A Feasibility Study of Answer-Agnostic Question Generation for Education

Liam Dugan, Eleni Miltsakaki, Shriyash Upadhyay, Etan Ginsberg,
Hannah Gonzalez, Dayheon Choi, Chuning Yuan, Chris Callison-Burch
University of Pennsylvania
{ldugan,elenimi,shriyash,etangins,hannahgl,dhachoi,dianacny,ccb}@seas.upenn.edu

Abstract

We conduct a feasibility study into the applicability of *answer-agnostic* question generation models to textbook passages. We show that a significant portion of errors in such systems arise from asking irrelevant or uninterpretable questions and that such errors can be ameliorated by providing summarized input. We find that giving these models human-written summaries instead of the original text results in a significant increase in acceptability of generated questions (33% → 83%) as determined by expert annotators. We also find that, in the absence of human-written summaries, automatic summarization can serve as a good middle ground.

1 Introduction

Writing good questions that target salient concepts is difficult and time consuming. Automatic Question Generation (QG) is a powerful tool that could be used to significantly lessen the amount of time it takes to write such questions. A QG system that automatically generates relevant questions from textbooks would help professors write quizzes faster and help students stay engaged when reviewing course material.

Previous work on QG has focused primarily on answer-aware QG models. These models require the explicit selection of an answer span in the input context, typically through the usage of highlight tokens. This adds significant overhead to the question generation process and is undesirable in cases where clear lists of salient key terms are unavailable. We conduct a feasibility study[1] on the application of *answer-agnostic* question generation models (ones which do not require manual selection of answer spans) to an educational context. Our contributions are as follows:

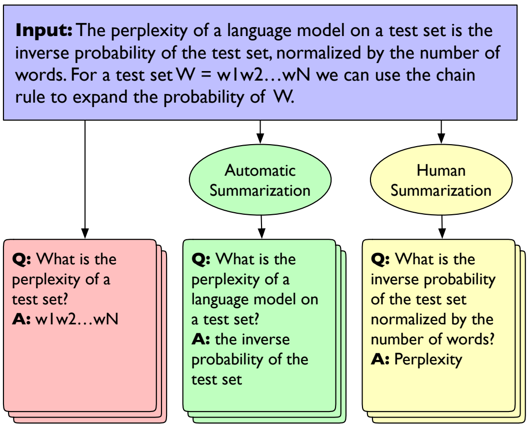
Input: The perplexity of a language model on a test set is the inverse probability of the test set, normalized by the number of words. For a test set W = w1w2...wN we can use the chain rule to expand the probability of W.
Automatic Summarization
Human Summarization
Q: What is the perplexity of a test set? **A:** w1w2...wN
Q: What is the perplexity of a language model on a test set? **A:** the inverse probability of the test set
Q: What is the inverse probability of the test set normalized by the number of words? **A:** Perplexity

Figure 1: Relevance, interpretability, and acceptability of generated questions are significantly improved when using human-written summaries (yellow) or automatically-generated summaries (green) as input instead of the original text (red).

- We show that the primary way answer-agnostic QG models fail is by generating irrelevant or uninterpretable questions.
- We show that giving answer-agnostic QG models human-written summaries instead of the original text results in significant increases in question acceptability (33% → 83%), relevance (61% → 95%), and in-context interpretability (56% → 94%).
- We show that, in absence of human-written summaries, providing automatically generated summaries as input is a good alternative.

2 Related Work & Background

Early attempts to use QG for educational applications involved generating gap-fill or "cloze" questions[2] (Taylor, 1953) from textbooks (Agarwal and Mannem, 2011). This procedure has been shown to be effective in classroom settings (Zavala and Mendoza, 2018) and students' scores on this style

[1] The data collected and software used is available at https://github.com/liamdugan/summary-qg

[2] For example, Q: "Dynamic Programming was introduced in ____" A: 1957

Findings of the Association for Computational Linguistics: ACL 2022, pages 1919 - 1926
May 22-27, 2022 ©2022 Association for Computational Linguistics

of generated question correlate positively with their scores on human-written questions (Guo et al., 2016). However, there are many situations where gap-fill questions are not effective, as they are only able to ask about specific unambiguous key terms.

