean number of seconds in each window is 92, whereas the median number of seconds is 60.

Unigrams	Trigrams
is (264)	what is the (65)
how (188)	how do we (30)
what (181)	the meaning of (13)
does (107)	the difference between (12)
why (104)	why do we (11)
are (103)	is it possible (9)

Table 1: A list of some of the most frequent unigrams and trigrams in student questions. The number in parentheses indicates the occurrence frequency.

[1] https://github.com/kevinros/INLG2022StudentQuestions

Question Examples
What is the point of compression? Will the access times really be that impactful to the overall indexing?
Are the doc-ids sorted with the term-ids in the "local" sort?
Can we get more examples of using gamma-code?
How does the gamma-code intergar compression method work? I did not understand the example from the video
I'm still very confused how integer compression actually reduces size of storage since some of the examples make it seem like you're using more bits than before on some inputs

Table 2: A few example questions from the lecture-question data set.

Table 1 depicts a list of some of the most common unigrams and trigrams present in student questions. Interestingly, many questions are concerned with the meaning or difference of the referenced content. The most common interrogative words are "how", "what", and "why". A few examples of questions from our data set are presented in Table 2. Note that there is significant noise in the data set: there are misspellings (e.g., "interger" in the third question), multiple questions submitted as one question, and general expressions of confusion instead of questions. Although we expect that our models would perform better if we removed noise through additional preprocessing, we felt that it was important to remain as close as possible to the original data to reflect a scaled learning scenario where manual preprocessing is impractical.

There are some clear limitations of our collected lecture-question data set. The size of the data set and the MOOC topic similarity make it difficult to know if our findings generalize to different data sets. Also, the lecture transcripts and questions are written in English, which certainly limit applicability. However, given the overall lack of data for this problem setting, we hope that our data set and methods can offer a starting point for future researchers and educators to extend student question generation into more general settings.

4 Methods for Question Generation

4.1 Problem Formulation

We now briefly formalize the proposed task of student question generation from lecture transcripts. Our data set consists of (L_i, Q_i) pairs. In each pair, lecture window $L_i = (w_j)_{j=1}^{n_i}$ is a sequence of n_i word tokens. The start and end positions of lecture window L_i are determined by the start and end timestamps submitted in the question Q_i. Question $Q_i = (q_k)_{k=1}^{m_i}$ is a sequence of m_i word tokens. We aim to generate question Q_i given its corresponding lecture window L_i.

Formally, we model question generation as a sequence-to-sequence language generation task. Let model M be a sequence-to-sequence language model initialized with trainable parameters ϕ. Our goal is to maximize the probability of $M_\phi(Q_i|L_i)$. In other words, the input to model M is lecture window L_i and the desired output is the corresponding student question Q_i.

4.2 RQ1: T5 and docTTTTTquery

For the sequence-to-sequence language model architecture, we use T5 (Raffel et al., 2019), which is based on the standard encoder-decoder transformer architecture (Vaswani et al., 2017). We choose T5 due to its state-of-the-art performance in the text summarization task, which closely resembled our question generation problem formulation (i.e., "summarizing" the lecture window as a question). The idea behind T5's original implementation was to improve performance by having a single model learn many different tasks (translation, summarization, classification, etc.) as sequence-to-sequence text tasks via discrete fixed prompt instructions (Liu et al., 2021). Specifically, each training example began with a pre-defined discrete prompt (e.g., "translate English to German:") which served the purpose of instructing the model to handle the input data according to the task described in the prompt.

We test two existing instantiations of T5, namely t5-base and docTTTTTquery (Nogueira et al., 2019a). The former was trained in accordance with the original T5 paper and the latter is a version of t5-base fine-tuned on query generation given relevant passages using the MS-MARCO data set (Nguyen

et al., 2016). To answer **RQ1**, we fine-tune both models to determine if a model pre-trained to generate search engine queries offers any performance benefits on our student question generation task. For completeness, we also include the performance of the base docTTTTTquery model that is not fine-tuned on our data set.

