An Empirical Comparison of LM-based Question and Answer Generation Methods

Asahi Ushio and **Fernando Alva-Manchego** and **Jose Camacho-Collados**

Cardiff NLP, School of Computer Science and Informatics, Cardiff University, UK

{UshioA,AlvaManchegoF,CamachoColladosJ}@cardiff.ac.uk

Abstract

Question and answer generation (QAG) consists of generating a set of question-answer pairs given a context (e.g. a paragraph). This task has a variety of applications, such as data augmentation for question answering (QA) models, information retrieval and education. In this paper, we establish baselines with three different QAG methodologies that leverage sequence-to-sequence language model (LM) fine-tuning. Experiments show that an end-to-end QAG model, which is computationally light at both training and inference times, is generally robust and outperforms other more convoluted approaches. However, there are differences depending on the underlying generative LM. Finally, our analysis shows that QA models fine-tuned solely on generated question-answer pairs can be competitive when compared to supervised QA models trained on human-labeled data.

1 Introduction

Question and answer generation (QAG) is the task of generating a set of question-answer pairs given an input context such as a document, a paragraph or a sentence. QAG can be applied to develop question answering (QA) models without human supervision (Lewis et al., 2019; Zhang and Bansal, 2019; Puri et al., 2020) and as a data augmentation mean for QA model understanding (Shakeri et al., 2020; Bartolo et al., 2021). Moreover, QAG is used as an aid of educational systems (Heilman and Smith, 2010; Lindberg et al., 2013), to improve information retrieval models (Pyatkin et al., 2021; Lewis et al., 2021), and as a tool for model interpretation (Perez et al., 2020; Lee et al., 2020).

QAG stems from question generation (QG) (Mitkov and Ha, 2003; Du et al., 2017; Zhou et al., 2017; Du and Cardie, 2018), which consists of generating a question given an answer on the input context. Despite QG being widely studied in the language model era (Murakhovs'ka et al., 2022;
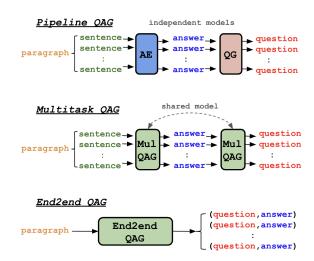

Figure 1: Overview of the considered QAG approaches.

Ushio et al., 2022), QAG is a more complex task, since the answer needs to be generated and not assumed to be part of the input. Therefore, it is unclear what types of QAG models work in practice as no comprehensive comparisons have been established so far.

In this paper, we formalize QAG as a task that generates question-answer pairs given a context, and compare three simple QAG strategies based on fine-tuning encoder-decoder language models (LMs) such as T5 (Raffel et al., 2020) and BART (Lewis et al., 2020). Our three proposed approaches (illustrated in Figure 1) consist of: (1) pipeline QAG, which decomposes the task into answer extraction and question generation, learning a separate model for each subtask; (2) multitask QAG, which uses a shared single model to train both subtasks instead of independent ones; and (3) end2end QAG, which uses end-to-end sequence-to-sequence learning to generate question-answer pairs directly. Finally, we compare these three approaches on a multi-domain QA-based evaluation, where QA models are trained with the question-answer pairs that each QAG model generates. All

the QAG models are publicly released via Hug-gingFace (Wolf et al., 2020)[1], and available on the online demo[2].

2 Related Work

There are a few works that leverage pre-trained LMs for QAG. For example, Alberti et al. (2019) first fine-tuned BERT (Devlin et al., 2019) on answer extraction and QG, and generate question-answer pairs by extracting an answer, on which the associated question is generated. Puri et al. (2020) followed a similar idea by fine-tuning an autoregressive LM for QG. In contrast, Shakeri et al. (2020) fine-tuned a single LM on answer extraction and QG jointly. Lee et al. (2020) trained an LSTM sequence-to-sequence model from scratch to generate question and answer sequentially. More recently, Bartolo et al. (2021) used a QAG model to generate adversarial examples for QA. Similarly, Lewis et al. (2021) improved on extractive QA by generating millions of question-answer pairs via QAG. In these two last cases, the model to fine-tune was BART (Lewis et al., 2020).