In recent years, with the advent of large crowd-sourced datasets for extractive question answering (QA) such as SQuAD (Rajpurkar et al., 2018), neural models have become the primary methods of choice for generating traditional interrogative style questions (Kurdi et al., 2019). A common task formulation for neural QG is to phrase the task as *answer-aware*, that is, given a context passage $C = \{c_0, ..., c_n\}$ and an answer span within this context $A = \{c_k, ..., c_{k+l}\}$, train a model to maximize $P(Q|A, C)$ where $Q = \{q_0, ..., q_m\}$ are the tokens in the question. These models are typically evaluated using n-gram overlap metrics such as BLEU/ROUGE/METEOR (Papineni et al., 2002; Lin, 2004; Banerjee and Lavie, 2005) with the reference being the original human-authored question as provided by the extractive QA dataset.

The feasibility of using *answer-aware* neural QG in an educational setting was investigated by Wang et al. (2018), who used a BiLSTM encoder (Zhang et al., 2015) to encode C and A and a unidirectional LSTM decoder to generate Q. They trained on the SQuAD dataset (Rajpurkar et al., 2018) and evaluated on textbooks from various domains (history, biology, etc.). They showed that generated questions were largely grammatical, relevant, and had high n-gram overlap with human-authored questions. However, given that we may not always have a list of key terms to use as answer spans for an input passage, there is a desire to move past *answer-aware* QG models and evaluate the feasibility of *answer-agnostic* models for use in education.

Shifting to answer-agnostic models creates new challenges. As Vanderwende (2008) claims, the task of deciding what is and is not important is, itself, an important task. Without manually selected answer spans to guide it, an *answer-agnostic* model must itself decide what is and is not important enough to ask a question about. This is typically done by separately modeling $P(A|C)$, i.e., which spans in the input context are most likely to be used as answer targets for questions. The extracted answer spans are then given to an answer-aware QG model $P(Q|A, C)$. This modeling choice allows for more controllable QG and more direct modeling of term salience.

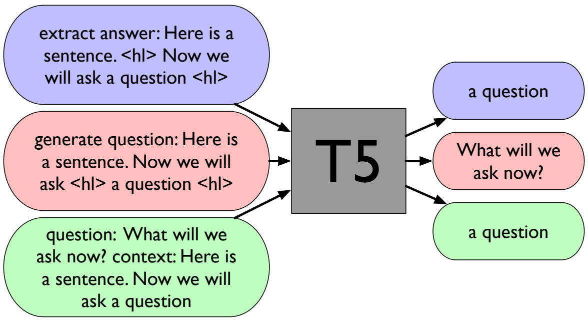
extract answer: Here is a sentence. <hl> Now we will ask a question <hl>
generate question: Here is a sentence. Now we will ask <hl> a question <hl>
question: What will we ask now? context: Here is a sentence. Now we will ask a question
T5
a question
What will we ask now?
a question

Figure 2: Diagram of the model's three different fine-tuning tasks: Answer extraction, question generation, and question answering

Previous work done by Subramanian et al. (2018) trained a BiLSTM Pointer Network (Vinyals et al., 2015) for this answer extraction task and showed that it outperformed an entity-based baseline when predicting answer spans from SQuAD passages. However, their human evaluation centered around question correctness and fluency rather than relevance of answer selection. Similar follow-up studies also fail to explicitly ask annotators whether or not the extracted answers, and subsequent generated questions, were relevant to the broader topic of the context passage (Willis et al., 2019; Cui et al., 2021; Wang et al., 2019; Du and Cardie, 2018; Alberti et al., 2019; Back et al., 2021).