4.3 RQ2: Continuous Prefix Tuning

Continuous prefix tuning, proposed by (Li and Liang, 2021), is a method for fine-tuning large generative models that has been shown to perform well in low-data scenarios. We are interested in studying the effects of continuous prefix tuning for generating student questions. Thus, we adapt Li and Liang's approach to T5 and measure the performance on our collected lecture-question data set. We now provide a formal overview of their continuous prefix tuning applied to question generation.

For the continuous prefix tuning setting, language model M is still initialized using pre-trained parameters ϕ. Consider a continuous prefix $p \in \mathbb{R}^{d \times n}$. Here, d is the input embedding dimension of model M and $n \in \mathbb{N}_{\geq 0}$ is the chosen length of the prefix, which must be strictly less than the maximum sequence length of M. The input to the language model then becomes $L_i' = [p; L_i]$. The goal is to maximize $M_{\phi,p}(Q_i|L_i')$ where parameters ϕ are fixed and prefix p is free. In other words, we freeze the original parameters of the language model and aim to learn the values of p which best help M generate Q_i. Note that p is the same across all training, validation, and testing pairs.

For our implementation, we also follow the continuous prefix tuning reparameterization approach of (Li and Liang, 2021), which they found to increase training stability. As noted earlier, the prefix p is a continuous matrix with values determined by the fine-tuning process on the training data. To determine the values of p, we fix hyperparameter $n' \in \mathbb{N}_{>0}, n' \leq n$ and randomly initialize $p' \in \mathbb{R}^{d \times n'}$. Then, we define p as the output of a two-layer neural network N parameterized by ψ. In other words, we compute prefix p as $p = N_\psi(p')$, where parameters ψ are learned during training. For our experiments, N_ψ is a fully-connected two layer neural network with hidden dimension h.

To answer **RQ2**, we fine-tune t5-base and docTTTTTquery with the continuous prefix modification and compare the resulting performance metrics to the traditionally fine-tuned models.

4.4 Evaluation

We randomly split the 90 MOOC lectures into 85 training lectures and 5 testing lectures. We split on the lecture level to avoid the possibility of the model seeing overlapping windows during training and testing. This split results in 483 questions in the training set and 53 questions in the testing set. Then, for three iterations, we randomly hold out 10 lectures from the 85 training lectures as a validation set. The validation lecture sets are disjoint across all iterations, and each model is trained and validated on the same splits. The hyperparameters for each model are selected based on the highest averaged ROUGE-1 F_1 score over the validation splits. After selecting the hyperparameters, we retrain the model on the entire 85 lecture training set. To measure the performance of our trained models, we report the single-run precision, recall, and F_1 score for the ROUGE-1, ROUGE-2, and ROUGE-L measurements averaged across all generated and ground-truth questions in the testing set. All ROUGE scores are computed using the default settings of the rouge-score python package.[2] Regarding hyperparameter selection, the number of epochs ranges from $[1, 10]$ and the learning rate ranges from {1e-6, 1e-5, 1e-4}.

5 Question Generation Results

Table 3 contains the single-run ROUGE-1, ROUGE-2, and ROUGE-L scores on the test set for each best-performing model on the validation set. The first column lists the name of each model. "FT" refers to traditional fine-tuning and "Prefix" refers to continuous prefix tuning. For the run labeled "docTTTTTquery", we evaluate the model on the test set without any training. There is no corresponding "t5-base" run because the original model was not trained to generate questions. The second column labeled "R" is the recall, the third column labeled "P" is the precision, and the fourth column labeled "F_1" is the F_1 score.

5.1 Hyperparameters

The final hyperparameter selections for each model are reported in Table 4. Each validation run takes approximately one hour on a single Nvidia GeForce 1070x GPU with a batch size of one. Note that the number of parameters are essentially the same

[2] https://pypi.org/project/rouge-score/ Rouge-score is released under Apache License 2.0, which permits research use.