While all these studies use the three methods that we analyse in this paper (i.e. pipeline, multi-task and end2end), these are not easily comparable, as there are important differences among them in terms of settings, dataset, input to the LMs, and evaluation metrics. Moreover, except for Lewis et al. (2021), none of the proposed QAG models have been made publicly available. Finally, the two most recent studies using BART (Bartolo et al., 2021; Lewis et al., 2021) have not performed any evaluation on the QAG model, as it is included as a part of a larger pipeline. We summarize the comparison of these prior works and our evaluation at Table 1.

3 Question & Answer Pair Generation

Given an input context c (e.g. a paragraph), QAG aims to generate natural question-answer pairs $\mathcal{Q}_c$ related to the information in c: $\mathcal{Q}_c = \{(q^1, a^1), (q^2, a^2), \dots\}$. In what follows we describe three different approaches for QAG based on fine-tuning language models.

	Pipe.	Multi.	E2E	Open	Eval.
Alberti et al. (2019)	✓	✗	✗	✗	✓
Puri et al. (2020)	✓	✗	✗	✗	✓
Lee et al. (2020)	✗	✓	✗	✗	✓
Shakeri et al. (2020)	✓	✗	✓	✗	✓
Bartolo et al. (2021)	✓	✗	✗	✗	✗
Lewis et al. (2021)	✓	✗	✗	✓	✗
Ours	✓	✓	✓	✓	✓

Table 1: Comparison of our paper and previous studies involving LM-based QAG. The first three columns include the QAG methods used in the corresponding paper: pipeline (Pipe.), multitask (Multi.), and end-to-end (E2E). The fourth column indicates whether QAG models were released open-source (Open). Finally, the last column refers to whether the paper includes QAG evaluation (Eval.).

3.1 Pipeline QAG

The QAG task can be decomposed into two simpler subtasks, answer extraction (AE) and QG, where the AE model P_{ae} first generates an answer candidate $\tilde{a}$ on a sentence s in context c, and then the QG model P_{qg} generates a question $\tilde{q}$ that is answerable by answer $\tilde{a}$ given context c. The AE and QG models can be trained independently on any paragraph-level QG datasets that consist of quadruples (c, s, a, q) by maximizing the conditional log-likelihood of:

$$\tilde{a} = \arg\max_a P_{ae}(a|c, s) \quad (1)$$

$$\tilde{q} = \arg\max_q P_{qg}(q|c, s, a) \quad (2)$$

where the log-likelihood is factorized into token-level predictions, similar to other sequence-to-sequence learning settings (Sutskever et al., 2014). In practice, the input to the AE model takes the form of:

$$[c_1, \dots, \texttt{<hl>}, s_1, \dots, s_{|s|}, \texttt{<hl>}, \dots, c_{|c|}]$$

where s_i and c_i are the $i-$th token of s and c respectively, $|\cdot|$ represents the number of tokens in a text, and <hl> is the highlighted token to mark the sentence in the context, following the QG formulation of Chan and Fan (2019) and Ushio et al. (2022). Likewise, the input to the QG model takes the answer into account by:

$$[c_1, \dots, \texttt{<hl>}, a_1, \dots, a_{|a|}, \texttt{<hl>}, \dots, c_{|c|}]$$

where a_i is the $i-$th token of a. At inference time, we simply replace the gold answer a of the QG

model (2) by the prediction from the AE model (1), and run the inference over all the sentences in context c to obtain question-answer pairs. Consequently, the pipeline approach can generate, at most, as many pairs as sentences in c.

3.2 Multitask QAG

Instead of training independent models for each subtask, a shared model can be fine-tuned on both AE and QG jointly in a multitask learning manner. To be precise, we mix the training instances for AE and QG altogether, and randomly sample a batch at each iteration of fine-tuning. Each subtask is distinguished by a task prefix added at the beginning of the input text: "extract answer" (AE) and "generate question" (QG).

