A Generative Approach to Question Answering

Rajarshee Mitra
Microsoft,
Hyderabad
ramitra@microsoft.com

Abstract

Question Answering has come a long way from answer sentence selection, relational QA to reading and comprehension. We shift our attention to generative question answering (gQA) by which we facilitate machine to read passages and answer questions by learning to generate the answers. We frame the problem as a generative task where the encoder being a network that models the relationship between question and passage and encoding them to a vector thus facilitating the decoder to directly form an abstraction of the answer. Not being able to retain facts and making repetitions are common mistakes that affect the overall legibility of answers. To counter these issues, we employ copying mechanism and maintenance of coverage vector in our model respectively. Our results on MS-MARCO demonstrates it's superiority over baselines and we also show qualitative examples where we improved in terms of correctness and readability.

1 Introduction

Question Answering is a crucial problem in language understanding and a major milestone towards human-level machine intelligence. Datasets like SQuAD, MS-MARCO and others have led to plethora of contributions in machine reading and comprehension. The next-generation QA systems can be envisioned as the ones which can read passages and write long answers to questions. We formulate generative question answering as a form of QA where we expect the machine to produce an abstractive answer **A** that encompasses all the information from passage **P** required to answer a question **Q**. Eventually, such systems, if capable of understanding what information is necessary and what is not, will enable us to acquire information from multiple sources and present them in the form of a summarized answer.

The assumption, prevalent in most of the existing approaches, that the answer should be always a particular sub-span in the passage is a very strict one.

Generating answers should have to carefully incorporate any facts and entities which is necessary to answer the question as well as simultaneously discarding irrelevant information from P. This requires building complex relations between the question and the passage. What makes it further challenging is not only a need for good generative model but also the readability of the generations. Even if one achieves good results in terms of lexical similarity metrics like ROUGE-L, the determination of how much *correctness* and *readability* is preserved is a very significant concern in this task.

Our generative model is inspired by *Seq2Seq* model (Sutskever et al. (2014) which is the basis of various NLG tasks like translation and summarization. In this paper, we propose our model that learns alignment between question and passage words to produce rich query-aware passage representation and using this same representation to directly decode the answer while attending to all the states in the representation. Learning end-to-end, our approach has no dependency on any external extractive labels. We also propose an approach where we make our decoder RNN state computation attention-aware by modifying the computed state of the previous step with attended encoder context.

While building such model, we noticed that, often, it replaces correct entities with similar incorrect ones (eg. correct year by incorrect year). This

hinders the overall correctness of the answer being generated. To tackle this, we incorporate a *copying mechanism* (Gu et al. (2016), See et al. (2017)) that learns when to copy an important entity directly from the passage instead of generating anything from vocabulary. This makes our approach *abstractive-cum-extractive*. Furthermore, a common error in generative models is repetitiveness in the text being generated. This also affects the generation that follows. A common practice, being used in similar tasks, like machine translation and summarization, is keeping track of a coverage vector (Tu et al. (2016)) that keeps track of which encoder states have been attended to what extent in the past.

Our main novelty lies in demonstrating that it is possible to generate answers directly from modeled relationship between question and passage without the need to build extraction model or providing any positional labels (like start/end). We show that the decoder with the help of pointer-generator networks can itself choose to copy or generate answer words.

2 Related Work

Traditional QA models like Kumar et al. (2016) and Seo et al. (2016a) have shown some fascinating results in the form of relational based QA Weston et al. (2015) or machine reading and comprehension by Wang and Jiang (2016), (Seo et al., 2016b) and Wang et al. (2017) with promising results. Their methods proved how successful is pointer networks of Vinyals et al. (2015). Considerable amount of work has also been done on passage ranking (Tang et al., 2017). Tan et al. (2017) has taken a generative approach where they add a decoder on top of their extractive model and thus leveraging the extracted evidence to synthesize the answer. However, the model still relies strongly on the extraction to perform the generation which means it is essential to have the start and end labels (a span) for every example. Even also, when they are multiple regions in P that contribute to A, the model will need to predict multiple spans.

Our approach differs from this in the sense that it only relies on the content of Q and P to generate the answer and the extraction-abstraction *soft* switch happens as a part of the learning procedure without the need for any dependency on extraction.