In our study, we explicitly ask annotators to determine whether or not a generated question is relevant to the topic of the textbook chapter from which it is generated. In addition, we show that models trained for answer extraction on SQuAD frequently select irrelevant or ambiguous answers when applied to textbook material. We show that summaries of input passages can be used instead of the original text to aid in the modeling of topic salience and that questions generated from human-written and automatically-generated summaries are more relevant, interpretable, and acceptable.

3 Methodology

To perform answer-agnostic QG, we follow work done by Dong et al. (2019) and Bao et al. (2020) who show that language models, when fine-tuned for both QA and QG, perform better than models tuned for only one of those tasks. We assume that answer extraction will aid both QA and QG and thus use a model that was fine-tuned on all three. We considered using UniLM (Bao et al., 2020) or ProphetNet (Qi et al., 2020) but ultimately chose a T5 language model (Raffel et al., 2020) fine-tuned

	Key-Term Coverage (§5)	Total # Sents	Avg. Sent Length
A1's Summary	77.6%	279	17.56
A2's Summary	80.7%	243	19.28
A3's Summary	53.4%	148	15.37

Table 1: Analysis of summaries written by our three RAs. Key-Term Coverage is percentage of bolded textbook key terms present in the summary. Average sentence length reported in tokens (space-delimited).

on SQuAD due to the clean separation between tasks afforded by T5's task-specific prefixes such as "generate question:" and "extract answer:".[3]

The three fine-tuning tasks that were used to train the model we used are illustrated in Figure 2. For question generation, the model is trained to perform *answer-aware* question generation by modeling $P(Q|A,C)$. For question answering, the model is trained to perform extractive QA by modeling $P(A|C,Q)$. Finally, for answer extraction, instead of modeling $P(A|C)$, the model is trained to model $P(A|C')$ with $C' = \{c_0, ..., c_s, ..., c_e, ..., c_{n+2}\}$ where c_s and c_e are highlight tokens that denote the start and end of the sentence within which we want to extract an answer span.

To generate questions, we iteratively highlight the start and end of each sentence in a given passage and extract at most one answer span per sentence.[4] We then generate one question per extracted answer span using the same model in an answer-aware fashion. Passages longer than 512 tokens are split such that no sentences are divided between sub-passages and all sub-passages have a roughly equal number of sentences.

4 Experiments

Our first experiment evaluates the performance of the model on the original text extracted from Jurafsky and Martin (2020)'s textbook "Speech and Language Processing 3rd Edition."[5] To ensure proper comparison, we manually extracted the text from our three chapters of interest (Chapters 2, 3, and 4). When extracting text, all figures, tables, and equations were omitted and all references to them were either replaced with appropriate parenthetical citations or removed when possible. In total, we generated 1208 question-answer pairs from the original text.

Our second experiment evaluates the performance of the model on human-written summaries. We recruited three research assistants (RAs) as part of an undergraduate research experience to write abstractive summaries for each subsection of the same three chapters of the textbook.[6] They were encouraged to make their summaries easily readable by humans rather than to be easily understandable by machines but otherwise no specific guidelines were given. We report some statistics about these summaries in Table 1 and include examples in Appendix E. From these three sets of summaries we generated a total of 667 question-answer pairs.

Our final experiment evaluates the performance of the model on automatically generated summaries. To perform this automatic summarization we used a BART (Lewis et al., 2020) language model which was fine-tuned for summarization on the CNN/DailyMail dataset (Nallapati et al., 2016).[7] The same chunking procedure as described in Section 3 was performed on input passages that were larger than 512 tokens. The summarized output sub-passages were then concatenated together before running question generation. In total, we generated 318 question-answer pairs from our automatic summaries.

5 Evaluation

For evaluation, we randomly sampled 100 question-answer pairs from each of the three experiments to construct our evaluation set of 300 questions. We tasked the same set of RAs to evaluate the quality of the question-answer pairs. All 300 pairs were given to all three annotators. We asked the following yes/no questions:

(i). (Acceptable) Would you directly use this question as a flashcard?

