The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Semi-Distantly Supervised Neural Model for Generating Compact Answers to Open-Domain Why Questions

**Ryo Ishida, Kentaro Torisawa, Jong-Hoon Oh,
Ryu Iida, Canasai Kruengkrai, Julien Kloetzer**
National Institute of Information and Communications Technology, Kyoto, 619-0289, Japan
{ishida.ryo, torisawa, rovellia, ryu.iida, canasai, julien}@nict.go.jp

Abstract

This paper proposes a neural network-based method for generating compact answers to open-domain why-questions (e.g., "Why was Mr. Trump elected as the president of the US?"). Unlike factoid question answering methods that provide short text spans as answers, existing work for why-question answering have aimed at answering questions by retrieving relatively long text passages, each of which often consists of several sentences, from a text archive. While the actual answer to a why-question may be expressed over several consecutive sentences, these often contain redundant and/or unrelated parts. Such answers would not be suitable for spoken dialog systems and smart speakers such as Amazon Echo, which receive much attention in these days. In this work, we aim at generating non-redundant compact answers to why-questions from answer passages retrieved from a very large web data corpora (4 billion web pages) by an already existing open-domain why-question answering system, using a novel neural network obtained by extending existing summarization methods. We also automatically generate training data using a large number of causal relations automatically extracted from 4 billion web pages by an existing supervised causality recognizer. The data is used to train our neural network, together with manually created training data. Through a series of experiments, we show that both our novel neural network and auto-generated training data improve the quality of the generated answers both in ROUGE score and in a subjective evaluation.

1 Introduction

Answering non-factoid questions, to which sentences and other long expressions are expected as answers, is a difficult task. Why-questions (e.g., "Why was Mr. Trump elected as the president of the US?") form a relatively well-studied class of non-factoid questions but the existing methods for why-question answering (why-QA) simply retrieve from a text archive relatively long text passages, each of which typically consists of several sentences as exemplified in the answer passages in Table 1, as the answer. The retrieved answers usually contain redundant parts, which make them unsuitable as answers to be read aloud by spoken dialog systems and smart speakers such as Amazon Echo. The aim of this work is to develop a method to generate non-redundant compact answers to Japanese open-domain why-questions from given answer passages that are retrieved by an existing web-based open-domain why-question answering system. The answers should be short and comprehensible so that they can be understood by users when read aloud as a question's responses by spoken dialog systems and smart speakers. We limit our compact answers to single sentences that end with the Japanese word "*tame* (because)", which correspond to sentences that starts with "because" in English, like the answer "Because the ocean's water mass is displaced by an earthquake" to the why-question "Why does a tsunami occur?" To the best of our knowledge, this is the first attempt to generate such compact answers to why-questions. We developed a novel neural network based method and exploit automatically generated training data, which is generated with the help of a *supervised* causality recognizer, in addition to manually created training data, to train the network. Accordingly, our method can be characterized as *semi-distantly supervised neural network model*. We created

Question: Why did Yubari City in Hokkaido prefecture go bankrupt? **Answer Passage:** The central government and the company should have paid **the costs of closing coal mines** to save the remaining inhabitants' lives. But the central government and Hokkaido's inadequate financial support **caused an excessive expense for Yubari's city government**, and the large remainder of municipal bonds due to this expense became the largest cause of Yubari's bankruptcy **Compact Answer:** Because the costs of closing coal mines caused an excessive expense of Yubari's city government.
Question: Why does a non-rotating shot (in football) move unpredictably? **Answer Passage:** A non-rotating shot is a shot that is kicked without rotating the ball as much as possible, while keeping the ball in a state where it **is likely to receive strong air resistance**. Since the trajectory of a non-rotating shot appears to sway irregularly, it is called a swaying ball, and, because it is unpredictable, it is the type of shots that goal keepers most hate to encounter. **Compact Answer:** Because the ball is likely to receive strong air resistance.

Table 1: Examples of why-questions, answer passages and compact answers

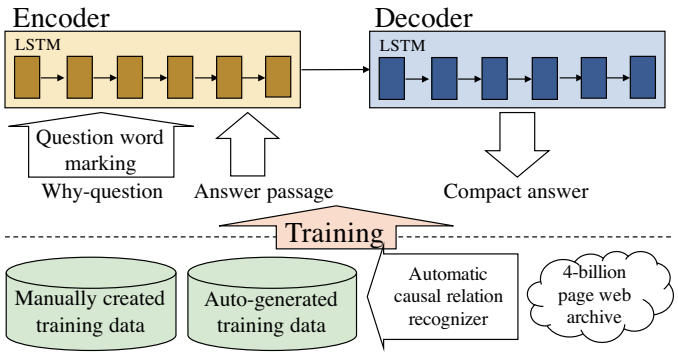

Figure 1: Proposed method

our own dataset for this task and the experimental results using the dataset revealed that both the novel network and auto-generated training data contributed to improving the quality of the generated answers.