3.3 End2end QAG

Instead of breaking down QAG into two separate components, we can directly model it by converting the question-answer pairs into a flattened sentence y, and fine-tuning a sequence-to-sequence model to generate y from c. Let us define a function that maps $\mathcal{Q}_c$ to a sentence as:

$$\mathcal{T}(\mathcal{Q}_c) = \text{``}\{t(q^1, a^1)\} \mid \{t(q^2, a^2)\} \mid \ldots \text{''} \quad (3)$$

$$t(q, a) = \text{``question:}\{q\}, \text{answer:}\{a\}\text{''} \quad (4)$$

where each pair is textualized with the template (4) and joined by a separator |. The end2end QAG model P_{qag} is then optimized by maximizing the following conditional log-likelihood:

$$\tilde{y} = \arg\max_y P_{\text{qag}}(y|c) \quad (5)$$

4 Evaluation

4.1 Experimental Setting

Data. QAG models are trained on SQuAD (Rajpurkar et al., 2016). As their outputs consist of arbitrary questions and answers, reference-based NLG evaluation metrics traditionally used in QG research (Papineni et al., 2002; Denkowski and Lavie, 2014; Lin, 2004; Mohammadshahi et al., 2022) are unsuitable. As such, we conduct an extrinsic evaluation by training QA models on the data generated by the QAG models. For this, we rely on SQuADShifts (Miller et al., 2020), an English reading comprehension dataset in four domains (Amazon/Wikipedia/News/Reddit). For both SQuAD and SQuADShifts, we rely on the train/validation/test splits provided in QG-Bench (Ushio et al., 2022).

Multi-domain QA Evaluation. Given a QAG model to be assessed, we first generate question-answer pairs on each domain of SQuADShifts, and fine-tune DistilBERT (Sanh et al., 2019) on the generated pseudo QA pairs, where F_1 and exact match on the test set are considered as the target metric. This SQuADShifts QA-based evaluation can be used to probe the robustness of the model across domains, as well as for the overall performance by averaging metrics over the domains. Our QA evaluation relies on Tune,[3] an efficient grid search engine for parameter optimization, to find optimal hyperparameters during QA model fine-tuning.

Base Models. For all comparison systems (i.e. pipeline, multitask and end2end), we experiment with T5 (Raffel et al., 2020) and BART (Lewis et al., 2020) as base LMs, with the model weights t5-{small,base,large} and facebook/bart-{base,large} shared on HuggingFace.[4] Moreover, we report the results of a QG model that takes the gold answers from the provided QA training set as input (QG-only). This is similar to the pipeline method but excluding the AE component.

4.2 Results

Table 2 shows the SQuADShifts QA evaluation results for the three approaches considered. Interestingly, the top-2 best models, BART$_{\text{LARGE}}$ (multitask) and T5$_{\text{LARGE}}$ (end2end), outperform *Gold QA* (i.e., the model using the human-labeled gold annotations) in two out of four domains, as well as the average in both F_1 and exact match. Even smaller models such as T5$_{\text{SMALL}}$ are competitive with respect to using the gold standard question-answer pairs.

Given the results, it is unclear which approach provides the best performance, as BART$_{\text{LARGE}}$ (multitask) achieves the best average F_1 score (including the best results on Amazon and Reddit domains in both metrics), while T5$_{\text{LARGE}}$ (end2end) obtains the best average exact match (as well as the best results on Wiki and NYT domains in both metrics). Among the QAG approaches, T5 consistently works better with the end2end QAG, while BART is not well-suited when used end2end. A possible explanation is that T5 has observed sentences with structured information due to its multitask pre-training objective, while BART did not have

---

[3]https://docs.ray.io/en/latest/tune/index.html
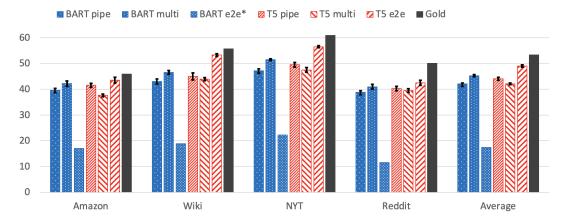[4]See Appendix A for details on the procedure to find optimal hyperparameters during model fine-tuning.