(ii). (Grammatical) Is this question grammatical?

(iii). (Interpretable) Does the question make sense out of context?

(iv). (Relevant) Is this question relevant?

(v). (Correct) Is the answer correct?

We provided many example annotations to our annotators and wrote clear guidelines about each cat-

[3] https://huggingface.co/valhalla/t5-base-qa-qg-hl

[4] This comes from a limitation of the answer extraction model. The model is highly likely to extract the same answer span when run on a sentence multiple times. Future work should seek to improve this weakness. There are many cases where asking multiple questions on one sentence is desirable.

[5] https://web.stanford.edu/ jurafsky/slp3/

[6] RAs were compensated with inclusion as co-authors

[7] https://huggingface.co/facebook/bart-large-cnn

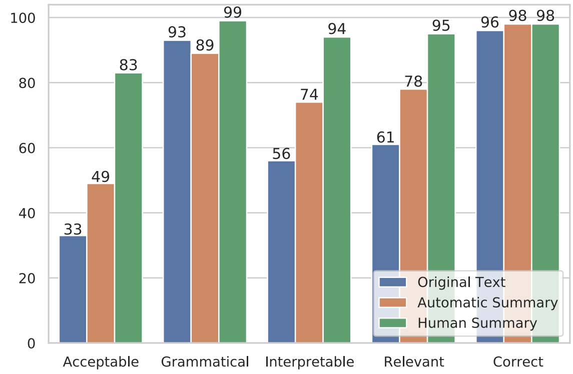

Figure 3: Results of our human evaluation for each input method. Numbers represent the proportion of questions that were labeled as having the given attribute (as determined by majority vote among our three annotators).

Source	n	Qs	As	Qs or As
Original Text	1209	70.9%	70.3%	88.6%
Auto Summary	318	44.9%	43.0%	60.1%
Human Summary	667	63.9%	68.4%	86.1%

Table 2: Coverage of bolded key terms from the textbook. Numbers represent percentage of bolded key terms present in any of the n question/answer pairs selected from the given source.

egory to ensure high agreement. Our full annotator guidelines can be found in Appendix B.

In Figure 3 we report the results of our evaluation across the three sources. We note that a majority of observed errors in the original text questions stem from them being either irrelevant or uninterpretable out of context. We also see that generating questions directly from human-written summaries significantly improves relevance and interpretability, resulting in over 80% being labeled as acceptable by annotators. Finally, in the case of automatic summaries, we see that relevance and interpretability are improved as compared to the original text questions while grammaticality suffers.

In Table 2 we evaluate the coverage of our generated questions. Coverage was calculated by extracting the bolded key terms from the textbook chapters and sub-string searching for each term among all questions and answers from a given source. Interestingly, if we think of the results from Figure 3 as precision scores and Table 2 as recall, we can see that human summaries have high precision high recall, original text has low precision high recall, and automatic summaries strike a balance between the two.

	A1	A2	A3	Pairwise IAA
Acceptable	69.7	48.7	47.7	(0.41, 0.50, 0.33)
Grammatical	98.3	90.7	86.3	(0.16, 0.49, 0.10)
Interpretable	79.7	70.7	59.7	(0.51, 0.43, 0.32)
Relevant	79.0	71.3	69.0	(0.41, 0.29, 0.25)
Correct	91.7	90.7	90.0	(0.03, 0.08, 0.06)

Table 3: Comparison between our three annotators (A1, A2, A3) on all 300 questions across all categories. Numbers represent percentages of "Yes" answers. Pairwise Inter-Annotator Agreement is calculated by Cohen κ and is reported in the order (A1-A2, A2-A3, A3-A1).