Table 1 shows examples of why-questions, their correct answer passages that are provided by a publicly available *open-domain* why-QA system, WISDOM X (Mizuno et al. 2016; Oh et al. 2016), and gold-standard compact answers created by human annotators. (Our target language is Japanese but we use English translations of Japanese examples for ease of explanation throughout this paper.) The words in boldface in the passages are those appearing in the compact answers. As can be seen clearly, most parts of the answer passages are actually peripheral to the why-question and the compact answers are much shorter than the answer passages. Another important point is that the words in the compact answers may be scattered over several sentences in the passages. Consequently, finding the words that should appear in a compact answer and generating the answers are not trivial tasks. Also note that the underlined words are those appearing in the questions or their synonyms and that many of these underlined words do not appear in the compact answers. Although the underlined words are redundant, existing why-QA systems must search for them through a text archive to find the correct answer passages, and the passages must contain the words. Furthermore, those words, as well as those in the compact answer, may be scattered over several sentences. This is a reason why conventional why-QA systems must provide relatively long passages inevitably.

Figure 1 gives an overview of our method to generate a non-redundant compact answer from such redundant answer passages. We use an extension of a neural-network-based method for automatic summarization called a *pointer-generator network* (See, Liu, and Manning 2017). The network consists of two recurrent neural networks, which are called an *encoder* and a *decoder*, respectively. In our settings, the encoder reads an answer passage and the decoder generates a compact answer. We extend the network so that the words in the passage are *marked* if they also appear in the question and the resulting marks are given to the encoder. We call this technique *question word marking*. Our expectation is that by using such marks the encoder can more effectively focus on the part of the passage in which a compact answer is written and thus can propagate the information of this part to the decoder. This framework may appear similar to the query-focused neural summarizer (Nema et al. 2017). However, our method uses the information concerning a query (why-question) in the answer-passage encoder, while the query-focused neural summarizer does not use the information in the passage encoder. Actually, we show that the query-focused summarizer does not work well for our task in our empirical evaluation. In conventional query-focused summarization data (Nema et al. 2017), the words in queries are likely to appear in the gold-standard summary, but, in our data set, the question words are less likely to appear in the compact answer because they are often redundant. This difference may have contributed to the methods' difference in effectiveness.

Also, we use automatically generated data for training. For this purpose, we exploit an automatic causality recognizer (Oh et al. 2013), which can identify text expressing causal relations (e.g., "Mr. Trump was elected as the president because many people had ill feelings against the political elites.") and recognize their effect parts ("Mr. Trump was elected as the president") and cause parts ("many people had ill feelings against the political elites"). Then, we combine several causal relations identified by the recognizer to form a *virtual* passage, assuming that the cause part of one of the combined causal relations is a compact answer to be generated and the effect part of the same causal relation is a why-question. Through an empirical evaluation, we show that the quality of the answers generated by our method improves significantly by adding a large number of such virtual passages as training data to a small amount of manually created training data.

Our task is similar to the SQuAD machine comprehension task (Rajpurkar et al. 2016) in the sense that an answer is generated from a passage. In our experiments, we modified the neural QA system DrQA (Chen et al. 2017), which works well on the SQuAD, so that it can deal with Japanese and our task, and compare it with our proposed method, but it achieved significantly worse results than those of our methods. This suggests that our task is not a trivial

variant of the SQuAD task at least. In the SQuAD, answers must be short consecutive word sequences in passages, while our answers do not have to be consecutive words. Actually, around 40% of the gold-standard answers in our test set did not match any consecutive sequence in the passage. Additionally, the language difference may be another cause of the low performance of DrQA.

This paper is organized as follows. Section 2 gives an overview of related works. Section 3 presents our proposed methods. The dataset we created and the experimental results are described in Section 4.

2 Related Work

Several neural methods have been proposed for why-question answering (Oh et al. 2017; Sharp et al. 2016; Tan et al. 2016; dos Santos et al. 2016) and showed significant performance improvement over the methods using conventional machine learning methods such as SVMs (Girju 2003; Higashinaka and Isozaki 2008; Oh et al. 2012; 2013; 2016; Verberne et al. 2011). However, as mentioned in the introduction, their answers are passages consisting of several sentences and they cannot provide compact answers like ours.

Another task related to ours is automatic text summarization. Again, several neural methods for this task have been proposed (Rush, Chopra, and Weston 2015; See, Liu, and Manning 2017; Nema et al. 2017; Zhou et al. 2017) after the success of neural machine translation based on encoder-decoder models (Bahdanau, Cho, and Bengio 2014) since a similar methodology can be applied to summarization. As mentioned in the Introduction, we use pointer-generator network (See, Liu, and Manning 2017), which is one of the state-of-the-art method for the summarization task, as the starting point for developing our method. One of the most serious problems in neural summarization is rare words that do not appear in training data: a naive neural summarizer often generates irrelevant words in a summary at the position where such rare words in the original texts should be. The pointer-generator network can *copy* rare words from an original text to a summary, regardless whether they appear in training data or not, and can reduce the risk of generating irrelevant words.

3 Proposed Method

We extend the pointer-generator network (See, Liu, and Manning 2017), which is one of the state of the art automatic summarizer, to generate compact answers to why-questions. In the following, we explain the pointer-generator network and then describe the extension added to the network and our automatic training data generation method.