Figure 2: Downsampled (equal-sized) SQuADShifts QA evaluation results (F_1 score with 95% confidence interval) for T5$_{\text{LARGE}}$ multitask/pipeline/end2end and BART$_{\text{LARGE}}$ pipeline, compared with the original result of each model and the gold QA dataset.

such training instances as it was trained only on a denoising sequence-to-sequence objective.

4.3 Generation Size Analysis

In the SQuADShifts QA evaluation, the number of question-answer pairs generated by QAG models is often larger than the human-labelled gold dataset in each domain, as shown in Table 3.[5] Therefore, to fairly compare the quality of generated question-answer pairs, we randomly downsampled the number of the generated question-answer pairs to match the size of the gold dataset. For this analysis we focus on the best-performing T5$_{\text{LARGE}}$ and BART$_{\text{LARGE}}$ QAG models[6], and run the same SQuADShifts QA evaluation with the downsampled pairs. Figure 2 shows the average of F_1 scores over 10 independent trials with different random seeds at downsampling.[7] In this experiment, no model outperforms the gold QA baseline. This indicates that the human-annotated gold dataset is still more informative and data efficient than the generated question-answer pairs. Also, since the pipeline/multitask QAG models generate more pairs than the end2end model, downsampling has a larger effect on the pipeline and multitask models than the end2end model. This means that the T5$_{\text{LARGE}}$ (end2end) model can generate question-answer pairs of higher quality than those generated by BART$_{\text{LARGE}}$ (multitask), although they are equally competitive in the main experiment (§ 4.2).

---

[5]The size of generated question-answer pairs in each domain can be found in Appendix B.

[6]The end2end BART$_{\text{LARGE}}$ results match those from Table 2, since it had less data than the gold dataset.

[7]See Appendix C for the comparison of exact match.

4.4 QAG Model Comparison

So far, we have compared the three QAG approaches in terms of performance. However, performance is not the only criterion to consider when choosing a QAG model, since each approach has its own advantages and limitations in terms of computational cost and usability. From the perspective of computational complexity, end2end QAG is faster than the others at both of training and inference, because it can generate a number of question-answer pairs at once in a single paragraph pass. In contrast, both multitask and pipeline need to parse every sentence separately, and a single prediction consists of two generations (i.e. answer extraction and question generation). Essentially, the relative increase of computational cost from end2end QAG to pipeline/multitask QAG can be approximated by the average number of sentences in each paragraph. In terms of memory requirements, both multitask and end2end QAG rely on a single model, but pipeline QAG consists of two models, requiring twice as much memory storage. Finally, while computational-wise end2end is the lightest model, both pipeline and multitask approaches can generate a larger number of question-answer pairs on average, with the added benefit of being able to run the models on individual sentences. Table 4 shows a practical comparison of the three approaches.

5 Conclusion

In this paper, we formalized QAG as a task to generate pairs of questions and answers given an input context, and established baselines with three different QAG approaches. To compare them, we