Model	R	P	F_1
	ROUGE-1 (%)		
t5-base FT	20.06	14.47	14.82
t5-base Prefix	20.13	21.56	18.63
docTTTTTquery	14.41	25.17	16.83
docTTTTTquery FT	15.70	23.34	17.45
docTTTTTquery Prefix	17.19	24.00	18.74
	ROUGE-2 (%)		
t5-base FT	1.697	1.656	1.502
t5-base Prefix	3.267	3.391	3.043
docTTTTTquery	3.237	4.596	3.358
docTTTTTquery FT	4.011	4.730	3.903
docTTTTTquery Prefix	4.790	6.247	5.010
	ROUGE-L (%)		
t5-base FT	15.82	11.57	11.77
t5-base Prefix	16.89	17.65	15.47
docTTTTTquery	13.17	22.32	15.18
docTTTTTquery FT	14.34	20.64	15.76
docTTTTTquery Prefix	15.47	21.00	16.73

Table 3: The recall, precision, and F_1 scores for ROUGE-1, ROUGE-2, and ROUGE-L measurements for the question generation approaches on the test set. "FT" refers to traditional fine-tuning and "Prefix" refers to continuous prefix tuning.

across both models (220M (Raffel et al., 2019)), as docTTTTTquery is a fine-tuned version of t5-base. Additionally, the prefix reparameterization parameters can be dropped once the model is trained. Preliminary experiments indicated that larger prefixes tended to perform better, so we fix the prefix length (n, in Table 4) to be sufficiently large. The random seed is set to 42 across all runs. All other hyperparameters are set to the default settings.

5.2 Answering RQ1

To answer **RQ1**, we compare the performance of the t5-base models with the performance of the docTTTTTquery models. Beginning with the FT models for both cases, we find that docTTTTTquery FT has lower ROUGE-1 and ROUGE-L recall scores than t5-base FT but higher ROUGE-1 and ROUGE-L precision scores. Additionally, docTTTTTquery FT has a higher recall and precision for ROUGE-2. In all three ROUGE cases, docTTTTTquery FT has a higher F_1 score than t5-base FT. There is also a similar trend for the Prefix models. Namely, docTTTTTquery Prefix has lower ROUGE-1 and ROUGE-L recall scores than t5-base Prefix, but higher precision and F_1 scores. Moreover, docTTTTTquery Prefix has a higher recall, precision, and F_1 score for the ROUGE-2 measurement. Based on the averages of the ROUGE scores, we see that the docTTTTTquery models are generally more precise than the t5-base models. To test the statistical significance of the precision improvement, we performed a one-tailed Wilcoxon signed-rank test comparing the precision of each t5-base run to its respective docTTTTTquery run. We selected the Wilcoxon signed-rank test because of its non-parametric property, as the distributions of precision appear to be non-normal. All runs were significant at $p = 0.05$, except for the ROUGE-1 precision between the t5-base Prefix model and the docTTTTTquery Prefix model ($p = 0.077$), and the ROUGE-L precision between the t5-base Prefix model and the docTTTTTquery Prefix model ($p = 0.080$). In conclusion, pre-training on search engine query generation appears to offer clear benefit in increasing the precision, though the benefit appears to be more for traditional fine-tuning.

5.3 Answering RQ2

To answer **RQ2**, we compare the performance of the FT model variants with the Prefix model variants. Beginning with the t5-base models, we find that the runs have similar recall scores for ROUGE-1, whereas the t5-base Prefix has higher recall scores for ROUGE-2 and ROUGE-L. The precision and F_1 scores for all ROUGE measurements are higher for t5-base Prefix. For the docTTTTTquery models, we find that docTTTTTquery Prefix has the highest recall across all three ROUGE measurements. There is no clear trend for the precision. However, the increases in recall are enough for docTTTTTquery Prefix to have the highest F_1 scores for all three ROUGE measurements. Similar to the previous research question, we performed a one-tailed Wilcoxon signed-rank test comparing the F_1 scores of each Prefix model to its respective FT model, in order to test for improvement. Only the ROUGE-1 and ROUGE-L scores between the t5-base FT model and the t5-base Prefix model were significant at $p = 0.05$. Note that the ROUGE-2 score comparison between the docTTTTTquery FT model and the docTTTTTquery Prefix model had $p = 0.055$. From these results, there seems to be marginal benefit for using continuous prefix tuning in a low-data setting to generate student questions.