3.1 Pointer-generator Network

A pointer-generator network generates a summary using two LSTMs (Hochreiter and Schmidhuber 1997). The first LSTM, which is called *encoder*, reads an input passage, which is denoted by $p = \langle w_1, w_2, ..., w_{|p|} \rangle$ where w_i ($1 \leq i \leq |p|$) is a word, and generates vector representations of the passage. The other LSTM, which is called *decoder*, generates a summary $y = \langle y_1, y_2, ..., y_T \rangle$ where y_t ($1 \leq t \leq T$) is a word, from the vector representations. More specifically, we use as the encoder a bidirectional LSTM, which actually consists of two LSTMs, $LSTM^l$ and $LSTM^r$. $LSTM^l$ reads the passage from left to right and computes the hidden state h_i^l corresponding to i-th word w_i, while $LSTM^r$ reads the passage in the opposite direction and computes a hidden state h_i^r.

$$h_i^l = \mathrm{LSTM^l}(h_{i-1}^l, \mathrm{emb}(w_i))$$
$$h_i^r = \mathrm{LSTM^r}(h_{i+1}^r, \mathrm{emb}(w_i))$$

Here, $\mathrm{emb}(w_i)$ denotes the embedding vector for w_i, which is learned during training. Note that, for computing h_i^l, $LSTM^l$ looks at h_{i-1}^l, which is a hidden state for the left neighbor word w_{i-1} of w_i. This means that $LSTM^l$ reads the passage from left to right. In the same way, $LSTM^r$ looks at the hidden sate h_{i+1}^r for the right neighbor for computing h_i^r and $LSTM^r$ reads the passage from right to left. The *initial* hidden states for this recursive computation are set as $h_0^l = 0$ and $h_{|p|+1}^r = 0$. The hidden state h_i of the whole bidirectional LSTM for w_i is defined as the concatenation of two states, h_i^l and h_i^r, i.e., $h_i = \left[h_i^l, h_i^r\right]$.

The decoder computes the probability distribution $P_{\mathrm{vocab}}(y_t)$ of the t-th word y_t in the summary from the hidden state of the decoder s_t, the *context vector* h_t^* and the *decoder input* x_t, as shown below. The *decoder input* x_t is calculated from the previous *context vector* h_{t-1}^* and the previous output word y_{t-1}. At the test time, the previous word y_{t-1} is a generated word for the summary, while $t-1$-th word in the gold standard summary is given as y_{t-1} during training. In both of summarization and training, a special symbol, which indicates the beginning of the text, is used as the initial input word y_0.

$$P_{\mathrm{vocab}} = \mathrm{softmax}\left(V'\left(V\left[s_t, h_t^*\right] + b\right) + b'\right)$$

Here, P_{vocab} is a vector whose dimension is the vocabulary size. (We denote the probability of w in the vector by $P_{\mathrm{vocab}}(w)$.) Each element in it is a probability of a corresponding word. V and V' are trainable matrices and b and b' are trainable vectors. The *context vector* h_t^* is computed using attention distribution a_i^t (Bahdanau, Cho, and Bengio 2014) and a hidden state h_i of the encoder, given trainable matrices W_h, W_s and vector b_{attn}.

$$h_t^* = \sum_i a_i^t h_i$$
$$a^t = \mathrm{softmax}(e^t)$$
$$e_i^t = v^\top \tanh(W_h h_i + W_s s_t + b_{\mathrm{attn}})$$

s_t is computed by a decoder LSTM $LSTM^d$.

$$s_t = \mathrm{LSTM^d}(s_{t-1}, x_t)$$

The initial state s_0 of $LSTM^d$ can be the concatenated vector $\left[h_{|p|}^l, h_1^r\right]$, where $h_{|p|}^l$ and h_1^r are the *final* hidden states of the encoder LSTMs $LSTM^l$ and $LSTM^r$. However, the original implementation of the pointer-generator network[1]

[1] https://github.com/abisee/pointer-generator

used the following initial state, which has half the dimension of the above concatenated vector, where W_r is a trainable matrix, b_r is a trainable vector, and ReLU is a Rectified Linear Unit (Nair and Hinton 2010).

$$s_0 = \mathrm{ReLU}\left(W_r\left[h^l_{|p|}, h^r_1\right] + b_r\right)$$

We also use this initial state for our networks.

Actually, if we generate a summary according to the distribution P_{vocab}, this is a basic sequence-to-sequence model with attention mechanism (Bahdanau, Cho, and Bengio 2014). A problem of this model is that it can only generate words in its pre-defined lexicon, which are covered by training data and for which the vector P_{vocab} contains probabilities. Our task, like the general summarization task, needs to generate rare words including technical terms, but it is difficult to cover such words in training data. The pointer-generator network was developed to avoid this problem by *copying* such rare words in the original passage to a summary. More precisely, the probability distribution $P(y_t)$ of the word to be generated is defined as following,

$$P(y_t) = P_{\mathrm{gen}}P_{\mathrm{vocab}}(y_t) + (1 - P_{\mathrm{gen}}) \sum_{i:w_i=y_t} a^t_i$$

$$P_{\mathrm{gen}} = \sigma\left(W^\top_{h^*}h^*_t + W^\top_s s_t + W^\top_x x_t + b_{\mathrm{ptr}}\right)$$

where $W^\top_h$, $W^\top_s$, $W^\top_x$ are trainable matrices and b_{ptr} is a trainable vector and σ is the sigmoid function. When P_{gen} is small, y_t, a word to be generated is likely to be a copy of w_i in the passage. Otherwise, y_t is chosen just like a normal sequence-to-sequence model with attention mechanism.