In Table 3 we report the pairwise inter-annotator agreement (IAA) as well as a per-annotator scoring breakdown. We use pairwise Cohen κ instead of Fleiss κ to better highlight the difference in agreement between certain pairs of annotators.[8]. While at first glance it may seem that agreement is low for grammaticality and correctness, this is somewhat expected for highly unbalanced classes (Artstein and Poesio, 2008). For the other three categories (relevance, interpretability, acceptability) we see pairwise agreement of approximately 0.4, suggesting a fair degree of agreement for such seemingly ambiguous categories.

6 Conclusion and Future Work

In this work we show that answer-agnostic QG models have difficulty both choosing relevant topics to ask about and generating questions that are interpretable out of context. We show that asking questions on summarized text ameliorates this in large part and that these gains can be approximated by the use of automatic summarization.

Future work should seek to further explore the relationship between summarization and QG. Work done concurrently to ours by Lyu et al. (2021) already has promising results in this direction, showing that training a QG model on synthetic data from summarized text improves performance on downstream QA.

Additionally, future work should focus on further refining and standardizing the metrics used for both automatic and human evaluation of QG. As noted by Nema and Khapra (2018) n-gram overlap metrics correlate poorly with in-context interpretability and evaluation on downstream QA fails to address the relevance of generated questions.

[8] Examples of questions for each category on which there was significant disagreement are listed in Appendix D

Acknowledgements

We graciously thank Suraj Patil for providing the fine-tuned question generation model used in this project. His training and inference code provided a great starting point for our experiments. We're very grateful for his support.

We would also like to thank Prof. Dan Jurafsky and Prof. James Martin for providing us with the raw latex files for their textbook. These files were very helpful for extraction purposes and saved us a lot of time.

Finally, we would like to thank the members of our lab for suggestions and feedback. In particular, Dan Deutsch and Alyssa Hwang were particularly influential in shaping the current version of this paper. Their great suggestions made the writing much clearer and much more understandable.

References

Manish Agarwal and Prashanth Mannem. 2011. Automatic gap-fill question generation from text books. In *Proceedings of the Sixth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 56–64, Portland, Oregon. Association for Computational Linguistics.

Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic QA corpora generation with roundtrip consistency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6168–6173, Florence, Italy. Association for Computational Linguistics.

Ron Artstein and Massimo Poesio. 2008. Survey article: Inter-coder agreement for computational linguistics. *Computational Linguistics*, 34(4):555–596.

Seohyun Back, Akhil Kedia, Sai Chetan Chinthakindi, Haejun Lee, and Jaegul Choo. 2021. Learning to generate questions by learning to recover answer-containing sentences. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1516–1529, Online. Association for Computational Linguistics.

Satanjeev Banerjee and Alon Lavie. 2005. 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*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Songhao Piao, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2020. Unilmv2: Pseudo-masked language models for unified language model pre-training. In *ICML*.

Shaobo Cui, Xintong Bao, Xinxing Zu, Yangyang Guo, Zhongzhou Zhao, Ji Zhang, and Haiqing Chen. 2021. Onestop QAmaker: Extract question-answer pairs from text in a one-stop approach. *ArXiv*, abs/2102.12128.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. *Advances in Neural Information Processing Systems*, 32.

Xinya Du and Claire Cardie. 2018. Harvesting paragraph-level question-answer pairs from Wikipedia. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1907–1917, Melbourne, Australia. Association for Computational Linguistics.

Qi Guo, Chinmay Kulkarni, Aniket Kittur, Jeffrey P. Bigham, and Emma Brunskill. 2016. Questimator: Generating knowledge assessments for arbitrary topics. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, IJCAI'16, page 3726–3732. AAAI Press.

Daniel Jurafsky and James H Martin. 2020. *Speech and language processing 3rd edition draft*. Prentice Hall NJ.