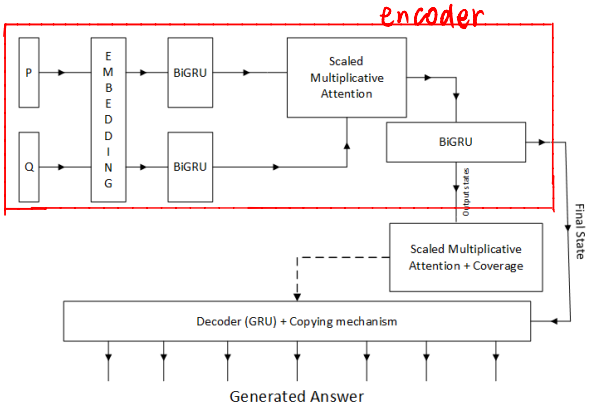


Figure 1: Block diagram of our model. Both Attention1 and Attention2 is the scaled multiplicative attention we discuss in (3).

3 Model Details

Our model consists of an encoder that compute representation by using context and attentive layers and an attentive (Bahdanau et al. (2014)) decoder with pointer-generator networks to decide when to copy or generate. We also keep a track of the attention at each time step to maintain a coverage vector to improve readability.

3.1 Representation

We use the standard GRU (Cho et al. (2014)) as the building block of our recurrence for computing representations for both questions and passages.

Initially, both **P** and **Q** can be expressed by their respective word embeddings as Q = $\{w_t^Q\}_{t=1}^m$ and P = $\{w_t^P\}_{t=1}^n$.Further, the representations are built by multi-layered bi-directional GRU and weight sharing being done between **P** and **Q**.

$$u_t^Q = \text{BiGRU}_\theta(u_{t-1}^Q, w_t^Q)$$
$$u_t^P = \text{BiGRU}_\theta(u_{t-1}^P, w_t^Q) \quad (1)$$

where u_t^Q and u_t^P are the hidden states of the GRU at t_{th} time step.

3.2 Attentive Layer

We use a multiplicative attention with a scaling factor (Vaswani et al., 2017) to compute alignment between question and passage words u_i^Q and u_i^P respectively. Specifically, for each u_i^P, we take the weighted sum of all u^Q which is then concatenated to u_i^P to form it's final representation. We also use gating mechanism (Tan et al., 2017) to provide varying importance to passage words (4).

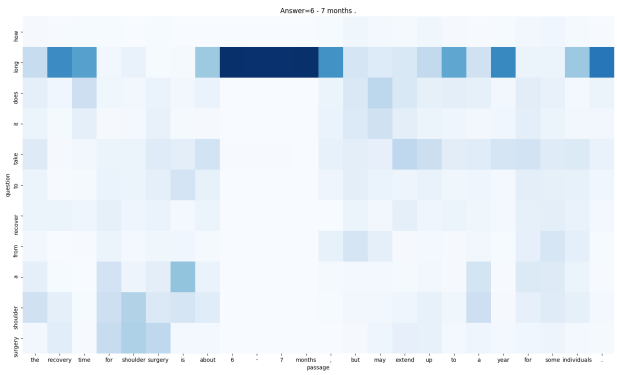


Figure 2: An alignment matrix between Q and P shows how the model associates each word in Q with each word in P. As an example, when the model reads the word *long* in the question, it focuses most of it's attention on *6-7 months* in passage. All the columns are softmax normalized.

Mathematically, we can express the attentive layer as:

$$u^{Q^-} = tanh(W^Q u^Q); u^{P^-} = tanh(W^P u^P)$$
$$a_{ij} = \frac{1}{\sqrt{D}}(u_i^{P^-}.u_j^{Q^-T})[D = hidden\_dim] \quad (2)$$
$$w_i = softmax(a_i)$$
$$v_t^P = w_t u^Q \quad (3)$$
$$g = \sigma(W_g[v_t^P; u_t^P]) \quad (4)$$
$$[v_t^P; u_t^P]^* = g[v_t^P; u_t^P]$$
$$c_t = \text{BiGRU}(c_{t-1}, [v_t^P; u_t^P]^*) \quad (5)$$

u^{Q^-} and u^{P^-} are non-linear ($tanh$) transformations of question and passage representations u^Q and u^P respectively. a computes dot product attention between transformed query and passage representations followed by a scaling factor $\sqrt{D}$. This also implies that for each passage word, we find the weighted importance of all question words, scale them and finally a softmax normalization to produce weights. With the weights, we thus get, we find the weighted sum of all the question words for that particular passage word (3). This is the new question-attended passage representation v_t^P. We further concatenate this with the original passage representation u_t^P followed by a sigmoid gating (4) to represent how important this passage time step is for the encoding.