3.2 Proposed Model and Alternative Model

The idea of our extension to the pointer-generator network, which we call *question word marking*, is simple. To give information about question to the network, we mark each word in the passage p according to whether it appears in the question $q = \langle q_1, q_2, ..., q_m\rangle$. More precisely, we modify the input to the encoder LSTMs as,

$$h^l_i = \mathrm{LSTM^l}(h^l_{i-1}, \tanh(W^\top_q\,[\mathrm{emb}(w_i), f(w_i, q)])$$

$$h^r_i = \mathrm{LSTM^r}(h^r_{i+1}, \tanh(W^\top_q\,[\mathrm{emb}(w_i), f(w_i, q)])$$

W_q is a trainable matrix, and $f(w, q)$ is defined as follows, where v_{qw} and v_{nqw} are d-dimensional learnable vectors and they represent the marks for the words in question q and the other words. Note that $d = 32$ in our experiments.

$$f(w, q) = \begin{cases} v_{\mathrm{qw}} & \text{if } w \text{ is a content word in } q \\ v_{\mathrm{nqw}} & \text{otherwise} \end{cases}$$

Question word marking is similar to the use of answer position features (Zhou et al. 2017), which were introduced to generate factoid questions given passages and answers in the passages. The authors of the paper (Zhou et al. 2017) proposed to mark an answer in a passage using the BIO scheme, i.e., label B marks the beginning of an answer, label I continues the answer and label O is given to the words that are not part of the answer. Then the labels are converted to real value vectors and given to the encoder. This approach differs however from question word marking in the type of words marked: answers to factoid questions in (Zhou et al. 2017), and words in a question in our work. Also, as noted in the Introduction, the words in a why-question may be scattered over an answer passage even if it is correct answers. Hence, whether the answer position features works well for why-QA is not a trivial question.

An alternative way to give an information from a question to the model is to use a *query attention model* proposed by (Nema et al. 2017), which was designed for the task of generating summaries relevant to given queries. They modified the probability distribution P_{vocab}

$$P_{\mathrm{vocab}} = \mathrm{softmax}(V'(Vs_t + V_{\mathrm{dec}}d_t))$$

and d_t is computed using the hidden states in another LSTM, which is added to the network to read a given query, as well as the hidden states for the passage like the previous models.

By regarding our why-question as its query, this model can be used for our task. The difference from our method is that the information of the question is not given to the passage encoder like ours. Actually, our experiments revealed that this query attention does not give performance improvement. Our compact answers include the words in a question rarely, while in the original query-base summarization the words in queries seems to appear in gold-standard summaries frequently. This difference may have lead to the ineffectiveness of the query attention model in our task.

3.3 Automatic Training Data Generation

Another idea for improving answer generation is to use automatically generated training data in addition to the manually created training data. Since our answer passages are relatively long, they can contain several causal relations in various forms. In preliminary experiments, the model trained using only manually created data was often confused by those multiple causal relations: it tended to copy to the output answer a word sequence around clue terms that marks causal relations such as "because," regardless of its relation or relative position to the topic of a given question. As a result, the resulting answer was a cause of some topic, but the topic was often different from that of the question.

To avoid this problem, as shown in Figure 2, we automatically generate training data by artificially creating passages as combinations of several causal relations. We first automatically extract expressions representing causal relations out of a large-scale web archive, and create pairs of a why-question (e.g., "Why does a tsunami occur?") and an answer (e.g., "Because the ocean's water mass is displaced by an earthquake.") out of the cause part and the effect part from the causality expressions (e.g. "$[\text{Tsunami occurs}]_{effect}$ because $[\text{the ocean's water mass is displaced by an earthquake}]_{cause}$"). We then create passages by merging three causality expressions that are randomly chosen from the set of the causality expressions extracted from the web archive (see Figure 2) and combine the auto-generated passage with each pair of the question and answer generated from one of the causality expression. Since each passage contains three causality expressions, we can create three passage-question-answer triples from it.

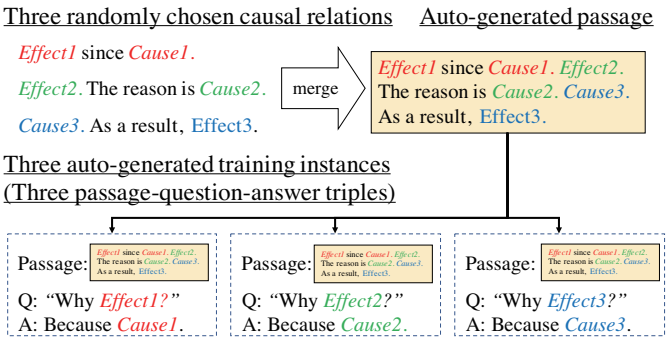

Figure 2: Automatic generation of training data

We applied our implementation of an automatic causality recognizer proposed in (Oh et al. 2013) to 4 billion Japanese web pages to extract 101 million causality expressions. The recognizer is a sequential tagger using CRF (Lafferty, McCallum, and Pereira 2001) that assigns BIO labels for each instance of cause and effect in the sentences. It can recognize inter-sentential causal relations, such as "A huge tsunami occurred. This is because the ocean's water mass was displaced by the Great East Japan Earthquake." On the other hand, the input is limited to the sentences that match the predefined regular expressions including the clue keywords, such as "because" and "reason," that are related to causality. The cause and effect parts recognizable by our recognizer are also limited to the consecutive word sequences. Although such limitations could be harmful, our empirical evaluation revealed that the training data generated using those causal relations is actually useful.