Model	Learning Rate	Epochs	n'	h	n	dropout	hidden activation
t5-base FT	1e-5	8	-	-	-	-	-
t5-base Prefix	1e-5	7	20	800	100	0.5	tanh
docTTTTTquery	-	-	-	-	-	-	-
docTTTTTquery FT	1e-5	7	-	-	-	-	-
docTTTTTquery Prefix	1e-4	7	20	800	100	0.5	tanh

Table 4: Best-found hyperparameters for each trained model on the validation set. The last five columns correspond to the prefix reparameterization hyperparameters described in Section 4.3.

5.4 Qualitative Analysis

Because our testing set is small, quantitative analysis and significance testing may not offer a clear picture into the qualitative differences between the models. Therefore, we perform a brief qualitative comparison among the ground truth questions and the generated questions for each model. A few question examples are presented in Table 5, and the corresponding lecture windows are presented in Table 6. These examples were hand-selected to demonstrate some interesting characteristics of the question generation models.

The selected examples presented in Table 5 offer some possible explanations for the variations in ROUGE scores from Table 3. Notably, the docTTTTTquery models typically generate shorter questions than the t5-base models. The average number of words generated per question by the t5-base FT and t5-base Prefix model were 24 and 15 respectively, whereas the docTTTTTquery, docTTTTTquery FT, and docTTTTTquery Prefix averages were 8, 10, and 10, respectively. Additionally, t5-base FT sometimes generates multiple subquestions for a single ground truth. This may explain why the docTTTTTquery models generally have higher precision scores but lower recall scores. We believe that the docTTTTTquery models generate shorter questions because search engine queries tend to be much shorter than our collected student questions (Craswell et al., 2020). Moreover, the similarity of generated questions (e.g., between the docTTTTTquery models in the second example) may help explain the non-significant score results. It is also important to note that non-significant differences in ROUGE score may not imply non-significant differences in question meaning or quality. In the first example of Table 5, the docTTTTTquery FT generated question and the docTTTTTquery Prefix generated question only differ by a word ("delta" versus "gamma"). Despite this small difference, the questions have completely different meaning and answers.

Another interesting observation is the apparent inability of the models to capture mathematical expressions. For example, in the second question group, all models miss "(1+logx)" and "x-2(logx)". One possible explanation is that the transcript does not fully capture these equations, which can be observed in the second row of Table 6. This might be addressable by incorporating multi-modal contexts (e.g., lecture transcripts *and* lecture slides), which is an interesting direction for future work.

We also observe cases where there is a mismatch between the topics present in the ground truth and the topics present in the generated questions. This seems to indicate that a more controlled generation approach may be necessary.

5.5 Limitations

There are a few limitations of our generation experiments. First, due to the difficulty in collecting the data for our study, the data set that we have managed to create at this point is small. While the data set was already useful to allow us to investigate multiple variants of T5-based models to make some preliminary conclusions, it is unclear to what extent we can generalize those conclusions. Thus, in the future, it is important to further increase the size of the data set, which we are planning to do by continuously collecting the questions from students in future classes. With larger data sets, we can also more thoroughly explore the cutting-edge models for this task. Second, our notion of question "ground truth" may not be the best way to capture students' questions. This is because there can be many equally-valid questions for a given lecture window, even if they are different from one another. Third, the ROUGE scores may not effectively capture the high-level semantics of a question. That is, small changes in question word choice or structure can result in similar ROUGE scores while drastically altering the meaning.

Model	Question
Ground Truth	Does is the delta-code use gamma-code twice recursively?
t5-base FT	What is the difference between delta coding and delta coding? Is it possible to use delta coding for inverted index distribution?
t5-base Prefix	What is the difference between delta and gamma?
docTTTTTquery	what is gamma coding
docTTTTTquery FT	what is the difference between delta and delta coding?
docTTTTTquery Prefix	what is the difference between delta and gamma coding?
Ground Truth	What is the reasoning for making the first (1+logx) unary and the x-2(logx) uniform? The method for encoding seem random to me.
t5-base FT	What is the problem? Is there a way to use binary code to calculate the remaining part of the value of x?
t5-base Prefix	what is the difference between unary code and binary code?
docTTTTTquery	what is the difference between unary and binary code
docTTTTTquery FT	what is the difference between unary and binary code?
docTTTTTquery Prefix	what is the difference between unary and binary code?
Ground Truth	What is the risk associated with discarding documents that are potentially relevant?
t5-base FT	is there any way to determine the most likely relevance in the documents?
t5-base Prefix	What is the purpose of having a diverse set of ranking methods?
docTTTTTquery	what is diverse set of ranking methods
docTTTTTquery FT	what is the difference between a diverse set of ranking methods and an unjudged pool of documents?
docTTTTTquery Prefix	what is meant by having a pool of relevant documents that aren't being ranked?