Approach		Average	Amazon	Wiki	NYT	Reddit
Gold QA		*53.3/37.3*	*45.9/30.4*	*55.6/38.7*	*61.4/46.9*	*50.1/33.4*
BART_BASE	QG only	49.4/33.9	42.3/26.7	54.3/37.2	59.3/44.8	41.9/27.0
	Pipeline	50.0/32.4	48.4/29.8	49.4/31.1	53.0/36.0	**49.5/32.7**
	Multitask	**50.8/33.2**	**49.4/30.5**	**50.6/32.1**	**55.0/39.2**	48.4/31.1
	End2end	34.0/21.4	29.3/16.5	35.4/23.2	44.6/31.1	26.6/15.0
BART_LARGE	QG only	49.4/33.8	43.3/27.4	54.0/36.7	59.4/44.6	41.1/26.4
	Pipeline	51.7/34.0	49.0/30.2	52.5/33.7	55.3/40.0	49.7/32.3
	Multitask	**54.3/36.7**	**53.6/34.3**	54.1/36.4	**57.7/41.8**	**51.6/34.4**
	End2end	17.5/10.2	17.1/9.1	18.9/11.3	22.3/14.7	11.6/5.7
T5_SMALL	QG only	48.5/33.1	43.8/27.7	50.5/34.5	55.2/41.0	44.4/29.1
	Pipeline	45.4/27.4	41.6/23.8	46.9/27.2	48.9/32.1	**44.1/26.6**
	Multitask	44.0/25.2	42.1/22.9	44.3/24.1	45.6/27.9	44.0/25.7
	End2end	**48.3/31.7**	**42.3/24.8**	**54.7/37.2**	**55.2/40.1**	41.1/24.8
T5_BASE	QG only	50.7/34.7	43.3/27.4	54.4/37.1	57.7/43.2	47.3/31.1
	Pipeline	51.7/33.6	**50.6/31.1**	52.8/34.0	53.8/37.1	**49.6/32.4**
	Multitask	49.6/30.9	48.8/28.9	48.1/28.5	52.5/35.1	49.1/31.2
	End2end	**51.8/35.4**	44.9/26.9	**56.9/40.1**	59.9/45.3	45.3/29.2
T5_LARGE	QG only	48.9/33.4	42.7/26.8	53.2/36.2	58.5/43.9	41.5/26.7
	Pipeline	52.0/33.9	**50.9/31.1**	51.7/32.9	55.9/39.5	**49.7**/32.0
	Multitask	49.5/30.9	46.3/26.0	49.6/30.6	53.0/35.9	49.0/31.2
	End2end	**53.7/37.3**	49.1/**31.1**	56.1/**40.1**	**60.7/45.5**	48.8/**32.5**

Table 2: SQuADShifts QA evaluation results (F_1/exact match) of different QAG models. As an upperbound, we included the results of the same QA model trained on the gold human-annotated SQuADShifts training set (*Gold QA*). The best score among the QAG approaches within each LM is boldfaced, and the best result in each domain across all models is underlined.

Approach	Size (training / validation)
Gold QA	3,141 / 1,571
BART_LARGE (pipeline)	11,900 / 8,192
BART_LARGE (multitask)	11,752 / 8,103
BART_LARGE (end2end)	2,012 / 1,399
T5_LARGE (pipeline)	12,239 / 8,417
T5_LARGE (multitask)	12,148 / 8,357
T5_LARGE (end2end)	6,555 / 4,550

Table 3: Average number of question-answer pairs generated for SQuADShifts QA evaluation by each model over all the domains.

	Cost	Memory	Generated QA
Pipeline	$9.2x$	$2x$	$2.7x$
Multitask	$9.2x$	x	$2.7x$
End2end	x	x	x

Table 4: Comparison among the three proposed QAG approaches in terms of training cost, memory requirements, and generated question-answer pairs, using end2end as a reference. The comparison is performed for T5_LARGE with the data used for the main experiments (§ 4.1). Generated QA are averaged across the four SQuADShifts domains.

conducted a multi-domain QA based evaluation that measures the performance of a QAG model by fine-tuning QA models on the QA training dataset generated by the QAG model. Our evaluation shows that end2end QAG models that generate questions and answers simultaneously are generally the most reliable. Nonetheless, establishing a multitask paradigm with separation between answer extraction and question generation can have added benefits, especially when using LMs such as BART. In general, the results are promising, as they show that these artificially-generated QA datasets rival in quality with those annotated by humans, which could save large amount of resources.

Acknowledgements

Limitations

In this paper, we studied paragraph-level QAG models, which limits their input up to around 500 tokens, and the same approach cannot be easily applied to longer documents. Also, the answer is an entity or a phrase consisting of a few tokens and the question requires one-hop reasoning, so our models are not able for use in generating longer answers or multi-hop questions. As far as the languages are concerned, the models studies here are English only and to adapt SQuADShifts QA evaluation in other languages, we need QA datasets to train and evaluate the QAG model in those languages.

The focus on this paper was on evaluating the quality of generated question-answer pairs. As such, we do not attempt to achieve the best QA model possible, but rather use question answering as an extrinsic evaluation. This extrinsic evaluation could be further enhanced with an intrinsic manual evaluation that we did not perform in this paper. Finally, given computational constraints, our QA evaluation is based on a single model only. Again, the goal here was not to achieve the best QA performance, but we acknowledge than using different models could lead to different results.