The causality recognizer achieved a precision of 83.8%, a recall of 71.1%, and an F-measure of 77.0% in a 10-fold cross-validation using a gold-standard dataset of 16,051 entries in which the cause parts and the effect parts were marked by a human annotator. In the evaluation, we regarded a causal relation as correctly recognized if and only if the automatically recognized cause and effect parts had overlap with the gold-standard cause and effect parts, respectively. In this paper's experiments, we used all the gold-standard data as training data.

4 Experiments

This section describes our datasets, the parameters of our neural model, and reports the experimental results.

4.1 Data

Our dataset is a set of triples of a why-question, an answer passage and a compact answer. First, we created our training, development and validation sets. Our human annotators manually created *open-domain* why-questions. We gave them to a publicly available Web-based QA system, WISDOM X (Mizuno et al. 2016), which has an open-domain why-QA module (Oh et al. 2016). The system retrieves answer passages from a 4 billion page-scale Web archive and provides a ranked list of answer passages for each question.

- A compact answer must not be longer than 25 Japanese characters.
- The content words in a compact answer must be included in the passage, but the conjugated forms of verbs can be changed so that the resulting compact answer appears natural, and functional words that do not appear in the passage can be used freely.
- The end of the compact answer must be the Japanese word "*tame*(because)."
- Pronouns must not appear in a compact answer. If the part of the passage that corresponds to a compact answer includes pronouns, they must be replaced with their referents in the passage.

Table 2: Rules for creating compact answers

Each answer passage consists of five or seven consecutive sentences. Note that those passages are limited to the ones that have one of several pre-defined clue terms that indicate the existence of causal relations, such as "because." Also note that there is no guarantee that a given question has something to do with the causal relations linked to those clue terms. There are cases where the question should be matched to another causal relation that is implicitly expressed in the passage without such clue terms, like "The large earthquake occurred and a tsunami attacked the coast." where *the large earthquake* should be regarded as the cause of *the tsunami's attack*.

We chose the top 20 answer passages in the ranked list for each question. Then, we asked human annotators to create a compact and non-redundant answer from each passage that provides a correct answer to the question according to the rules given in Table 2. Though each passage was given to three annotators, since they might not agree on whether compact answers could be written based on the passages, there were passages for which three compact answers were not obtained. After removing passages for which no annotators could write a compact answer, we obtained 19,639 question-passage-answer triples for 2,915 questions.

We randomly split the triples into a training set (15,310 triples for 2,060 questions), a validation set (2,271 triples for 426 questions), and a development set (2,238 triples for 429 questions) in such a way that no pair among the sets had common questions.

As for the test set, our annotators created 426 open-domain why-questions, and these were given to the same why-QA system. We selected the top three passages from all of the passages retrieved by the system and asked six annotators to create a compact answer for each passage according to the rules in Table 2. Then, the three best compact answers for each passage were selected by a vote of the same six annotators. This time, we did not remove passages without any compact answer, in order to assess the behavior of the methods when given noisy passages. In all, we obtained 2,339 passage-question-answer triples. The average length of all passages, questions, and compact answers in all the datasets were 188.0 words, 10.8 words, and 9.2 words, respectively. Note that the compact answers were much shorter than the answer passages. This also suggests the difficulty of our task.

4.2 Evaluation using the ROUGE scores

We tested the following variants of the proposed methods.

Proposed The proposed method (i.e., a pointer-generator network (hereafter, PG) with the question word marking) that was trained only with the manually created training data (MCD).

Proposed+N Proposed method trained using N automatically generated training instances on top of the MCD.

Proposed+N-MCD Proposed method trained using N automatically generated training instances only, without using the MCD.

Since the performance of **Proposed+N** and **Proposed+N-MCD** depends on the size of the auto-generated training data, we tried 250K, 500K, 750K, and 1M as values for N. As baselines to be compared with the proposed methods, we tested the following methods.

Passage A baseline method that provides a whole passage as a *compact* answer.

CRF This method provides, as a compact answer to a question, among the causal relations obtained by applying the causality recognizer to a passage (See Section 3), the cause part of the causal relation whose effect part has the largest word overlap with the question.

PG Original pointer-generator network. This does not use any information concerning the question.

PG+QAtt Pointer-generator network with the query attention model. The information concerning a given question is used only through the query attention mechanism described in Section 3.2, and the question word marking in our proposed method is not used.

DrQA An open-domain neural question answering system (Chen et al. 2017), which achieved a high performance against the SQuAD dataset (Rajpurkar et al. 2016). We trained this method only with the MCD.

Note that **Passage** and **CRF** are non-neural baselines but all the other methods utilize neural networks. Among many methods applicable to the SQuAD, we chose **DrQA** because the source code written by the authors of the paper was publicly available[2] and it works without a parser, which may cause problems in applying the methods to Japanese. We modified the system so that it can deal with Japanese and our task (details given later). All the methods were given the text inputs tokenized by the Japanese morphological analyzer MeCab[3]. The neural methods except for **DrQA** were implemented in PyTorch[4] on the top of OpenNMT-py[5] and we ran the experiments on Tesla M40 GPGPUs.

The results in terms of ROUGE scores are presented in Table 3. We showed the results of **Proposed+N-MCD** only for the best performing setting, with $N = 250K$. The parameters for each neural methods were tuned according to the ROUGE-1 score on the development set, as described later. The table presents results on the test set obtained with the tuned parameters, along with those on the development set (inside parentheses). **Proposed+250K** achieved the best performance on the test set among those of all the methods. All the methods using the auto-generated training data consistently showed more than 5% improvement over their versions that do not use the auto-generated data in all types of ROUGE scores. However, in the case where we trained the model with only the auto-generated data, **Proposed+N-MCD**, the performance drops considerably. These observations suggest that the auto-generated training data is effective, but manually created data is also indispensable.