The penultimate stage of the attentive layer contains a bi-directional GRU (5) that acts a smoothing layer over the concatenation of original and question-attended passage representation.

We concatenate c_T and u_T^Q and perform a non-linear transformation of it before passing them to the decoder.

$$c_T = [\overleftarrow{c_T}; \overrightarrow{c_T}]; h = \tanh(W_h[c_T; u_T^Q]) \quad (6)$$

Figure 2 shows a particular example of how relationship between question and passage is modeled.

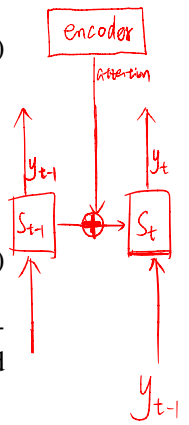
3.3 Decoding

We use the GRU cell to compute our decoder states and the same scaled attention as in 2.
The decoder decodes the information encoded by the query-aware passage encoder, aggregating information from various parts in the passage to produce the final answer. As we have seen in Figure 2, different regions of the passage can be weighted more if they are relevant to the query. In such cases, often we have seen from question-passage heatmaps, several areas in the passage are marked important. For eg. when the query is asking for some *year*, all the words in the passage which denotes a year stands out more or less from other passage words (this issue is also highlighted in Table 1). It is now importantly the decoder's job to pick the correct year. The decoder is initialized with combination of both question representation encoded passage representation (from 6)

Making the RNN attention-aware: We use the attended context concatenated with previous decoder state as input to the RNN to compute the current state. This makes the RNN aware about previous attentions. This differs from (Luong et al., 2015), where they attend after RNN state computation. We have observed that our difference causes a gain of at least 1 ROUGE-L point. The decoder state computation can be expressed as:

$$h^- = tanh(W_h h_i); s_{t-1}^- = tanh(W_s s_{t-1})$$
$$e_i^t = \frac{1}{\sqrt{D}}(W_h h_i^- W_s s_{t-1}^-)$$
$$a^t = \text{softmax}(e^t)$$
$$h_t^* = \sum_i a_i^t h_i; s_{t-1}^* = \tanh(W_c[h_t^*; s_{t-1}])$$
$$s_t = GRU(s_{t-1}^*, y_{t-1}) \quad (8)$$

(7)



where a_t is the attention distribution over the encoder states at t^{th} time-step in the decoder and

Question: when was the death penalty abolished?
Passage: The last executions UK took place in 1964 , and the death penalty was abolished in 1998 . In 2004 the UK became a party to the 13th Protocol to the European Convention on Human Rights and prohibited the restoration of the death penalty . Death penalty , capital punishment , or execution is the legal process of putting a person to death as a punishment for a crime . Modern History of Death Penalty . 1608 - Earliest death penalty in the British American Colonies handed out for UNK
Model with copying: 1998
Model without copying: 1964

Table 1: Replacement of correct entity by negative and similar one. We show two results from two models – with and without point-gen

Q: what is a urethra
P: in anatomy , the urethra is a tube that connects the urinary bladder to the urinary meatus for the removal of fluids from the body . 1 infection of the urethra is urethritis , said to be more common in females than males . 2 urethritis is a common cause of dysuria (pain when urinating) . 3 related to urethritis is so called urethral syndrome . 4 passage of kidney stones through the urethra can be painful , which can lead to urethral strictures .
Model with coverage: is a tube that connects urinary bladder to the urinary meatus for the removal of fluids from the body .....
Model without coverage: the urethra a tube that connects urinary urinary bladder the the tube that connects the urinary bladder fluids the urinary body

Table 2: Here are two types of repetitions we notice: consecutive identical (two times "urinary") words and duplicate phrases ("tube that connects the urinary bladder"). We show two results from two models – with and without coverage

s_t is the t^{th} decoder hidden state. The attention methodology used here is identical to the one used in the attentive layer. Based on a_t, the decoder decides which encoder state to focus more on. h_t^* is the attended context vector of the encoder at t_{th} decoding time step. We use this context vector in concatenation with the decoder GRU's previous state s_{t-1} to compute current state s_t. Finally, we compute the decoder output probability as $P_{vocab} = \text{softmax}(W_y s_t + b_y)$. This is a *fully* abstractive approach.