Ethics Statement

Since pre-trained LMs are known to inherit undesirable biases and tend to generate toxic contents in some edge cases (Schick et al., 2021), the QAG

models we developed in the paper could potentially generate a question or an answer including such texts. Nevertheless, we have done internal validation on the generated question-answer pairs and we have not found such examples in the data analysed in this paper.

References

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.

Max Bartolo, Tristan Thrush, Robin Jia, Sebastian Riedel, Pontus Stenetorp, and Douwe Kiela. 2021. Improving question answering model robustness with synthetic adversarial data generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8830–8848, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Ying-Hong Chan and Yao-Chung Fan. 2019. A recurrent BERT-based model for question generation. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 154–162, Hong Kong, China. Association for Computational Linguistics.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 376–380, Baltimore, Maryland, USA. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

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.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352, Vancouver, Canada. Association for Computational Linguistics.

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

Dong Bok Lee, Seanie Lee, Woo Tae Jeong, Donghwan Kim, and Sung Ju Hwang. 2020. Generating diverse and consistent QA pairs from contexts with information-maximizing hierarchical conditional VAEs. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 208–224, Online. Association for Computational Linguistics.

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.

Patrick Lewis, Ludovic Denoyer, and Sebastian Riedel. 2019. Unsupervised question answering by cloze translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4896–4910, Florence, Italy. Association for Computational Linguistics.

Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021. PAQ: 65 million probably-asked questions and what you can do with them. *Transactions of the Association for Computational Linguistics*, 9:1098–1115.

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.

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, Sofia, Bulgaria. Association for Computational Linguistics.

John Miller, Karl Krauth, Benjamin Recht, and Ludwig Schmidt. 2020. The effect of natural distribution shift on question answering models. In *International Conference on Machine Learning*, pages 6905–6916. PMLR.

Ruslan Mitkov and Le An Ha. 2003. Computer-aided generation of multiple-choice tests. In *Proceedings*

of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing, pages 17–22.

Alireza Mohammadshahi, Thomas Scialom, Majid Yazdani, Pouya Yanki, Angela Fan, James Henderson, and Marzieh Saeidi. 2022. Rquge: Reference-free metric for evaluating question generation by answering the question.

Lidiya Murakhovs'ka, Chien-Sheng Wu, Philippe Laban, Tong Niu, Wenhao Liu, and Caiming Xiong. 2022. MixQG: Neural question generation with mixed answer types. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 1486–1497, Seattle, United States. 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.

Ethan Perez, Patrick Lewis, Wen-tau Yih, Kyunghyun Cho, and Douwe Kiela. 2020. Unsupervised question decomposition for question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8864–8880, Online. Association for Computational Linguistics.

Raul Puri, Ryan Spring, Mohammad Shoeybi, Mostofa Patwary, and Bryan Catanzaro. 2020. Training question answering models from synthetic data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5811–5826, Online. Association for Computational Linguistics.

Valentina Pyatkin, Paul Roit, Julian Michael, Yoav Goldberg, Reut Tsarfaty, and Ido Dagan. 2021. Asking it all: Generating contextualized questions for any semantic role. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 1429–1441, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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. *Journal of Machine Learning Research*, 21:1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Timo Schick, Sahana Udupa, and Hinrich Schütze. 2021. Self-diagnosis and self-debiasing: A proposal for reducing corpus-based bias in NLP. *Transactions of the Association for Computational Linguistics*, 9:1408–1424.

Siamak Shakeri, Cicero Nogueira dos Santos, Henghui Zhu, Patrick Ng, Feng Nan, Zhiguo Wang, Ramesh Nallapati, and Bing Xiang. 2020. End-to-end synthetic data generation for domain adaptation of question answering systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5445–5460, Online. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems*, 27.

Asahi Ushio, Fernando Alva-Manchego, and Jose Camacho-Collados. 2022. Generative language models for paragraph-level question generation. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Abu Dhabi, U.A.E. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Shiyue Zhang and Mohit Bansal. 2019. Addressing semantic drift in question generation for semi-supervised question answering. 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)*, pages 2495–2509, Hong Kong, China. Association for Computational Linguistics.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2017. Neural question generation from text: A preliminary study. In *National CCF Conference on Natural Language Processing and Chinese Computing*, pages 662–671. Springer.