Another important point is that **Proposed** gave a better result than **PG** although both methods were trained using the same manually created data. Since **Proposed** can be seen as **PG** with the question word marking, this suggests that the marking is effective. On the other hand, **PG+QAtt** gave a worse result than that of **PG**, which suggests that the query attention is not effective for this task.

For all the neural methods except for **Proposed+N**, **Proposed+N-MCD** and **DrQA**, we tuned the vocabulary set for LSTMs and the dimensions of the word embedding and LSTM hidden states according to the ROUGE-1 score on the development set. We tried three combinations of dimensions for the word embedding and the hidden states, namely (128, 128), (128, 256) and (256, 256). We tried two settings for the vocabulary size. In one setting, we set the vocabulary set of both the encoder and decoder to the 50,000 words that appeared most frequently in all the passages of the training set. In the other setting, we used the same set for the encoder but for the decoder, we used the 50,000 words that appeared most frequently in the answers within the training data. We also tested the pointer-generator networks with the coverage method (See, Liu, and Manning 2017) and those without it. Table 3 shows the performance of the networks that achieved the best performance against the development set during these trials. For **Proposed+N**

[2] https://github.com/facebookresearch/DrQA

[3] http://taku910.github.io/mecab/

[4] http://pytorch.org

[5] https://github.com/OpenNMT/OpenNMT-py

Method	QWM	Training data	Query-Att.	ROUGE-1	ROUGE-2	ROUGE-L	Avg.Length (words)
Non-neural Baseline Methods							
Passage	-	-	-	9.25(10.1)	7.34(7.94)	9.02(9.87)	180
CRF	-	-	-	32.7(34.3)	25.5(26.6)	32.0(33.6)	17.6
Neural Baseline Methods							
PG	no	MCD	no	36.5(40.7)	19.0(21.9)	36.0(40.1)	7.6
PG+QAtt	no	MCD	yes	36.1(40.4)	18.4(21.5)	35.4(39.8)	8.1
DrQA	-	-	-	24.7(28.6)	15.4(18.6)	23.4(27.6)	12.2
Proposed Methods							
Proposed	yes	MCD	no	38.2(43.5)	21.6(25.1)	37.8(43.0)	7.5
Proposed+250K	yes	MCD+AGTD(250K)	no	**44.9**(48.8)	**31.0**(33.6)	**44.4**(48.2)	9.5
Proposed+500K	yes	MCD+AGTD(500K)	no	44.1(48.6)	30.1(33.4)	43.5(48.0)	9.6
Proposed+750K	yes	MCD+AGTD(750K)	no	43.7(**49.4**)	29.9(**34.2**)	43.2(**48.7**)	9.5
Proposed+1M	yes	MCD+AGTD(1M)	no	44.4(48.7)	30.8(33.7)	43.9(48.1)	9.7
Proposed+250K-MCD	yes	AGTD(250K)	no	28.6(32.5)	20.7(23.9)	27.8(31.5)	26.4

QWM stands for the Question Word Marking. MCD and AGTD stand for Manually Created training Data and Auto-Generated Training Data. The figures inside the parentheses are the scores on the development set.

Table 3: ROUGE scores

and **Proposed+N-MCD**, we used the same parameters and choice as those for **Proposed**.

As for the other parameters, we used the following values in all of the neural methods, except for **DrQA**. The optimizer is adagrad (Duchi, Hazan, and Singer 2011). We applied early stopping using the validation data and the maximum number of epochs is 10. The batch size, maximum gradient norm, learning rate and beam size for beam search of the decoded results are 16, 2.0, 0.01 and 4, respectively. The coverage loss weight is 1.0 if applicable. Also note that, unlike See, Liu, and Manning, we did not impose any restriction on the length of the input passage.

Though **DrQA** works quite well for the SQuAD dataset, it achieved significantly worse performance than those of our proposed methods. We tried to modify the method[6], change the parameter values in the default setting[10] and even weaken the format/definition of the correct compact answers so that it could deal with Japanese and execute our task. The peformance figure in Table 3 is the best one obtained during the trials. The biggest problem surely was the difference of the format of answers. While, in the SQuAD dataset, an answer is a consecutive text span in a passage, in our dataset an answer may not match any text span in the passage because the annotators created the answers rather freely. For training, we extracted all the content words in answers in our data and computed the shortest span in a passage that cover all the content words. The obtained text span was used as gold standard for training, although it can be reudndant as a compact answer and may not be natural Japanese text. For testing, we tried two methods for creating the gold standard: (i) the same method as that for the training, and (ii) keeping the original answer except for the Japanese word "*tame* (because)", which the annotators must add regardless whether it appears in a passage or not. Also, we ignored mismatch of conjugated forms when calculating the ROUGE scores for **DrQA**. In spite of such efforts, we could not achieve performance comparable to those of our proposed methods. This suggests that our task is not a trivial variant of the SQuAD task, although we have to admit that there is also a chance that the worse performance is due to the language difference.