Ghader Kurdi, Jared Leo, Bijan Parsia, and Salam Al-Emari. 2019. A systematic review of automatic question generation for educational purposes. *International Journal of Artificial Intelligence in Education*, 30.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Chenyang Lyu, Lifeng Shang, Yvette Graham, Jennifer Foster, Xin Jiang, and Qun Liu. 2021. Improving unsupervised question answering via summarization-informed question generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4134–4148, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Çağlar Gulçehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence RNNs and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 280–290, Berlin, Germany. Association for Computational Linguistics.

Preksha Nema and Mitesh M. Khapra. 2018. Towards a better metric for evaluating question generation systems. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3950–3959, Brussels, Belgium. Association for Computational Linguistics.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, Pennsylvania, USA. Association for Computational Linguistics.

Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 2401–2410.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. *arXiv preprint arXiv:1806.03822*.

Sandeep Subramanian, Tong Wang, Xingdi Yuan, Saizheng Zhang, Adam Trischler, and Yoshua Bengio. 2018. Neural models for key phrase extraction and question generation. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 78–88, Melbourne, Australia. Association for Computational Linguistics.

Wilson L. Taylor. 1953. "Cloze procedure": A new tool for measuring readability. *Journalism Quarterly*, 30(4):415–433.

Lucy Vanderwende. 2008. The importance of being important: Question generation. In *Proceedings of the 1st Workshop on the Question Generation Shared Task Evaluation Challenge, Arlington, VA*.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Siyuan Wang, Zhongyu Wei, Zhihao Fan, Yang Liu, and Xuanjing Huang. 2019. A multi-agent communication framework for question-worthy phrase extraction and question generation. In *AAAI*.

Zichao Wang, Andrew S. Lan, Weili Nie, Andrew E. Waters, Phillip J. Grimaldi, and Richard G. Baraniuk. 2018. Qg-net: A data-driven question generation model for educational content. In *Proceedings of the Fifth Annual ACM Conference on Learning at Scale*, L@S '18, New York, NY, USA. Association for Computing Machinery.

Angelica Willis, Glenn M. Davis, Sherry Ruan, Lakshmi Manoharan, James A. Landay, and Emma Brunskill. 2019. Key phrase extraction for generating educational question-answer pairs. *Proceedings of the Sixth (2019) ACM Conference on Learning @ Scale*.

Laura Zavala and Benito Mendoza. 2018. On the use of semantic-based aig to automatically generate programming exercises. In *Proceedings of the 49th ACM Technical Symposium on Computer Science Education*, SIGCSE '18, page 14–19, New York, NY, USA. Association for Computing Machinery.

Shu Zhang, Dequan Zheng, Xinchen Hu, and Ming Yang. 2015. Bidirectional long short-term memory networks for relation classification. In *Proceedings of the 29th Pacific Asia Conference on Language, Information and Computation*, pages 73–78, Shanghai, China.

A Software and Data

The code and data used in this project can be found in our project repository.[9] The repository houses the 300 annotated questions, the 2,194 unannotated questions, the text sources used (three chapters of cleaned text from Jurafsky and Martin, three sets of human summaries, one set of automatic summaries), and the code used to generate the questions. We also provide scripts to reproduce the coverage analysis as well as the analysis of our annotations.

B Annotator Guidelines

In Table 4, we report the annotation guidelines given to our annotators. In the original document, under each category, 3 or more example annotations were given, each containing an explanation as to why the selection was made. Categories such as grammaticality had 10 or more examples given to ensure maximum agreement between annotators. Several discussion sessions were held between the authors and annotators to ensure that the guidelines were well understood.

During annotation, annotators were given the original textbook chapters to use as reference material and were allowed to use online search engines to check for grammaticality and correctness.

