Multi-hop Question Generation without Supporting Fact Information

John Emerson, Yllias Chali
University of Lethbridge
Alberta, Canada
john.emerson@uleth.ca, yllias.chali@uleth.ca

Abstract

Question generation is the parallel task of question answering, where given an input context and optionally, an answer, the goal is to generate a relevant and fluent natural language question. Although recent works on question generation have experienced success by utilizing sequence-to-sequence models, there is a need for question generation models to handle increasingly complex input contexts with the goal of producing increasingly elaborate questions. Multi-hop question generation is a more challenging task that aims to generate questions by connecting multiple facts from multiple input contexts. In this work we apply a transformer model to the task of multi-hop question generation, without utilizing any sentence-level supporting fact information. We utilize concepts that have proven effective in single-hop question generation, including a copy mechanism and placeholder tokens. We evaluate our model's performance on the HotpotQA dataset using automated evaluation metrics and human evaluation, and show an improvement over the previous works.

Introduction

Question generation is an important task in information retrieval and interaction. The task aims to automatically generate fluent natural language questions that are related to a supplied input context. Optionally, an answer span within the input context may be identified in order to guide the generation process toward a specific topic. In addition to multiple practical applications, question generation models can also be used to automatically generate datasets for the parallel task of question answering.

Although question generation has been extensively researched in recent years, most of the previous work has focused on generating questions from input contexts of limited sizes, usually consisting of a single sentence or small paragraph (Du, Shao, and Cardie 2017), (Zhao et al. 2018). Furthermore, the questions generated by those works only require a limited amount of reasoning in order to be answered. Min et al. (2018) found that the answer to approximately 90% of the questions in the SQuAD dataset (Rajpurkar et al. 2016) can be derived from a single sentence within the input context.

Multi-hop question generation increases the difficulty of the question generation task by expanding the input context to include multiple, related paragraphs. The questions generated by a multi-hop QG system should require the reader to understand the content presented in each of the context paragraphs, and then reason over that information, connecting evidence from multiple paragraphs in order to formulate an answer. The increased size of the input context and the requirement to combine multiple pieces of evidence increase the difficulty of designing an effective model for this task. While a larger input context allows for the possibility of generating much more detailed and complex questions, it also increases the amount of irrelevant information that must be filtered out so that the generated questions do not stray to far from the desired topic.

In this paper, we examine the effectiveness of a transformer-based model for the multi-hop question generation task in a setting free of sentence-level supporting fact information. We utilize techniques that have proven effective when implementing single-hop question generation models based on the transformer architecture, including a copy mechanism and placeholder tokens. We utilize the HotpotQA dataset (Yang et al. 2018) to train our model and evaluate its performance. We then compare our results to the previous works on multi-hop question generation. The contributions of this paper are as follows: (1) Demonstrate that transformer models are an effective way to generate high-quality questions in a multi-hop setting by showing an improvement in multiple evaluation metrics; (2) Show that techniques such as a copy mechanism and placeholder tokens, which have proven effective in the single-hop setting, can be successfully leveraged for the multi-hop question generation task.

Related Work

The previous research on question generation falls into two main categories: rule-based approaches and neural-based approaches.

Rule-based approaches to single-hop question generation utilize carefully designed, hand-crafted rules to transform sentences from their declarative form into a question. Most of the early work on question generation falls into this category (Heilman and Smith 2010), (Lindberg et al. 2013), (Chali and Hasan 2015).

Neural-based approaches to single-hop question generation rely on the sequence-to-sequence paradigm (Sutskever, Vinyals, and Le 2014). They have become the dominant approach since (Du, Shao, and Cardie 2017) first utilized an attention-based model to generate reading comprehension questions from sentences and paragraphs. More recently, Scialom, Piwowarski, and Staiano (2019) investigated the effectiveness of transformer models for the single-hop question generation task.