4.3 Subjective Evaluation

We also had human annotators evaluate the quality of the output answers. We gave 300 questions manually created by annotators to the why-QA system used for the ROUGE score evaluation and applied **Proposed**, **Proposed+750K** and **PG** to the top-ranked output passage of each question. We chose **Proposed+750K** because it gave the best ROUGE score on the development set. First, we asked three human annotators to assess the *fluency* of the answers by labelling each of them according to the following: *Good* if the answer is natural for Japanese; *Acceptable* if the answer is not natural but the semantic content is understandable; and *Unacceptable* in all other cases. Table 4 presents the total number of each type of label given by the annotators to the answers provided by each method, along with its ratio for each method. (Since we asked three annotators to assess each answer, each method

[6]We stopped to use a syntactic parser in **DrQA** because the performance achieved without parsers on the SQuAD dataset is actualy comparable to the best performance achieved with a parser in our environment when the default setting and word embeddings distributed from Stanford University [7]are used ($EM = 69.4$, $F1 = 78.9$ vs. $EM = 69.4$, $F1 = 78.7$). We also modified the setting of **DrQA** so that it uses MeCab [8], which is a morphological analyzer for Japanse, as a tokenizer and the pre-trained Japanese word embedding that are distributed from Facebook Research. [9]We also disabled pos, named entity recognition and lemma features. Note that our proposed methods used none of them.

[7]https://nlp.stanford.edu/projects/glove/

[8]http://taku910.github.io/mecab/

[9]https://github.com/facebookresearch/fastText/blob/master/pretrained-vectors.md

[10]We tried changing some parameters in the default settings of **DrQA**. We changed the dimension of hidden states from 256 to 128, the maximum number of words to be genered by an ecoder from 15 to 30, the learning rate from 0.1 to 0.01 (the same value for the proposed methods) and the number of LSTM's layers from 3 to 1. We also tried the version that does not use the pre-trained word embeddings.

has 900 labels for 300 answers.) Fleiss' Kappa (Fleiss 1971) among the three annotators was 0.64, which shows moderate agreement. Note that **Proposed+750K** achieved better results than those of the other methods, and **Proposed** worked better than **PG**. This suggests that our methodologies are effective for improving the fluency.

As a second subjective evauation, we then asked three human annotators to assess the *informativeness* of the answers, given only the questions and the compact answers and not the passages. This means that this evaluation is about whether the answers are *convincing* enough without the help of external contexts, i.e., the passages. This evaluation was designed for assessing whether the answers were sufficiently understandable in situations such as the use of a spoken dialog systems, where the users cannot access external contexts.

The results of informativeness are presented in Table 5. Each annotator gave to each instance one of three labels: *Good* if and only if a given compact answer contains enough elements that should appear in a correct answer to the question, and is recognizable and comprehensible as a correct answer; *Acceptable* when the answer does not contain some elements required in a correct answer but those elements can be inferred according to commonsense knowledge, and the correct answer can be guessed based on the given answer and the inference; and *Unacceptable* otherwise. Table 5 provide the total number of each label and its ratio. Fleiss's Kappa for assessing the inter-annotator agreement was 0.45, which indicates moderate agreement.

As can be seen from Table 5, **Proposed+750K** outperformed the other methods significantly. This is consistent with the evaluation using the ROUGE scores. On the other hand, the ratio of the *Good* labels is just 24%. To check

Models	Good	Accept	Unaccept
PG	474(52.7%)	325(36.1%)	101(11.2%)
Proposed	551(61.2%)	242(26.9%)	107(11.9%)
Proposed+750K	591(65.7%)	266(29.6%)	43(4.8%)

Table 4: Subjective evaluation results of fluency

Models	Good	Accept	Unaccept
PG	132(14.7%)	154(17.1%)	614(68.2%)
Proposed	154(17.1%)	141(15.7%)	605(67.2%)
Proposed+750K	217(24.1%)	148(16.4%)	535(59.4%)

Table 5: Subjective evaluation results of informativeness

whether this low ratio is due to the low quality of the passages retrieved by the why-QA system, we conducted a third subjective evaluation, in which all the answers given *Unacceptable* labels by all three annotators in the previous evaluation were removed from the dataset, based on the assumption that many of such unacceptable answers were generated from noisy passages. Also, to reduce the deviation of the labelling due to a lack of knowledge of the annotators, at this time, we gave the answer passages alongside the compact answers and why-questions. Here, *Good* and *Acceptable* can be given only when the passage supports the validity of the answers. In a sense, this evaluation gives an assessment of the informativeness where external contexts, i.e., the previous evaluation and passages, are given.

The results of this additional evaluation of informativeness are presented in Table 6. Fleiss's Kappa was 0.60, which indicates moderate agreement. The ratio of *Good* labels increased for all the methods and that of **Proposed+750K** reached 44%. Although this is still not a satisfactory figure, we believe that it is reasonably high as a result of the first attempt for this task. Also, this suggests the possibility that the quality of the compact answers can be enhanced by improving the why-QA systems used for retrieving passages. Another important point is that the number of *Good* labels increased: this suggests that some answers were not interpreted as good answers in the previous evaluation due to a lack of context or knowledge, but were understood as correct ones with the help of the passages. This suggests the need for another method which gives concise supplementary information when the user of a spoken dialog system cannot understand compact answers due to a lack of knowledge. Such highly intelligent functionality should be indispensable in future dialog systems and may be an interesting future work.