Table 5: A few examples of the ground truth question compared to each model's generated question.

Nevertheless, the scores presented in Table 3 and the examples presented in Table 5 indicate that it is possible to generate meaningful student questions from lecture content with low amounts of data. Moreover, we find benefits in the use of continuous prefix tuning and in the use of search engine queries to fine-tune pre-trained language models for question generation. Overall, we hope that our results can help guide future researchers for designing student question generation models in similar low-data settings.

6 Conclusion and Future Work

In this paper, we studied a new application scenario of question generation, where the goal is to generate interesting questions that can promote inquiry-based learning for students watching online lecture videos. The task is different from many existing question generation tasks in that the answers to the questions may not be available in the text context used to generate a question. We created and released a new data set for studying this problem. We also studied how to use various T5 models to solve the problem effectively. Experimental results showed that the task is challenging, but continuous prefix tuning and pre-training on search engine queries show promise in the direction of generating coherent and relevant questions in spite of limited training data. Moreover, the ability to use search engine queries as pre-training data hints at the scalability of precise student question generation due to the wide availability of queries.

Full exploration of the potential of the proposed methods and further evaluation of the benefits of the generated questions for real learners are important directions for future work. One particularly promising albeit difficult area for future work is to consider a more fine-grained question generation approach by conditioning the generation model not only on the lecture context but also the student context (i.e., a student's background knowledge of a subject). Additionally, generation could be framed in the context of multiple lecture locations at once instead of a single window. Another possible direction is to investigate methods for using language models to generate student questions about mathe-

Question	Lecture Window
Does is the delta-code use gamma-code twice recursively?	except that you replace the unary prefix with the gamma code. So that's even less conservative than gamma code, in terms of avoiding the small integers. So that means it's okay if you occasionally see a large number. It's, it's, you know, it's okay with delta code. It's also fine with gamma code. It's really a big loss for unary code, and they are all operating, ...
What is the reasoning for making the first (1+logx) unary and the x-2(logx) uniform? The method for encoding seem random to me.	this is basically the same uniform code and binary code are the same. And we're going to use this code to code the remaining part of the value of x. And this is basically, precisely, x minus 1, 2 to the flow of log of x. So the unary code or basically code with a flow of log of x, well, I added one there, and here. But the remaining part will, we using uniform code to actually code the difference between the x and
What is the risk associated with discarding documents that are potentially relevant?	We would first choose a diverse set of ranking methods, these are types of retrieval systems. And we hope these methods can help us nominate likely relevance in the documents. So the goal is to pick out the relevant documents.. It means we are to make judgements on relevant documents because those are the most useful documents from the users perspective...

Table 6: The lecture windows corresponding to the questions presented in Table 5.

matical formulas. Finally, it would be interesting to further explore alternative controlled methods for question generation in low-data settings, such as few-shot approaches or simpler, rule-based approaches.

Our proposed task of generating questions from indicated lecture content is inherently applied in nature, as it is centered around learners and instructors in an educational setting. Thus, future work should also consider the direct utility of learners and instructors as a future measure of model effectiveness. And as we discussed in Section 5.5, the "ground truth" question for a given context may not be consistent across individuals. Or, for a given individual, different questions may have different dimensions of utility. This leads to an interesting direction of exploring the types of questions that individuals find useful in various contexts.

In a more general sense, our problem setting could be cast in an outward direction by examining the reasons behind why learners ask questions or by examining the linguistic structures and characteristics of the asked questions. Better understandings of these directions may help drive more efficient or simpler model architectures, training procedures, and evaluation metrics.

With the growth of online education, particularly in the context of MOOCs, both instructors and students will find it valuable to be able to better understand, contemplate, and anticipate question-based interactions with course material. We thus hope our preliminary exploration provides a basis for future work on question generation in this application context, eventually creating natural language generation techniques that can be deployed on an online learning platform to automatically generate relevant questions to many online lectures and support inquiry-based learning for many online students.