Multi-hop question generation is a fairly new task and as such, little work has been completed on this topic. Gupta et al. (2020) were the first to tackle this problem. They utilized reinforcement learning to predict sentences within input contexts that support the creation of a question requiring multi-hop reasoning. They employed a Bi-LSTM to encode the input contexts and leveraged an attention-based decoder with a copy mechanism to generate the multi-hop questions. This approach can only be applied to datasets that are explicitly labelled with supporting fact information. Su et al. (2020) addressed the more general version of the multi-hop question generation task, which does not rely on identifying supporting facts within the input contexts. They utilized a graph convolutional network in conjunction with LSTMs and a reasoning gate to encode context information with multiple-hops. Their decoder consisted of an LSTM and a maxout pointer generator.

Proposed Model

The previous works on multi-hop question generation have relied on seqence-to-sequence models where both the encoder and decoder are comprised of LSTMs. Although these models have proven to be very effective, models based on recurrence offer little opportunity for parallelization during the training process. Vaswani et al. (2017) proposed the transformer as an efficient alternative to recurrence-based sequence transduction models. The transformer relies heavily on the concept of self-attention and positional encoding in order to process sequential data in a parallel fashion. Our model utilizes the transformer architecture as the basis of both the encoder and the decoder. We modify the decoder with a pointer-generator, allowing it to copy out of vocabulary tokens from the input contexts in addition to selecting tokens from a fixed vocabulary. We also incorporate the use of placeholder tokens into our model to improve its ability to handle out of vocabulary named entities.

Placeholder Tokens

Scialom, Piwowarski, and Staiano (2019) demonstrated that using placeholder tokens is an effective way to improve the performance of transformers when applied to the single-hop question generation task. Their research utilizes SQuAD dataset (Rajpurkar et al. 2016), where $\sim$52% of answers contain named entities. Since neural question generation models typically utilize vocabularies of a fixed size, many tokens in the input context will not be a part of this vocabulary and will instead be represented by the *[unk]* token. The authors replace each named entity token within the input context and reference question with a placeholder token corresponding to the type of named entity. For example, the sentence "Taylor Swift can really sing." would be preprocessed into "*[person_1] [person_2]* can really sing." As a post-processing step, all placeholder tokens in the generated question are converted back into the original named entities that they represent.

We utilized the spaCy library (Honnibal and Montani 2017)[1] to perform named entity recognition on the HotpotQA dataset (Yang et al. 2018) to determine the prevalence of named entities. We found that $\sim$75% of answers contained at least one named entity token and $\sim$60% of answers were comprised entirely of named entity tokens. As a result, we employ the same placeholder technique described above.

Encoder

We utilize the standard transformer encoder detailed by Vaswani et al. (2017), including the frequency-based positional encoding that they describe.

First, all tokens in the input contexts are embedded. The embeddings that represent placeholder tokens are learned, while the embeddings corresponding to non-placeholder (standard) tokens are frozen. We use this approach since learning the embeddings for the large number of standard tokens greatly increases the number of trainable parameters, along with the model's tendency to overfit to the training data. We generate answer position tags using the BIO tagging scheme (Zhou et al. 2017) and part-of-speech (POS) tags using the SpaCy library. The answer position and POS tags are embedded with learnable vectors. Finally, the three embeddings (token, answer position, and POS) are concatenated and summed with the positional encoding.

Since the input to the model consists of two separate but related context paragraphs, we separate them with a *[SEP]* token when encoding them. The motivation behind explicitly marking this boundary is to aid the model in encoding the connection between relevant entities across the two context paragraphs, resulting in a question that requires multiple reasoning hops.

Decoder

We utilize the standard transformer decoder described by Vaswani et al. (2017), modified with a pointer-generator similar to the design used by See, Liu, and Manning (2017). The pointer-generator gives the model the ability to choose between copying an out of vocabulary token from the input contexts or selecting a token from the fixed vocabulary. Similar to the encoder, embeddings corresponding to the placeholder tokens are learned, while embeddings corresponding to the standard tokens are frozen. The decoder also uses the frequency-based positional encoding described by Vaswani et al. (2017).