Models	Good	Accept	Unaccept
PG	148(29.4%)	49(9.7%)	307(60.9%)
Proposed	177(35.1%)	39(7.7%)	288(57.1%)
Proposed+750K	262(44.1%)	39(6.6%)	293(49.3%)

Table 6: Subjective evaluation results of informativeness with external contexts

5 Conclusion

This paper described a method for generating compact and non-redundant answers to a given why-questions. Our method applies a novel neural text summarizer to the long answer passages retrieved from 4 billion Web pages by an existing why-QA system, WISDOM X (Mizuno et al. 2016; Oh et al. 2016), and can generate compact answers with a length of around 10 words in average. To improve the quality of the compact answers, we attempted semi-distant supervision. We exploited a large amount of automatically generated training data as well as a relatively small amount of manually created training data.

As a future work, we are considering to use a wider range of auto-generated training data. We expect that various automatic causality recognizers (Hashimoto et al. 2012; 2014; Kruengkrai et al. 2017) would be useful for this purpose. Another direction is to use the attention mechanisms that can pay attention to causally associated word pairs such as "cigarettes" and "cancer" (Oh et al. 2017), although our attempts in this direction have not lead to better results so far.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2014. Neural machine translation by jointly learning to align and translate. *CoRR* abs/1409.0473.

Chen, D.; Fisch, A.; Weston, J.; and Bordes, A. 2017. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, 1870–1879.

dos Santos, C. N.; Tan, M.; Xiang, B.; and Zhou, B. 2016. Attentive pooling networks. *CoRR* abs/1602.03609.

Duchi, J. C.; Hazan, E.; and Singer, Y. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12:2121–2159.

Fleiss, J. L. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin* 76(5):378–382.

Girju, R. 2003. Automatic detection of causal relations for question answering. In *Proceedings of the ACL 2003 Workshop on Multilingual Summarization and Question Answering*, 76–83.

Hashimoto, C.; Torisawa, K.; Saeger, S. D.; Oh, J.; and Kazama, J. 2012. Excitatory or inhibitory: A new semantic orientation extracts contradiction and causality from the web. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, 619–630.

Hashimoto, C.; Torisawa, K.; Kloetzer, J.; Sano, M.; Varga, I.; Oh, J.; and Kidawara, Y. 2014. Toward future scenario generation: Extracting event causality exploiting semantic relation, context, and association features. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, 987–997.

Higashinaka, R., and Isozaki, H. 2008. Corpus-based question answering for why-questions. In *Third International Joint Conference on Natural Language Processing (IJCNLP 2008)*, 418–425.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural Computation* 9(8):1735–1780.

Kruengkrai, C.; Torisawa, K.; Hashimoto, C.; Kloetzer, J.; Oh, J.; and Tanaka, M. 2017. Improving event causality recognition with multiple background knowledge sources using multi-column convolutional neural networks. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI 2017)*, 3466–3473.

Lafferty, J. D.; McCallum, A.; and Pereira, F. C. N. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML 2001)*, 282–289.

Mizuno, J.; Tanaka, M.; Ohtake, K.; Oh, J.; Kloetzer, J.; Hashimoto, C.; and Torisawa, K. 2016. WISDOM X, DISAANA and D-SUMM: large-scale NLP systems for analyzing textual big data. In *26th International Conference on Computational Linguistics, Proceedings of the Conference System Demonstrations (COLING 2016)*, 263–267.

Nair, V., and Hinton, G. E. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on Machine Learning (ICML 2010)*, 807–814.

Nema, P.; Khapra, M. M.; Laha, A.; and Ravindran, B. 2017. Diversity driven attention model for query-based abstractive summarization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, 1063–1072.

Oh, J.; Torisawa, K.; Hashimoto, C.; Kawada, T.; Saeger, S. D.; Kazama, J.; and Wang, Y. 2012. Why question answering using sentiment analysis and word classes. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, 368–378.

Oh, J.; Torisawa, K.; Hashimoto, C.; Sano, M.; Saeger, S. D.; and Ohtake, K. 2013. Why-question answering using intra- and inter-sentential causal relations. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, 1733–1743.

Oh, J.; Torisawa, K.; Hashimoto, C.; Iida, R.; Tanaka, M.; and Kloetzer, J. 2016. A semi-supervised learning approach to why-question answering. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI 2016)*, 3022–3029.

Oh, J.; Torisawa, K.; Kruengkrai, C.; Iida, R.; and Kloetzer, J. 2017. Multi-column convolutional neural networks with causality-attention for why-question answering. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (WSDM 2017)*, 415–424.

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, 2383–2392.

Rush, A. M.; Chopra, S.; and Weston, J. 2015. A neural attention model for abstractive sentence summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP 2015)*, 379–389.

See, A.; Liu, P. J.; and Manning, C. D. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL 2017)*, 1073–1083.

Sharp, R.; Surdeanu, M.; Jansen, P.; Clark, P.; and Hammond, M. 2016. Creating causal embeddings for question answering with minimal supervision. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*, 138–148.

Tan, M.; dos Santos, C. N.; Xiang, B.; and Zhou, B. 2016. Improved representation learning for question answer matching. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, 464–473.

Verberne, S.; van Halteren, H.; Theijssen, D.; Raaijmakers, S.; and Boves, L. 2011. Learning to rank for *why*-question answering. *Inf. Retr.* 14(2):107–132.

Zhou, Q.; Yang, N.; Wei, F.; Tan, C.; Bao, H.; and Zhou, M. 2017. Neural question generation from text: A preliminary study. *CoRR* abs/1704.01792.