7 Acknowledgements

This work is supported in part by the National Science Foundation under Grant No. 1801652. We would also like to thank Dr. Heng Ji for her insightful comments.

8 Ethical Impact

As with any natural language generation approach that leverages large pre-trained models, there is the possibility of generating biased or offensive content. Careful consideration is needed to apply these findings to live scenarios, as there likely are many untested or unexpected behaviors of the underlying language models.

References

Guanliang Chen, Jie Yang, Claudia Hauff, and Geert-Jan Houben. 2018. LearningQ: a large-scale dataset

for educational question generation. In *Twelfth International AAAI Conference on Web and Social Media*.

Kathleen Cotton. 1988. Classroom questioning. *School improvement research series*, 5:1–22.

Nick Craswell, Daniel Campos, Bhaskar Mitra, Emine Yilmaz, and Bodo Billerbeck. 2020. ORCAS: 20 million clicked query-document pairs for analyzing search. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 2983–2989.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. *arXiv preprint arXiv:1705.00106*.

Nan Duan, Duyu Tang, Peng Chen, and Ming Zhou. 2017. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874.

Daniel C Edelson, Douglas N Gordin, and Roy D Pea. 1999. Addressing the challenges of inquiry-based learning through technology and curriculum design. *Journal of the learning sciences*, 8(3-4):391–450.

Wei-Jen Ko, Te-Yuan Chen, Yiyan Huang, Greg Durrett, and Junyi Jessy Li. 2020. Inquisitive question generation for high level text comprehension. *arXiv preprint arXiv:2010.01657*.

Geza Kovacs. 2016. Effects of in-video quizzes on mooc lecture viewing. In *Proceedings of the third (2016) ACM conference on Learning@ Scale*, pages 31–40.

Ghader Kurdi, Jared Leo, Bijan Parsia, Uli Sattler, and Salam Al-Emari. 2020. A systematic review of automatic question generation for educational purposes. *International Journal of Artificial Intelligence in Education*, 30(1):121–204.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

David Lindberg, Fred Popowich, John Nesbit, and Phil Winne. 2013. Generating natural language questions to support learning on-line. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 105–114.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *arXiv preprint arXiv:2107.13586*.

Karen Mazidi and Paul Tarau. 2016. Infusing NLU into automatic question generation. In *Proceedings of the 9th International Natural Language Generation conference*, pages 51–60.

Ruslan Mitkov, Ha Le An, and Nikiforos Karamanis. 2006. A computer-aided environment for generating multiple-choice test items. *Natural language engineering*, 12(2):177–194.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*.

Rodrigo Nogueira, Jimmy Lin, and AI Epistemic. 2019a. From doc2query to docTTTTTquery. *Online preprint*.

Rodrigo Nogueira, Wei Yang, Jimmy Lin, and Kyunghyun Cho. 2019b. Document expansion by query prediction. *arXiv preprint arXiv:1904.08375*.

Liangming Pan, Wenqiang Lei, Tat-Seng Chua, and Min-Yen Kan. 2019. Recent advances in neural question generation. *arXiv preprint arXiv:1905.08949*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

Vasile Rus, Brendan Wyse, Paul Piwek, Mihai Lintean, Svetlana Stoyanchev, and Cristian Moldovan. 2010. The first question generation shared task evaluation challenge. In *Proceedings of the Sixth International Natural Language Generation Conference (INLG 2010)*, pages 105–114.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Yansen Wang, Chenyi Liu, Minlie Huang, and Liqiang Nie. 2018a. Learning to ask questions in open-domain conversational systems with typed decoders. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2193–2203, Melbourne, Australia. Association for Computational Linguistics.

Zichao Wang, Andrew S Lan, Weili Nie, Andrew E Waters, Phillip J Grimaldi, and Richard G Baraniuk. 2018b. Qg-net: a data-driven question generation model for educational content. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, pages 1–10.

Ruqing Zhang, Jiafeng Guo, Lu Chen, Yixing Fan, and Xueqi Cheng. 2021. A review on question generation from natural language text. *ACM Transactions on Information Systems (TOIS)*, 40(1):1–43.