Pointer-Generator We utilize a pointer-generator inspired by See, Liu, and Manning (2017), but adapted for the transformer model. Such implementations are common in works utilizing transformer-based models (Prabhu and Kann

[1] https://spacy.io/

Context 1: The Androscoggin Bank Colisée (formerly Central Maine Civic Center and Lewiston Colisee) is a 4,000 capacity (3,677 seated) multi-purpose arena, in Lewiston, Maine, that opened in 1958. In 1965 it was the location...
Context 2: The Lewiston Maineiacs were a junior ice hockey team of the Quebec Major Junior Hockey League based in Lewiston, Maine. The team played its home games at the Androscoggin Bank Colisée. They were the second...
Answer: 3,677 seated.
Reference question: The arena where the Lewiston Maineiacs played their home games can seat how many people?

Figure 1: Bridge-type question from the HotpotQA dataset (Yang et al. 2018). Androscoggin Bank Colisée acts as a bridge entity between the context paragraphs.

2020), (Jiang et al. 2021). For each position in the generated question the pointer-generator utilizes a soft switch that determines whether to select a token from the fixed vocabulary or copy an out of vocabulary token from the input contexts.

$$p_{gen} = \sigma(w_{h^*}^T h_t^* + w_s^T s^t + w_x^T x_t + b_{ptr}) \quad (1)$$

$$P(w) = p_{gen} p_{vocab}(w) + (1 - p_{gen}) \sum_{i:w_i=w} a_i^t \quad (2)$$

The soft switch $p_{gen} \in [0, 1]$ for position t in the generated question is shown in equation 1, where w_{h^*}, w_s, and w_x are learnable vectors, and b_{ptr} is a learnable scalar. s^t represents the decoder hidden state, x_t represents the input to the decoder, and h_t^* represents a context vector which is calculated by summing the encoder hidden states, which are weighted by an attention distribution a^t. We derive a^t by averaging the heads of the encoder-decoder attention layer from the last decoder block. The probability of a token w appearing at position t in the generated question is calculated by summing its generation probability and copy probability, as shown in Equation 2, where i represents each position in the input context where the token w occurs.

Experiment

Dataset

We utilize the HotpotQA dataset (Yang et al. 2018) to train our model and evaluate its performance. HotpotQA was originally designed for the question answering task and consists of approximately 113k samples derived from Wikipedia articles. Each sample requires multiple reasoning hops across multiple input contexts in order to create a path of reasoning between a question and an answer. Samples in the HotpotQA dataset are divided into two different question types: *bridge* and *comparison*. Questions belonging to the bridge samples were generated by identifying a bridge entity that connects the two context paragraphs. Questions belonging to comparison samples were formed by selecting a different entity from each context paragraph and asking about a property that they share.

In a similar fashion to Su et al. (2020), we remove samples containing yes/no questions from the dataset, resulting in ∼92K usable samples. Of those samples, ∼79K are bridge questions and ∼13K are comparison questions. We split the usable samples into training (80%), validation (10%), and test (10%) sets. Since each sample also specifies its level of difficulty, we stratify the splits on this attribute, as well as on the question type.

Implementation Details

We implemented our model in Python using the PyTorch framework (Paszke et al. 2019)[2]. The encoder and decoder each consist of three layers, and each attention layer consists of 8 heads. The token embeddings have a dimension of 300, while the POS embeddings and answer position embeddings are of dimension 16 and 4, respectively. While the position-wise feed-forward layer has an intermediate dimension of 640, all other layers in the model have a hidden dimension of 320. We utilize the Adam optimizer (Kingma and Ba 2014) in conjunction with the learning rate schedule detailed by (Vaswani et al. 2017) with 4000 warmup steps. The dropout probability for all layers is set to .10. The Xavier initialization scheme (Glorot and Bengio 2010) is used to initialize all of the model's weights. We utilize a batch size of 16 and train the model for 50 epochs. For this model configuration we found that greedy decoding algorithm produced the best results.