3.3.1 Copying from source

However, to overcome the limitation of missing out entities from **P** (Table 1) or when dealing mostly extractive data, we apply pointer-generator networks to make it *abstractive-cum-extractive*.With pointer-generator model, at each decoder time-step, we make a *probabilistic* switch between whether to copy or simply choose from output distribution P_{vocab}. This is governed by p_{gen} which is computed as (See et al. (2017)):

$$p_{gen} = \sigma(w_h^T h_t^* + w_s^T s_t + b_{\text{ptr}}) \quad (9)$$

At each decoder time step, we use the decoder state s_t and the context vector h_t^*, to decide whether to copy from the encoder words or to generate the highest probable word from the decoder output.

$$P(w) = p_{gen} p_{vocab}(w) + (1 - p_{gen}) \sum_{i:w_i=w} a_i^t \quad (10)$$

From a high-level point of view, the pointer-generator model, at each decoder step t, adds the attention probabilities of each encoder words from a_t to their respective probabilities in the decoder output distribution p_{vocab} at that decoder time step. A sigmoid over the affine transformations produce a probability. Higher p_{gen} means the model

will mostly choose a word from vocabulary while lower means higher chance of copying a source word from passage. $P(w)$ is the final probability distribution which is used for generating answer word and computing the training loss.

3.3.2 Mitigating repetitions through coverage

Readability is a major concern in almost all NLG tasks (Table 2). The copying mechanism helps to preserve information from the encoder side and prevent repetition of words which we have observed in our experiments. We realized that alleviating this problem will eventually lead to better readability of answers. Hence, we incorporated *coverage mechanism*, which is pretty popular approach in machine translation, summarization, into our model. It is a mechanism in MT to avoid over-translation by keeping track of which source words are receiving attention too many times. Specifically, we used the attention distributions being computed at each decoder time step to maintain a coverage vector cov_t which is basically cumulative sum of attention probabilities.

$$cov_t = \Sigma_{i=0}^{t-1} a_i$$

Hence, at each time step, the decoder has information about how much each encoder state has been attended until the previous step. We add an extra term $W_c cov_t$ to (7). This makes the standard attention mechanism has a knowledge about which states has been attended already enough in the past and thus manages to curb repetitions.

3.4 Loss

At each decoder time step, we use the negative log-likelihood of the correct word and finally try to minimize the total loss over all the decoder steps.

$$loss = -\frac{1}{T}\sum_{t=0}^{T} \log P(w_t^*) \quad (11)$$

4 Data and Experiments

We conducted our experiments on MS-MARCO[1] data (Nguyen et al. (2016)). In our experiments, to form the passage-answer pair from training data, we select the correct input passage for an answer by determing which passage has a sub-span with highest ROUGE-L score with the reference answer Tan et al. (2017). We also verify that this ROUGE-L score is not less than 0.7. The filtered train data consists of 75000 examples. For evaluation, we take the correct passage for each query and generate answer from them. We relied on Stanford CoreNLP Manning et al. (2014) to tokenize all texts. Also, We restricted the number of words in it to 30000 and lengths of P and A to 200 and 50 respectively. We use a batch of 50 examples for updating our model while training for roughly 15000 iterations. We use *Glove* (Pennington et al. (2014)) to represent words and keep them *fixed*. We use hidden state dimension of 256 throughout the network. We use the Adadelta optimizer (Zeiler (2012)), with *epsilon*=1e-6, *rho*=0.95 and initial learning rate=1, to minimize our loss.