Approach	Model	Epoch	LR	LS	Batch
Pipeline (AE)	$BART_{BASE}$	4	0.00005	0.15	64
Pipeline (QG)	$BART_{BASE}$	7	0.0001	0.15	256
Multitask	$BART_{BASE}$	3	0.00005	0.15	128
End2end	$BART_{BASE}$	2	0.00001	0.15	128
Pipeline (AE)	$BART_{LARGE}$	5	0.00005	0.15	64
Pipeline (QG)	$BART_{LARGE}$	4	0.00005	0.15	128
Multitask	$BART_{LARGE}$	6	0.00001	0.15	64
End2end	$BART_{LARGE}$	14	0.00001	0.15	64
Pipeline (AE)	$T5_{SMALL}$	7	0.0001	0.15	64
Pipeline (QG)	$T5_{SMALL}$	9	0.0001	0.15	64
Multitask	$T5_{SMALL}$	7	0.0001	0.15	64
End2end	$T5_{SMALL}$	18	0.0001	0	64
Pipeline (AE)	$T5_{BASE}$	8	0.0001	0	64
Pipeline (QG)	$T5_{BASE}$	5	0.0001	0.15	64
Multitask	$T5_{BASE}$	6	0.0001	0.15	128
End2end	$T5_{BASE}$	17	0.0001	0.15	64
Pipeline (AE)	$T5_{LARGE}$	9	0.0001	0	128
Pipeline (QG)	$T5_{LARGE}$	6	0.00005	0.15	64
Multitask	$T5_{LARGE}$	3	0.0001	0.15	64
End2end	$T5_{LARGE}$	12	0.0001	0.15	64

Table 5: Optimal hyperparameters for each QAG model.

A   Hyper Parameters

At each QAG model fine-tuning, we search the optimal hyperparameters such as learning rate via lmqg[8], a hyperparameter search tool for sequence-to-sequence LM fine-tuning, and Table 5 shows the best hyperparameters. The maximum input length is fixed as 512, and the maximum output length is 256 for the end2end QAG and 32 for the others.

B   Size of QA Pairs at SQuADShifts QA evaluation

Table 6 shows the number of question-answer pairs generated from different QAG models in each domain at SQuADShifts QA evaluation. The size of the test sets are 4,942 (Amazon), 3,696 (Wiki), 5,032 (NYT), and 4,901 (Reddit).

C   Additional Results of Downsampled SQuADShifts QA evaluation

Figure 3 shows the exact match of the downsampled SQuADShifts QA evaluation experiment.