[9] https://github.com/liamdugan/summary-qg

Would you directly use this question as a flashcard? (Yes / No): A Yes answer to this question means that the generated question is salient, grammatically correct, non-awkwardly phrased and has one correct answer. If you answer Yes to this question you may skip the rest of the annotation for the given example – the answers for all other questions are assumed to be Yes. If you answer No, then please continue on to the rest of the questions. Importantly, if you *did* answer yes to all of the other questions, do not feel pressured to answer yes to this question. There are many reasons why you might not want to directly use a question as a flashcard (too easy, too general, etc.) that are not enumerated here.
Is this question grammatically correct? (Yes / No): A Yes answer to this question implies that a question has no grammatical errors. Awkwardly worded questions that are grammatical should be annotated as such (answer Yes for these questions).
Does this question make sense out of context? (Yes / No): This question asks if there are any references made by the question to other items that have been "previously discussed". For our use case, questions should never refer to other specific items in the text from which they were drawn. A Yes answer to this implies that the question is interpretable when taken on its own and is a question that someone would ask if there was no pre-existing context.
Is this question relevant? (Yes / No): A Yes answer to this question implies that the question being asked is important for understanding the main points that the chapter (and by extension the book) is attempting to teach. Questions that are relevant should be ones that would plausibly be asked on a quiz or a test from a fairly thorough course on computational linguistics. Questions that are about insignificant details or questions that are about specific illustrated examples that are not useful for understanding the main points of the chapter should be given a No. Anything that is relevant (or tangentially relevant) to computational linguistics should be given a Yes.
Is the answer to the question correct? (Yes / No): A Yes answer to this question implies that the answer given is one of a multitude of plausible correct answers to the question. If the question has multiple correct answers and the given answer is one of them, it should be annotated as a Yes. If the question is bad/ungrammatical or underspecified to such an extent that you cannot judge the answer properly, you should annotate Yes. However, irrelevant questions that are grammatical and reasonably interpretable should be annotated properly.

Table 4: Guidelines given to our human annotators before annotating for the acceptability, grammaticality, interpretability, relevance, and correctness of generated questions.

# Questions	Chapter 2 ($n = 139$)	Chapter 3 ($n = 93$)	Chapter 4 ($n = 66$)
Acceptable	54.0%	58.1%	53.0%
Grammatical	94.2%	93.5%	93.9%
Interpretable	74.1%	76.3%	72.7%
Relevant	72.7%	81.7%	83.3%
Correct	95.0%	100%	98.5%

Table 5: Distribution of human evaluation scores across the three chapters of annotation. Labels are determined via majority vote among our three annotators.

C Comparison Across Chapters

In Table 5 we report the distribution of scores across chapters. We note that scores are largely consistent across the three chapters, with lower average relevance for Chapter 2 questions possibly owing to the source material containing many worked examples of regular expressions.

D Example Disagreements

In Table 6, we list questions for which there was at least one dissenting annotator. We see that for categories such as "Relevant" and "Interpretable", annotations are often dependent on the level of granularity with which the topic is being discussed. For example, a question such as "Who named the minimum edit distance algorithm?" may or may not be relevant depending on how granular of a class the student is taking.

For categories such as "Correct" or "Acceptable" certain particularities about otherwise good questions can easily disqualify them from receiving a positive annotation. In the case of "What NLP algorithms require algorithms for word segmentation?", keen-eyed annotators would notice that the question is non-sensical, however others may note that both Japanese and Thai do, in fact, require word segmentation. Particularities such as these make this task very difficult, even for expert annotators.

E Example Summaries

In Table 7 we list two examples of textbook sections with their accompanying human and automatic summaries. We see that length of summary varies drastically between our annotators, each of them making different decisions on whether or not to keep or discard certain pieces of information. We also note that automatic summaries are much more extractive in nature while human summaries are generally more abstractive.