[1]Data is available for download at http://www.msmarco.org/dataset.aspx

Model	ROUGE-L	Perplexity
Seq2Seq baseline	-/37.70	-/-
gQA (+p-gen,+ cov.)	**74/59.5**	4.35/4.62
gQA (+p-gen)	70/57.5	3.36/4.05
gQA (w/o p-gen)	45/42.	4.3/4.48

Table 3: The best performance on the MS-MARCO development data, given the correct passage. We report the perplexity alongside ROUGE-L score. For each numeric column, score follows the *train/dev* format.

Question	average cost of a small bathroom
Passage	Homeowners have many options when it comes to bathroom remodels and the total cost depends on style and budget. The average bathroom remodel costs $ 9,254, but you can spend $ 3,500 and $ 7,000 - to fix up the essentials in a small-to medium-sized bathroom. On the other end of the spectrum, you could spend $ 13,000 to more than $ 20,000 turning your master bathroom into an oasis. Bathroom remodels provide some of the highest resale returns as a home improvement project.
Actual Answer	The average bathroom remodel costs $9,254.
Predicted Answer	$ 9,254

Figure 3: Example result: core information retained

5 Results and Analysis

We list our result on MS-MARCO data in Table 3. We do not consider negative passages and test

Question	determines meaning
Passage	determined. adjective de · ter · mined. : having a strong feeling that you are going to do something and that you will not allow anyone or anything to stop you . : not weak or uncertain : having or showing determination to do something
Actual Answer	Having a strong feeling that you are going to do something and that you will not allow anyone or anything to stop you
Predicted Answer	a strong feeling that you are going to do something and that you will not allow anyone, or anything stop you

Figure 4: Example result: almost same answer

the model only on correct passages for each query. We also include the results reported in Table 6 of Tan et al. (2017) on experimenting with basic sequence-to-sequence model with selected passage as the input. Results show that our model outperforms the generative baseline by a large margin. The learnt mean value of p_{gen} is mostly 0.7 which tells us that the data is mostly extractive in nature and forces the model to copy mostly. We also tried to use self-attention but that didn't improve performance, most probably, because the decoder attention is already aggregating information from different parts of passage. Not only improvement in score, but also coverage reduced repetitions in multiple instances. We provide some examples in Fig.3 and Fig.4.

6 Conclusion

We successfully build a generative model that can not only efficiently model relationship between question and passage but also can generate answers from the encoded relationship. We let the model decide to be in abstractive or extractive mode based on the nature of the data thus removing any dependency on any other external extractive feature. Moreover, we apply coverage in this QA task to mitigate frequent repetitions.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

Kyunghyun Cho, Bart van Merrienboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *CoRR*, abs/1406.1078.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Romain Paulus, and Richard Socher. 2016. Ask me anything: Dynamic memory networks for natural language processing. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, pages 1378–1387.

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, System Demonstrations*, pages 55–60.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A human generated machine reading comprehension dataset. In *Proceedings of the Workshop on Cognitive Computation: Integrating neural and symbolic approaches 2016 co-located with the 30th Annual Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain, December 9, 2016.*

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. *CoRR*, abs/1704.04368.

Min Joon Seo, Hannaneh Hajishirzi, and Ali Farhadi. 2016a. Query-regression networks for machine comprehension. *CoRR*, abs/1606.04582.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016b. Bidirectional attention flow for machine comprehension. *CoRR*, abs/1611.01603.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Chuanqi Tan, Furu Wei, Nan Yang, Weifeng Lv, and Ming Zhou. 2017. S-net: From answer extraction to answer generation for machine reading comprehension. *CoRR*, abs/1706.04815.

Duyu Tang, Nan Duan, Tao Qin, and Ming Zhou. 2017. Question answering and question generation as dual tasks. *CoRR*, abs/1706.02027.

Zhaopeng Tu, Zhengdong Lu, Yang Liu, Xiaohua Liu, and Hang Li. 2016. Modeling coverage for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA*, pages 6000–6010.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2692–2700.

Shuohang Wang and Jing Jiang. 2016. Machine comprehension using match-lstm and answer pointer. *CoRR*, abs/1608.07905.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 189–198.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2015. Towards ai-complete question answering: A set of prerequisite toy tasks. *CoRR*, abs/1502.05698.

Matthew D. Zeiler. 2012. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701.