The encoder and decoder utilize a shared vocabulary, which is built by considering the standard tokens and placeholder tokens separately. We first determine which standard tokens are included by selecting the 40,000 most frequent tokens that appear at least twice in the training data and are present in the glove.840B.300d.txt[3] file. Even though we randomly initialize the token embeddings, we still found it beneficial to restrict the vocabulary in this way. We then select all placeholder tokens that appeared more than five times within the training data. We do not convert the tokens to lower case during preprocessing.

Automatic Evaluation

We utilize the automated metrics that are commonly used in the previous work on automatic question generation. These metrics include BLEU (Papineni et al. 2002), ROUGE-L (Lin 2004), and METEOR (Lavie and Agarwal 2007).

Table 1 contrasts the performance of our model with (Su et al. 2020), who also tackle the multi-hop question generation task in a setting free of sentence-level supporting fact information. We generated the scores for our model by using the same evaluation script (Sharma et al. 2017)[4] as (Su et al. 2020).

[2]https://pytorch.org/
[3]https://nlp.stanford.edu/projects/glove/
[4]https://github.com/Maluuba/nlg-eval

	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE-L	METEOR
MulQG (Su et al. 2020)	40.15	26.71	19.73	15.20	35.30	20.51
Our Model	**42.13**	**30.44**	**23.84**	**19.42**	**39.26**	**22.78**

Table 1: Automatic evaluation metrics between our model and (Su et al. 2020)

	Syntactic correctness	Relevance
MulQG (Su et al. 2020)	4.4	3.34
Our Model	**4.72**	**3.94**

Table 2: Human evaluation between our model and (Su et al. 2020)

Human Evaluation

To further assess the performance of our system, we performed human evaluations on the results. Three English-speakers were asked to give a score from 1 (very poor) to 5 (very good) to the questions generated from both systems according to two criteria: syntactic correctness and relevance. Syntactic correctness indicates the grammaticality and the fluency and relevance demonstrates whether the question is meaningful and related to the sentence it is generated from. The three assessors performed the evaluations on 100 randomly selected question and answer pairs from the results. Table 2 shows the performance of our model with (Su et al. 2020).

Conclusion

In this work we have shown that transformer-based models are an effective solution to the multi-hop question generation task in a setting free of sentence-level supporting fact information. We also show that enhancements such as placeholder tokens and a copy mechanism, which have proven effective in the single-hop question generation task, can be successfully leveraged for the multi-hop variation. Finally, our results demonstrate a notable improvement in multiple evaluation metrics.

Acknowledgements

The research reported in this paper was conducted at the University of Lethbridge and supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada discovery grant and the NSERC Alliance - Alberta Innovates Advance Program grant.

Examples of System Generated Questions

Question Type: Bridge.
Reference Question: The first televised speech in the House of Commons of the United Kingdom was made by a politician that was assassinated by what group ?
Generated Question: The first televised speech in the House of Commons of the United Kingdom was made by a British Conservative politician who was assassinated by what organization ?

Question Type: Bridge.
Reference Question: The horse - collar tackle is most closely associated with a professional football player who was drafted by what team in 2002 ?
Generated Question: The horse - collar tackle is most closely associated with a former American college and professional football player who was drafted by what team in 2002 ?

Question Type: Bridge.
Reference Question: The Pangaea Expedition is a worldwide voyage led by a South African - born Swiss adventurer who studied what at Stellenbosch University ?
Generated Question: The Pangaea Expedition is a worldwide voyage led by a South African - born Swiss professional explorer and adventurer who studied what at Stellenbosch University in Western Cape ?

Question Type: Bridge.
Reference Question: Hobbycraft and it 's competitor High Street are located where ?
Generated Question: Hobbycraft is an arts and crafts superstore chain in which country ?

Question Type: Comparison.
Reference Question: Which was founded first Virginia Commonwealth University or Batangas State University ?
Generated Question: Which university was founded first Virginia Commonwealth University or Batangas State University ?