Acceptable	**Q**: What is another name for a corpus that NLP algorithms learn from? **A**: training corpus **Q**: What would happen if we accidentally trained the model on the test set? **A**: bias **Q**: What would give a lower cross-entropy? **A**: The more accurate model
Grammatical	**Q**: What are words like uh and um called fillers? **A**: filled pauses **Q**: What context do words that are in our vocabulary appear in a test set in? **A**: unseen **Q**: What word has the same lemma cat but are different wordforms? **A**: cats
Interpretable	**Q**: What gives us a way to quantify both of these intuitions about string similarity? **A**: Edit distance **Q**: What is another important step in text processing? **A**: Sentence segmentation **Q**: What seems to matter more than its frequency? **A**: whether a word occurs or not
Relevant	**Q**: What isn't big enough to give us good estimates in most cases? **A**: web **Q**: Who named the minimum edit distance algorithm? **A**: Wagner and Fischer **Q**: What do algorithms have to deal with? **A**: ambiguities
Correct	**Q**: What do square brackets not allow us to say? **A**: s or nothing **Q**: What NLP algorithms require algorithms for word segmentation? **A**: Japanese and Thai **Q**: What encode some facts that we think of as strictly syntactic in nature? **A**: Bigram probabilities

Table 6: Questions for which there was disagreement on the label for the given category

Original Text: What do we do with words that are in our vocabulary (they are not unknown words) but appear in a test set in an unseen context (for example they appear after a word they never appeared after in training)? To keep a language model from assigning zero probability to these unseen events, we'll have to shave off a bit of probability mass from some more frequent events and give it to the events we've never seen. This modification is called smoothing or discounting. In this section and the following ones we'll introduce a variety of ways to do smoothing: Laplace (add-one) smoothing, add-k smoothing, stupid backoff, and Kneser-Ney smoothing.	**Original Text**: As we saw in the previous section, naive Bayes classifiers can use any sort of feature: dictionaries, URLs, email addresses, network features, phrases, and so on. But if, as in the previous section, we use only individual word features, and we use all of the words in the text (not a subset), then naive Bayes has an important similarity to language modeling. Specifically, a naive Bayes model can be viewed as a set of class-specific unigram language models, in which the model for each class instantiates a unigram language model. Since the likelihood features from the naive Bayes model assign a probability to each word P(word|c), the model also assigns a probability to each sentence.
Automatic Summary: What do we do with words that are in our vocabulary (they are not unknown words) but appear in a test set in an unseen context? To keep a language model from assigning zero probability to these unseen events, we'll have to shave off a bit of probability mass from some more frequent events. This modification is called smoothing or discounting.	**Automatic Summary**: A naive Bayes Bayes model can be viewed as a set of class-specific unigram language models. The model for each class instantiates a language model. Since the likelihood features assign a probability to each word P(word|c), the model also assigns a probability to each sentence.
Human Summary (A1): We remove some probability mass for more frequent events and reassign it to unseen events with known words, and this is called smoothing or discounting. We study four 4 main methods of smoothing: Laplace smoothing, add-k smoothing, stupid backoff, and Kneser-Ney smoothing.	**Human Summary (A1)**: A naïve Bayes model can be viewed as a set of class-specific unigram language models.
Human Summary (A2): Smoothing or discounting is the procedure of transferring the probability mass of frequent events to other words that appear in the test set in an unseen context.	**Human Summary (A2)**: Naive Bayes models are similar to language modeling in that they can be viewed as a set of class-specific unigram language models. The probability of a sentence being positive is the total product of the individual probabilities that each word in the sentence is positive.
Human Summary (A3): Not assigning zero to the probability of an unseen word in the test set is called smoothing or discounting. There are different ways to do smoothing: Laplace, add-k smoothing, stupid backoff, Kneser-Ney smoothing.	**Human Summary (A3)**: A naive Bayes model can be viewed as a set of class-specific unigram language models, in which the model for each class instantiates a unigram language model.

Table 7: Examples of human and automatic summaries for two sections of "Speech and Language Processing". The left text is from Section 3.4 "Smoothing" and the right text is from Section 4.6 "Naive Bayes as a Language Model". We see that the automatic summaries tend to be more extractive while the human summaries are more abstractive.