	Approach	Size (training / validation)
Amazon	Gold QA	3,295 / 1,648
	$BART_{BASE}$ (pipeline)	14,824 / 10,273
	$BART_{LARGE}$ (pipeline)	15,204 / 10,569
	$T5_{SMALL}$ (pipeline)	15,343 / 10,643
	$T5_{BASE}$ (pipeline)	15,631 / 10,862
	$T5_{LARGE}$ (pipeline)	15,645 / 10,844
	$BART_{BASE}$ (multitask)	14,517 / 10,065
	$BART_{LARGE}$ (multitask)	15,057 / 10,452
	$T5_{SMALL}$ (multitask)	15,417 / 10,688
	$T5_{BASE}$ (multitask)	15,454 / 10,724
	$T5_{LARGE}$ (multitask)	15,479 / 10,734
	$BART_{BASE}$ (end2end)	990 / 706
	$BART_{LARGE}$ (end2end)	2,045 / 1,408
	$T5_{SMALL}$ (end2end)	6,419 / 4,470
	$T5_{BASE}$ (end2end)	7,053 / 4,889
	$T5_{LARGE}$ (end2end)	7,034 / 4,880
Wiki	Gold QA	2,646 / 1,323
	$BART_{BASE}$ (pipeline)	6,340 / 4,455
	$BART_{LARGE}$ (pipeline)	6,485 / 4,582
	$T5_{SMALL}$ (pipeline)	6,433 / 4,537
	$T5_{BASE}$ (pipeline)	6,518 / 4,597
	$T5_{LARGE}$ (pipeline)	6,518 / 4,596
	$BART_{BASE}$ (multitask)	6,267 / 4,415
	$BART_{LARGE}$ (multitask)	6,450 / 4,547
	$T5_{SMALL}$ (multitask)	6,377 / 4,504
	$T5_{BASE}$ (multitask)	6,466 / 4,564
	$T5_{LARGE}$ (multitask)	6,485 / 4,580
	$BART_{BASE}$ (end2end)	1,137 / 784
	$BART_{LARGE}$ (end2end)	1,718 / 1,214
	$T5_{SMALL}$ (end2end)	5,050 / 3,513
	$T5_{BASE}$ (end2end)	5,639 / 3,930
	$T5_{LARGE}$ (end2end)	5,515 / 3,882
NYT	Gold QA	3,355 / 1,678
	$BART_{BASE}$ (pipeline)	10,033 / 6,913
	$BART_{LARGE}$ (pipeline)	10,339 / 7,141
	$T5_{SMALL}$ (pipeline)	10,440 / 7,241
	$T5_{BASE}$ (pipeline)	10,583 / 7,312
	$T5_{LARGE}$ (pipeline)	10,595 / 7,330
	$BART_{BASE}$ (multitask)	9,857 / 6,781
	$BART_{LARGE}$ (multitask)	10,288 / 7,142
	$T5_{SMALL}$ (multitask)	10,404 / 7,191
	$T5_{BASE}$ (multitask)	10,537 / 7,293
	$T5_{LARGE}$ (multitask)	10,566 / 7,302
	$BART_{BASE}$ (end2end)	1,033 / 756
	$BART_{LARGE}$ (end2end)	2,230 / 1,567
	$T5_{SMALL}$ (end2end)	6,555 / 4,520
	$T5_{BASE}$ (end2end)	7,090 / 4,913
	$T5_{LARGE}$ (end2end)	7,037 / 4,876
Reddit	Gold QA	3,268 / 1,634
	$BART_{BASE}$ (pipeline)	15,206 / 10,236
	$BART_{LARGE}$ (pipeline)	15,572 / 10,474
	$T5_{SMALL}$ (pipeline)	15,853 / 10,688
	$T5_{BASE}$ (pipeline)	16,112 / 10,844
	$T5_{LARGE}$ (pipeline)	16,199 / 10,898
	$BART_{BASE}$ (multitask)	14,928 / 10,037
	$BART_{LARGE}$ (multitask)	15,214 / 10,271
	$T5_{SMALL}$ (multitask)	15,756 / 10,585
	$T5_{BASE}$ (multitask)	15,866 / 10,704
	$T5_{LARGE}$ (multitask)	16,063 / 10,813
	$BART_{BASE}$ (end2end)	691 / 477
	$BART_{LARGE}$ (end2end)	2,055 / 1,407
	$T5_{SMALL}$ (end2end)	5,853 / 4,015
	$T5_{BASE}$ (end2end)	6,902 / 4,708
	$T5_{LARGE}$ (end2end)	6,632 / 4,560

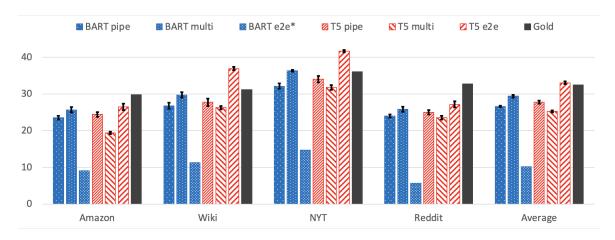Table 6: The number of question-answer pairs generated for SQuADShifts QA evaluation in each model.

Figure 3: Downsampled (equal-sized) SQuADShifts QA evaluation results (exact match with 95% confidence interval) for T5$_{\text{LARGE}}$ multitask/pipeline/end2end and BART$_{\text{LARGE}}$ pipeline, compared with the original result of each model and the gold QA dataset.