Question Type: Comparison.
Reference Question: Which was released first Point of Order or The Celluloid Closet ?
Generated Question: Which documentary was released first Point of Order or The Celluloid Closet ?

Question Type: Comparison.
Reference Question: Which is taller Trump Tower or Americas Tower ?
Generated Question: Which skyscraper is taller Trump Tower or Americas Tower ?

References

Chali, Y., and Hasan, S. A. 2015. Towards topic-to-question generation. *Computational Linguistics* 41(1):1–20.

Du, X.; Shao, J.; and Cardie, C. 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)*, 1342–1352. Vancouver, Canada: Association for Computational Linguistics.

Glorot, X., and Bengio, Y. 2010. Understanding the difficulty of training deep feedforward neural networks. In *AISTATS*.

Gupta, D.; Chauhan, H.; Tej, A. R.; Ekbal, A.; and Bhattacharyya, P. 2020. Reinforced multi-task approach for multi-hop question generation.

Heilman, M., and Smith, N. A. 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*, 609–617. Los Angeles, California: Association for Computational Linguistics.

Honnibal, M., and Montani, I. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Jiang, W.; Li, J.; Chen, M.; Ma, J.; Wang, S.; and Xiao, J. 2021. Improving neural text normalization with partial parameter generator and pointer-generator network. In *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 7583–7587.

Kingma, D. P., and Ba, J. 2014. Adam: A method for stochastic optimization.

Lavie, A., and Agarwal, A. 2007. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the Second Workshop on Statistical Machine Translation*, StatMT '07, 228–231. USA: Association for Computational Linguistics.

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

Lindberg, D.; Popowich, F.; Nesbit, J.; and Winne, P. 2013. Generating natural language questions to support learning on-line. In *Proceedings of the 14th European Workshop on Natural Language Generation*, 105–114. Sofia, Bulgaria: Association for Computational Linguistics.

Min, S.; Zhong, V.; Socher, R.; and Xiong, C. 2018. Efficient and robust question answering from minimal context over documents. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1725–1735. Melbourne, Australia: Association for Computational Linguistics.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, 311–318. USA: Association for Computational Linguistics.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. Pytorch: An imperative style, high-performance deep learning library. In Wallach, H.; Larochelle, H.; Beygelzimer, A.; dÁlché Buc, F.; Fox, E.; and Garnett, R., eds., *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc. 8024–8035.

Prabhu, N., and Kann, K. 2020. Making a point: Pointer-generator transformers for disjoint vocabularies. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing: Student Research Workshop*, 85–92. Suzhou, China: Association for Computational Linguistics.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. Squad: 100,000+ questions for machine comprehension of text.

Scialom, T.; Piwowarski, B.; and Staiano, J. 2019. Self-attention architectures for answer-agnostic neural question generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 6027–6032. Florence, Italy: Association for Computational Linguistics.

See, A.; Liu, P. J.; and Manning, C. D. 2017. Get to the point: Summarization with pointer-generator networks.

Sharma, S.; El Asri, L.; Schulz, H.; and Zumer, J. 2017. Relevance of unsupervised metrics in task-oriented dialogue for evaluating natural language generation. *CoRR* abs/1706.09799.

Su, D.; Xu, Y.; Dai, W.; Ji, Z.; Yu, T.; and Fung, P. 2020. Multi-hop question generation with graph convolutional network. In *Findings of the Association for Computational Linguistics: EMNLP 2020*. Association for Computational Linguistics.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to sequence learning with neural networks.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need.

Yang, Z.; Qi, P.; Zhang, S.; Bengio, Y.; Cohen, W. W.; Salakhutdinov, R.; and Manning, C. D. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering.

Zhao, Y.; Ni, X.; Ding, Y.; and Ke, Q. 2018. Paragraph-level neural question generation with maxout pointer and gated self-attention networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 3901–3910. Brussels, Belgium: Association for Computational Linguistics.

Zhou, Q.; Yang, N.; Wei, F.; Tan, C.; Bao, H.; and Zhou, M. 2017. Neural question generation from text: A preliminary study.